www.volsu.ru

МОДЕЛИРОВАНИЕ,
ИНФОРМАТИКА И УПРАВЛЕНИЕ

DOI: https://doi.org/10.15688/mpcm.jvolsu.2022.1.3

УДК 004.91, 81'33, 004.42
ББК 32.973, 81.1

Дата поступления статьи: 29.12.2021
Дата принятия статьи: 02.02.2022

О СОЗДАНИИ ВЕБ-СЕРВИСА ДЛЯ РАБОТЫ С КОРПУСОМ АРХИВНЫХ ДОКУМЕНТОВ[1]

Артур Валерьевич Павлов

Студент института математики и информационных технологий,
Волгоградский государственный университет
art.val.pavlov@yandex.ru, matf@volsu.ru
просп. Университетский, 100, 400062 г. Волгоград, Российская Федерация

Юлия Дмитриевна Сапич

Студент института математики и информационных технологий,
Волгоградский государственный университет
yds2402@yandex.ru, matf@volsu.ru
просп. Университетский, 100, 400062 г. Волгоград, Российская Федерация

Андрей Владимирович Светлов

Кандидат физико-математических наук, доцент кафедры
математического анализа и теории функций,
Волгоградский государственный университет
a.v.svetlov@gmail.com, andrew.svetlov@volsu.ru, matf@volsu.ru
https://orcid.org/0000-0002-8764-6132
просп. Университетский, 100, 400062 г. Волгоград, Российская Федерация

Анатолий Сергеевич Комендантов

Инженер-программист,
ООО «СЭТ»
anatoly.komendantov@gmail.com
ул. Смолячкова, 4/2, 194044 г. Санкт-Петербург, Российская Федерация

Аннотация. Статья посвящена разработке веб-сервиса лингвистического корпуса документов архивного фонда «Михайловский станичный атаман». Компоненты сервиса позволяют производить автоматизированный морфологический анализ текстов, создавать на основе него документы, формирующие корпус, сохранять их в базе данных, производить поиск по ним и получать созданные другими пользователями документы. Также в программе предусмотрена функция для ручной коррекции ошибок, возникающих при проведении автоматизированного морфологического анализа старославянских текстов, в которых присутствуют устаревшие символы.

Ключевые слова: лингвистический корпус документов, веб-сервис, автоматизация морфологического анализа, утилита MyStem, корпусная лингвистика.

Введение

Настоящая работа выполнена в рамках проекта коллектива ученых ВолГУ под руководством профессора О.А. Горбань по созданию корпуса документов фонда «Михайловский станичный атаман». Вкратце напомним, что этот фонд содержит исторически ценные документы канцелярии Войска Донского XVIII–XIX вв., хранящиеся в Государственном архиве Волгоградской области. Для введения их в научный оборот проделана большая предварительная работа по их оцифровке, выполнена транслитерация текста с сохранением орфографии оригинала, при этом проведена адаптация текста — выносные буквы даны в строку, титла раскрыты, диграфы устранены, добавлены пробелы после предлогов, все пометки на полях были перенесены в сноски (см. [2; 3; 7–9]).

В нынешнем виде эти документы стали пригодны для компьютерной обработки без необходимости создания специализированных средств, корректно работающих, например, с выносными буквами и титлами — можно использовать существующие инструменты для обработки текстов. Единственная существенная проблема — устаревшая лексика и графика, но она была в целом решена в предыдущих работах, как и ряд других проблем [1; 5; 6; 9].

На текущем этапе основной задачей является разработка технической и программной части корпуса. Фактически это означает создание «движка» корпуса документов, то есть программного обеспечения, решающего задачи хранения базы данных размеченных текстов, выполнения запросов к этой базе, а также предоставления пользователям удобного интерфейса для работы, не требующего специальной квалификации в области информационных технологий. При этом в процессе работы над предыдущими задачами было принято решение интегрировать процесс разметки документов в общую логику работы программного обеспечения корпуса, а не выделять его в отдельное приложение с необходимостью унификации формата представления разметки.

Таким образом, представляемая статья посвящена разработке REST-сервиса, который позволяет производить автоматизированный морфологический анализ текстов, создавать на основе него документы, формирующие корпус текстов, сохранять их в базе данных, производить поиск по ним и получать созданные другими пользователями документы. Также в программе предусмотрена функция для коррекции ошибок, возникающих при проведении автоматизированного анализа старославянских текстов, в которых присутствуют устаревшие символы.

1. Концепция сервиса

Как было отмечено выше, приложение представляет собой REST-сервис. REST — архитектурный стиль взаимодействия компонентов распределенного приложения в сети. REST — это согласованный набор ограничений, учитываемых при проектировании распределенной гипермедиа-системы [11].

Одним из основных аспектов REST-приложений является клиент-серверная архитектура, что приводит к разделению процесса их разработки на front-end и back-end.

Front-end — клиентская сторона интерфейса, в основе которой лежит HTML, CSS, JavaScript. Является связующим звеном между пользователями и сервером. Back-end — программно-аппаратная часть сервиса, разработанная на основе языка Java, Java-фреймворка Spring (Spring Boot), Swagger (фреймворк для спецификации REST API), СУБД MongoDB, фреймворк для автоматизации сборки Apache Maven. В ней, в нашем случае, содержится логика обработки документов, авторизации пользователей, ответов на запросы от других компонентов. В целом архитектура нашего веб-приложения имеет вид, показанный на рисунке 1.

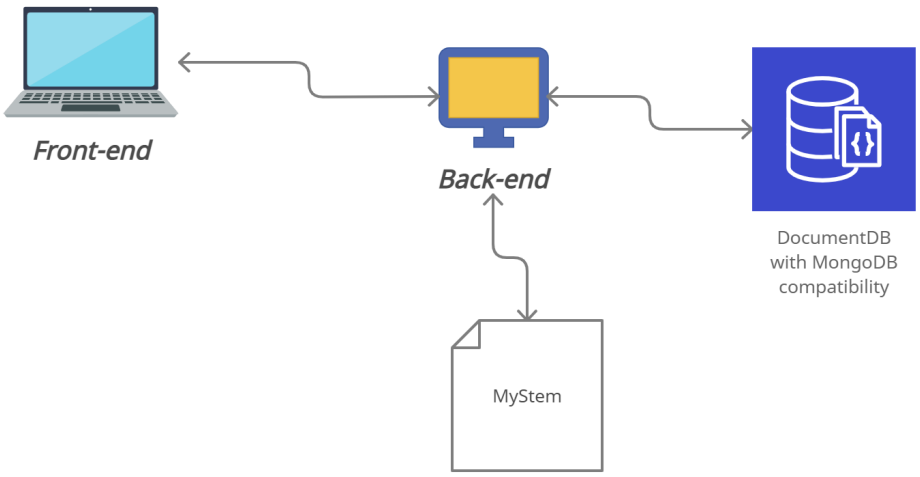

Рис. 1. Схема компонентов веб-приложения

Хранилищем для проанализированных документов, зарегистрированных пользователей и других необходимых для работы сервиса данных является облачная СУБД MongoDB Atlas. База данных (БД) представлена в виде физического хранилища коллекций. Каждая БД имеет свой собственный набор файлов в файловой системе. Обычно один MongoDB сервер имеет несколько БД. Коллекция — это группа документов MongoDB. Является эквивалентом простой таблицы в реляционной базе данных. Коллекция помещена внутри одной БД. Документ в коллекции может иметь различные поля. Чаще всего, все документы в коллекции созданы для одной, либо относящихся друг к другу целей. Документ — это набор пар «ключ — значение». Документ имеет динамическую схему. Это означает, что документ в одной и той же коллекции не обязан иметь один одинаковый набор полей или структуру, а общие поля в коллекции могут иметь различные типы данных. Помимо этого, существенным компонентом нашего приложения является MyStem И. Сегаловича [10] — консольная программа морфологического анализатора русского языка. Ранее было показано [5], что MyStem корректно

обрабатывает устаревшую лексику, если она передается в программу с использованием современной кириллицы. Поэтому в этом приложении мы продолжаем использовать метод адаптации текста для получения приемлемых результатов морфологической разметки посредством MyStem, вместо разработки каких-то новых инструментов для этой цели. Таким образом, сервис имеет опцию для работы с устаревшей кириллицей, при включении которой буквы зело, омега, и краткое и другие заменяются на «кс», «о» и «и» соответственно, и только после этого текст передается для анализа в MyStem, а затем в обработанном документе производится обратная замена символов (о разработке этого подхода см. [5]). Диаграмма прецедентов, описывающая весь функционал сервиса, представлена на рисунке 2.

Рис. 2. Варианты использования приложения для авторизованного пользователя

Отметим, что основным форматом данных в нашем приложении является JSON. Поэтому, используя MongoDB, нам не нужно проводить сложных преобразований данных из mystem, получаемых в запросах в том же формате JSON. Нам не требуется создавать сложную структуру БД, писать сложные запросы для вставки и выборки данных.

2. Описание интерфейса и принципов работы приложения

Структура сайта как интерфейса клиентской части сервиса приведена на рисунке 3. Работа пользователя с какими-либо функциями сервиса начинается с авторизации (получения токена), производящейся по логину и паролю. Функция authorization() получает значения логина и пароля из соответствующих полей и с помощью метода fetch(url, [options]) отправляет POST-запрос на сервер, на котором установлен наш back-end модуль. Блок-схема этой функции представлена на рисунках 4 и 5.

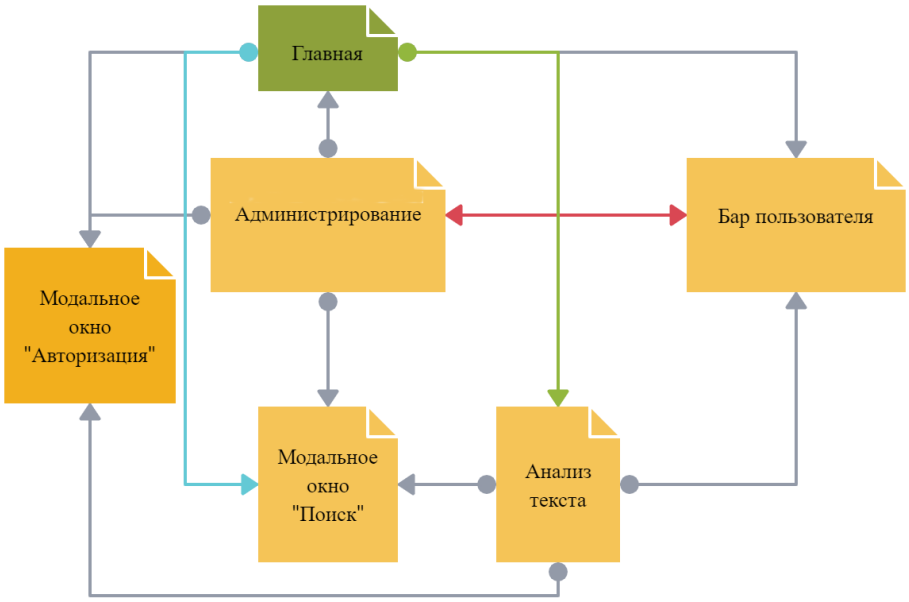

Рис. 3. Структура сайта

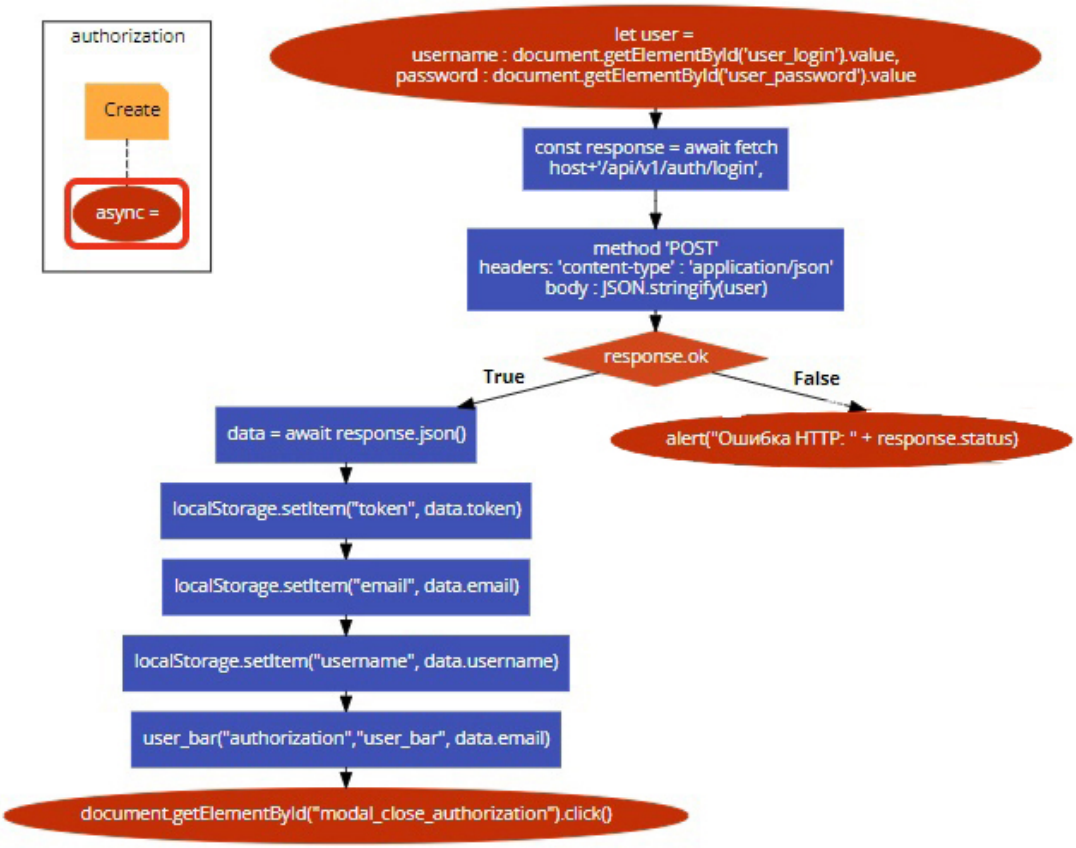

Рис. 4. Блок-схема функции authorization() с кодом

В глобальной переменной host указывается адрес сервера, к нему добавляется адрес конкретного ресурса сервера, в который должны попасть данные. Общение сервера с клиентом в основном происходит с помощью JSON-объектов, то есть неупорядоченных множеств пар «ключ: значение», заключенных в фигурные скобки {}, или целого массива таковых, заключенных в квадратные скобки []. Например, относительно функции authorization() нам известно, что по адресу /api/v1/auth/login back-end ожидает POST-запрос с JSON вида:

```
{
"password": "string",
"username": "string"
}.
```

Front-end в свою очередь собирает и формирует эти данные в ожидаемом виде и передает на сервер. После получения ответа от сервера остается только корректно его обработать и вывести пользователю.

Непосредственно морфологический анализ документа производится после загрузки пользователем документа со своего ПК с использованием стандартных элементов интерфейса, которые мы не будем подробно описывать. За обработку этого события отвечает функция filereader(). Она ожидает изменения входного потока с типом file, и как только это событие наступает, читает его содержимое в переменную string, а затем вызывает функцию text_analyse(string), которая отправляет запрос с текстом документа к серверу, и если он успешен, то front-end заменяет текущий контент страницы на ответ сервера.

Анализ текста производится посредством ресурса /api/v1/document/analyse, ожидаемые данные имеют вид:

```
{
"text": "string"
}.
```

На сервере за обработку текста отвечают несколько функций, одна из них по регулярному выражению удаляет лишние пробелы, заменяет устаревшие символы — все, что может помешать корректному определению слова. Затем текст передается в функцию, использующую ProcessBuilder для запуска MyStem. Для обработки текста динамически создается два временных файла, на вход и на выход работы программы. После завершения процесса алгоритм считывает текст из конечного файла и отправляет во front-end. В ответ ожидается массив из JSON-объектов analysis, где каждый элемент содержит слово из текста и все возможные варианты его морфологического разбора:

```
[{
"analysis": [
{
"gr": "string",
"lex": "string",
"qual": "string"
}
],
"text": "string"
}].
```

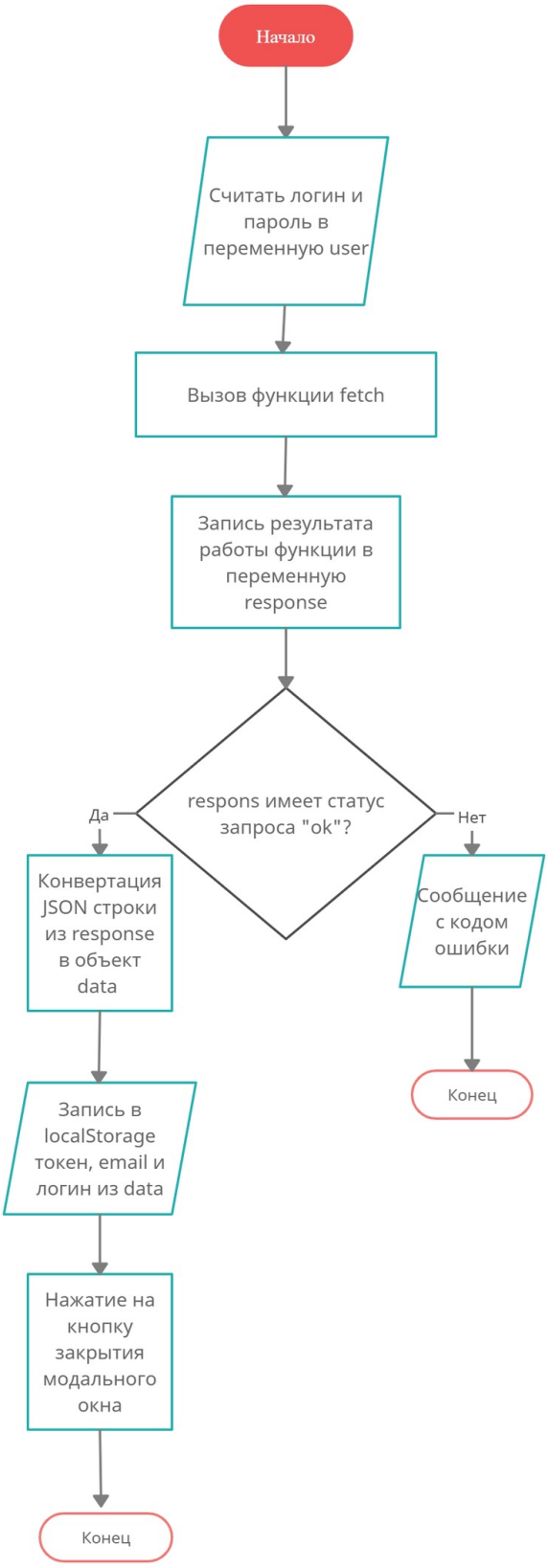

Рис. 5. Блок-схема функции authorization() с пояснениями

Отметим, что наше приложение предназначено для работы только с документами определенного формата, которые были подготовлены коллективом филологов в процессе предобработки рукописей архива. В частности, в оригинале исследуемых архивных документов еще отсутствовали какие-либо общепринятые способы деления текста на смысловые фрагменты — предложения. Это усложняет задачу создания корпуса таких текстов, потому что возникает вопрос о форме представления результатов поисковых запросов — обычно таким результатом является как раз предложение, содержащее найденные грамматические единицы. Вместо того чтобы выводить просто случайное окружение искомых слов, было принято решение провести искусственное разделение текстов на смысловые фрагменты в процессе их предобработки. Таким образом, документы, которые загружаются в базу данных, содержат специальный символ-разделитель, по которому производится фрагментация текста. Для пользователя документ после загрузки отображается уже в обработанном виде, разбитом визуально на такие искусственные предложения (рис. 6). Далее для каждого такого фрагмента создается объект details с элементом интерфейса select, причем этот элемент прикрепляется к каждому слову по отдельности и содержит список опций из всех вариантов морфологического разбора в атрибуте analysis (см. рис. 7).

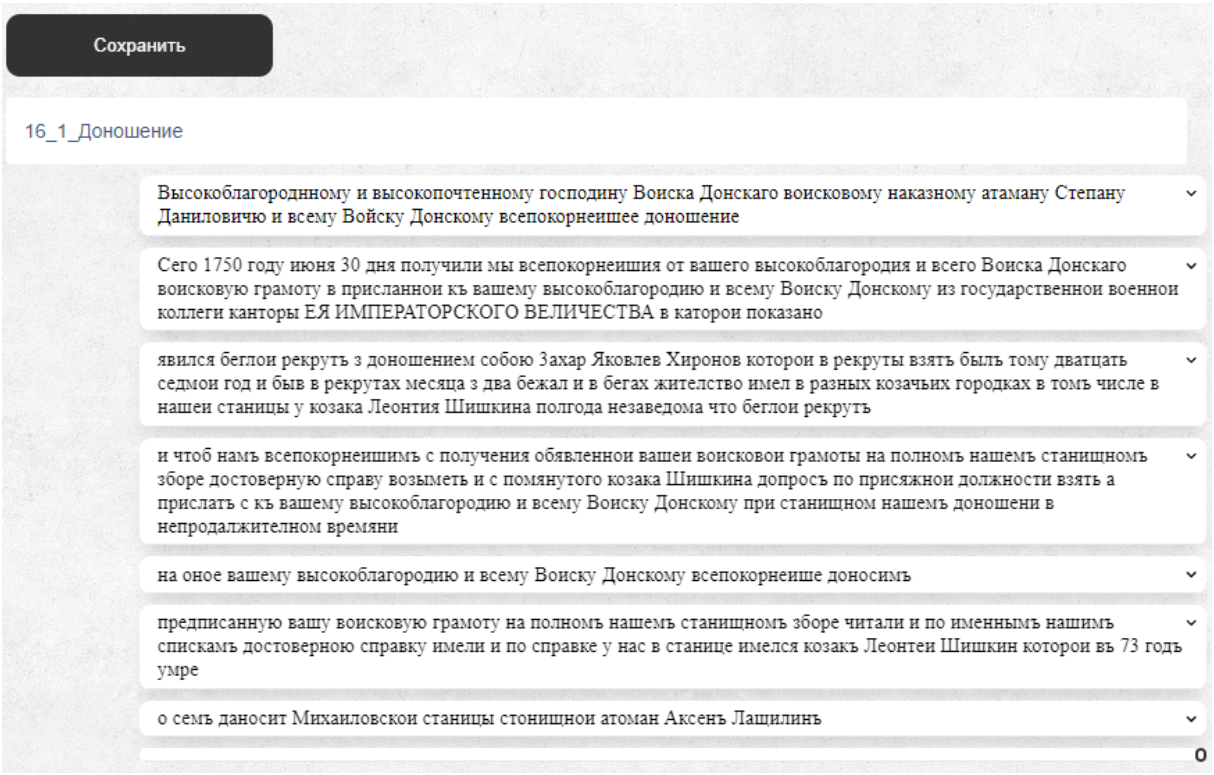

Рис. 6. Результат обработки документа

После того как пользователь обработал документ, по нажатию кнопки «Сохранить», функция analyse_collector() собирает со страницы все значения объектов типа select и отправляет их на сервер, формируя JSON-объект следующего вида:

```
"words": [
{ "analysis": {
```

```
"gr": "string",
"lex": "string",
"qual": "string"
},
"text": "string"
}].
```

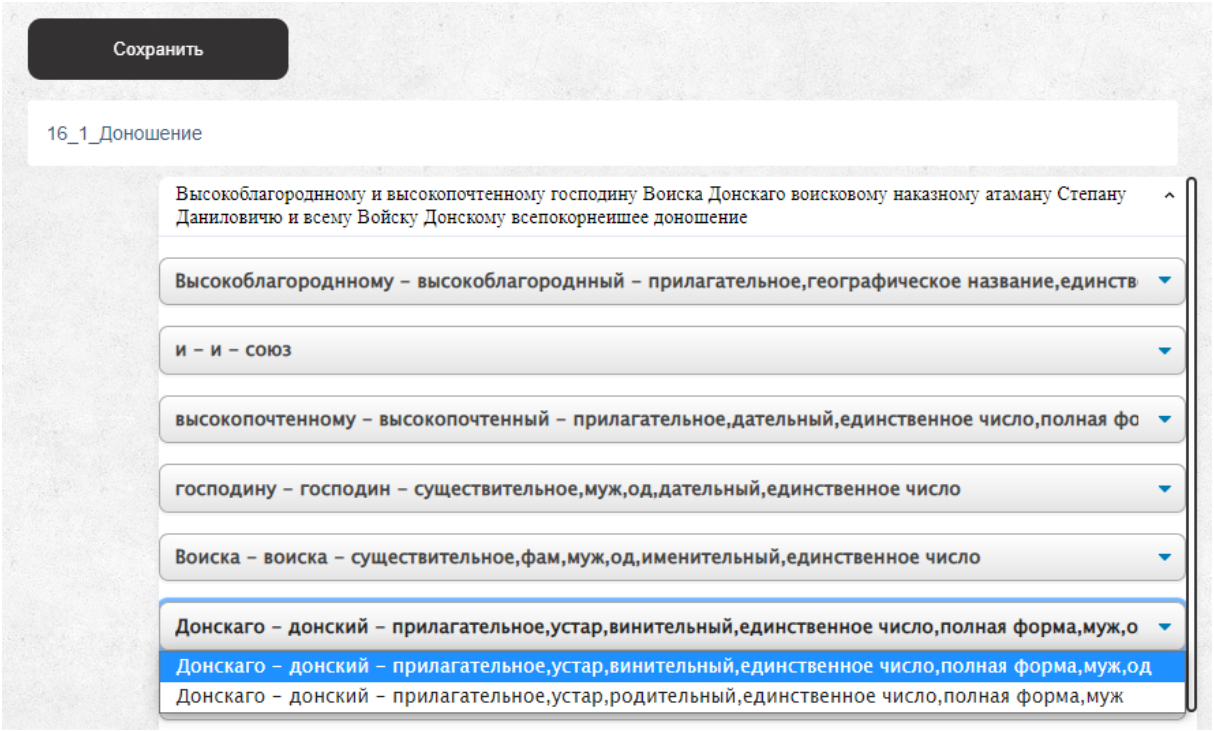

Рис. 7. Раскрытие элемента details с выбором варианта морфологического разбора

Сервер перенаправляет эти данные в MongoDB Atlas с помощью Spring Boot. Для того чтобы производить поиск по всем сохраненным в базу документам, необходимо обратиться к соответствующему модальному окну. Поиск производится по следующим критериям: название документа, содержимое, автор (пользователь, загрузивший документ) и морфологические признаки. Если с первыми элементами все довольно просто — это обычные текстовые поля, то морфологические признаки разбиты на группы, каждая группа в отдельном раскрывающемся элементе, в них находятся флажки, отвечающие каждый за свой признак (структура прописана в соответствии с документацией MyStem [4]) — далее производится выбор фрагментов документов, содержащих элементы с выбранными характеристиками (см. рис. 8).

По нажатию кнопки «Найти» производится вызов функции retrieveFormValue(), и она формирует «тело» запроса следующим образом:

```
{
"authorUsername": document.getElementById('author-username').value,
"gr": str,
```

```
"text": document.getElementById('document-text').value,
"title": document.getElementById('document-name').value
}.
```

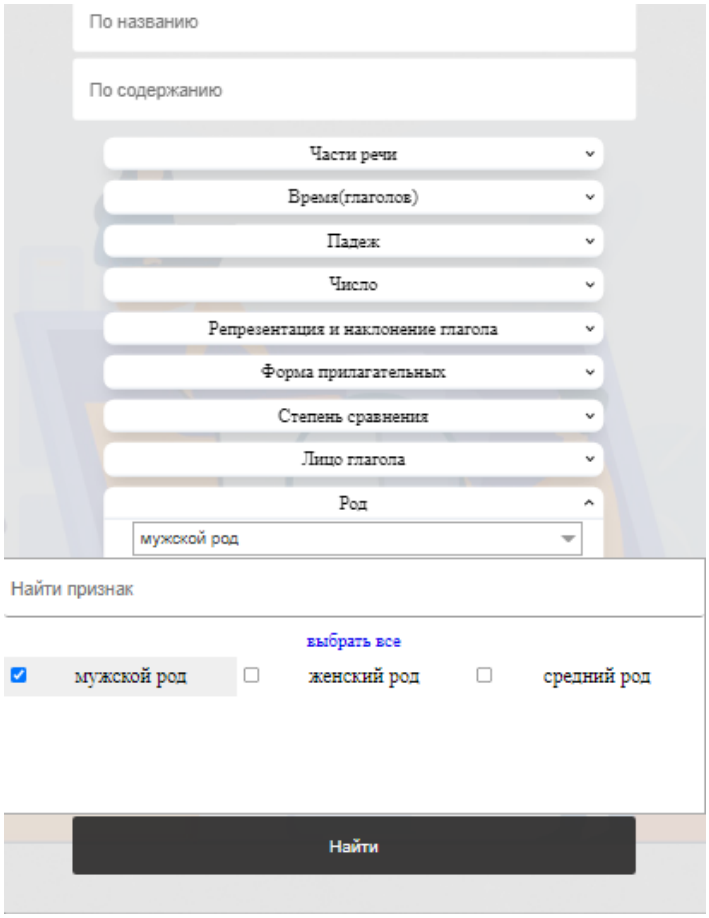

Рис. 8. Окно поиска с интерфейсом выбора морфологических признаков

Как можно заметить из структуры, значение текстовых полей получается по их идентификатору. Строка грамматических признаков str перед этим создается из массива, содержащего значения всех отмеченных флажков. В конце своей работы функция вызывает другую функцию — search(req_body). Она в свою очередь отправляет запрос на соответствующий ресурс на сервере. Если сервер вернет непустой ответ, то html-элемент, отвечающий за основной контент на странице, заменится результатами поиска. Преимущество такой архитектуры в том, что не нужно создавать отдельный html-файл для вывода результатов, а значит сама процедура происходит без переходов, обновлений или иных действий на любой странице веб-ресурса.

Для того чтобы продемонстрировать логику обработки ответа сервера, рассмотрим структуру получаемого объекта (см. рис. 9) и то, в каком виде он показывается пользователю сайта.

Стоит заметить, что результатами поиска являются не только слова-совпадения, но и некоторое их окружение. Ответ сервера составляет массив из объектов, где каждый объект — это окружение, содержащих искомое слово, также имеется название документа, которому принадлежит отрывок и другая информация.

```
▼0: {documentID: "61efd7856aba8c4ec03dc919", authorUsername: "User123", documentTitle: "22_1_Рапорт",…}
  authorUsername: "User123"
  ▼documentExcerpt: [,…]
    ▶0: {analysis: {lex: "высокоблагородин", gr: "ANUM,dat,sg,m", qual: "bastard"}, text: "высокоблагородному"}
    ▶1: {analysis: {lex: "и", gr: "CONJ", qual: "bastard"}, text: "и"}
    ▶2: {analysis: {lex: "высокопочтенный", gr: "A,dat,sg,plen,m", qual: "bastard"}, text: "высокопочтенному"}
    ▶3: {analysis: {lex: "господин", gr: "S,m,anim,dat,sg", qual: "bastard"}, text: "господину"}
    ▶4: {analysis: {lex: "воиска", gr: "S,famn,m,anim,nom,sg", qual: "bastard"}, text: "Воиска"}
    ▶5: {analysis: {lex: "донский", gr: "A,obsol,acc,sg,plen,m,anim", qual: "bastard"}, text: "Донскаго"}
    ▶6: {analysis: {lex: "воисковое", gr: "S,n,inan,dat,sg", qual: "bastard"}, text: "воисковому"}
    ▶7: {analysis: {lex: "атаман", gr: "S,m,anim,dat,sg", qual: "bastard"}, text: "атаману"}
    ▶8: {analysis: {lex: "данила", gr: "S,persn,m,anim,acc,sg", qual: "bastard"}, text: "Данилу"}
    ▶9: {analysis: {lex: "ефремович", gr: "S,patrn,m,anim,dat,sg", qual: "bastard"}, text: "Ефремовичу"}
  documentID: "61efd7856aba8c4ec03dc919"
  documentTitle: "22_1_Рапорт"
  markedWordIndex: 0
▶1: {documentID: "61efd78b6aba8c4ec03dc91a", authorUsername: "User123", documentTitle: "23_1_Доношение",…}
▶2: {documentID: "61efd7926aba8c4ec03dc91b", authorUsername: "User123", documentTitle: "39_1_Доношение",…}
▶3: {documentID: "61efd7996aba8c4ec03dc91c", authorUsername: "User123", documentTitle: "45_1_Рапорт",…}
▶4: {documentID: "61efd7a06aba8c4ec03dc91d", authorUsername: "User123", documentTitle: "55_1_Рапорт",…}
▶5: {documentID: "61efd7a66aba8c4ec03dc91e", authorUsername: "User123", documentTitle: "84_1_Рапорт",…}
```


Рис. 9. Структура объекта, возвращаемого сервером

Как видно из структуры объекта, каждый элемент содержит само слово, его первоначальную форму и перечень признаков. Алгоритм для каждого элемента массива создает конструкцию и добавляет к ней элемент <p> с текстом самого слова, где атрибут title содержит его морфологический разбор. Благодаря этому при наведении на каждое слово курсором мыши всплывает вся информация о нем. Если выбранные пользователем грамматические признаки при заполнении поисковой формы совпадают с морфологическим разбором слова, то элементу дополнительно присваивается CSS-стиль, который представляет собой оранжевую обводку, таким образом указывая пользователю на слова, которые он искал (рис. 10). Элемент выступает в качестве названия документа. В конце каждого отрывка добавляется кнопка, которой присваивается id документа, и по ее нажатию вызывается функция full_text(obj) — функция считывает id и отправляет запрос на сервер, после чего создает страницу в новом окне, демонстрируя пользователю полный текст документа.

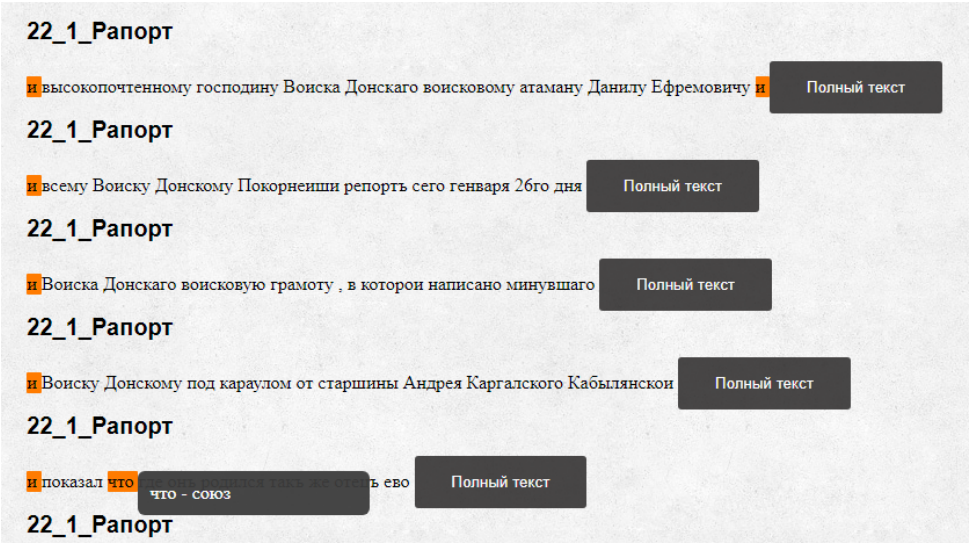

Рис. 10. Вывод ответа на поисковый запрос с признаками

Заключение

Таким образом, на данный момент нами создано программное обеспечение лингвистического корпуса документов архивного фонда. Точнее, разработка серверной части сервиса завершена полностью. Перечислим основные ресурсы на сервере:

POST /api/v1/auth/login — аутентифицировать пользователя;

POST /api/v1/auth/register — регистрация пользователя;

/api/v1/document/id — получить документ по ID;

DELETE /api/v1/document/id — удалить документ;

POST /api/v1/document/analyse — анализ текста;

POST /api/v1/document/save — сохранить документ;

POST /api/v1/document/search — поиск по документам.

Разработка клиентской части сервиса, то есть пользовательского интерфейса для доступа к ресурсам сервера, находится в финальной стадии. Приведем тут список разработанных функций, о наиболее существенных из которых мы говорили выше: analyse_collector, authorization, convert_gram, delete_doc, disp, filereader, full_text, get_replace_list, list_doc, logout, print_replace_list, retrieveFormValue, save_call, search, set_replace, text_analyse, url, user_bar.

К сожалению, на данный момент программное обеспечение корпуса не доступно для публичного тестирования, и данная статья лишь описывает его функционал. Однако мы планируем выложить разработанный веб-сервис в открытый доступ в ближайшее время.

ПРИМЕЧАНИЕ

[1] Работа выполнена при частичной финансовой поддержке гранта РФФИ № 19-012-00246.

СПИСОК ЛИТЕРАТУРЫ

1. Автоматизация процесса метаразметки архивных документов / Д. Ю. Филимонов, А. В. Светлов, О. А. Горбань, М. В. Косова, Е. М. Шептухина // Математическая физика и компьютерное моделирование. — 2020. — Т. 23, № 4. — С. 56–68. — DOI: https://doi.org/10.15688/mpcm.jvolsu.2020.4.6.

2. Балясова, Е. С. Региональные архивные документы XVIII века в аспекте корпусной лингвистики / Е. С. Балясова, Е. М. Шептухина // Коммуникативные аспекты современной лингвистики и лингводидактики : материалы междунар. науч. конф. — Волгоград : Изд-во ВолГУ, 2017. — С. 31–37.

3. Горбань, О. А. Черновой текст как основа реконструкции речемыслительной деятельности (на материале региональных документов XVIII в.) / О. А. Горбань, М. В. Косова, Е. М. Шептухина // Вестник Волгоградского государственного университета. Серия 2, Языкознание. — 2018. — Т. 17, № 4. — С. 40–54. — DOI: https://doi.org/10.15688/jvolsu2.2018.4.4.

4. Использование mystem. — Технологии Яндекса. — Электрон. текстовые дан. — Режим доступа: https://yandex.ru/dev/mystem/doc/index.html. — Загл. с экрана.

5. Комендантов, А. С. Автоматизация морфологической разметки архивных документов / А. С. Комендантов, А. Г. Матвеев, А. В. Светлов // Математическая физика и компьютерное моделирование. — 2019. — Т. 22, № 4. — С. 53–63. — DOI: https://doi.org/10.15688/mpcm.jvolsu.2019.4.4.

6. Светлов, А. В. Автоматизация процесса получения лингвистической информации: современные возможности / А. В. Светлов, А. С. Комендантов // Вестник Волгоградского государственного университета. Серия 2, Языкознание. — 2017. — Т. 16, № 2. — С. 39–46. — DOI: https://doi.org/10.15688/jvolsu2.2017.2.4.

7. Шептухина, Е. М. Войсковые грамоты середины XVIII века в аспекте категории модальности / Е. М. Шептухина, О. А. Горбань // Вестник Волгоградского государственного университета. Серия 2, Языкознание. — 2015. — № 5 (29). — С. 7–18. — DOI: http://dx.doi.org/10.15688/jvolsu2.2015.5.1.

8. Шептухина, Е. М. Этапы создания лингвистического корпуса войсковых грамот XVIII—XIX вв. архивного фонда «Михайловский станичный атаман» ГАВО / Е. М. Шептухина, О. А. Горбань // Гуманитарное образование и наука в техническом вузе : сб. докл. Всерос. науч.-практ. конф. с междунар. участием. — Ижевск : Изд-во Ижев. гос. техн. ун-та им. М.Т. Калашникова, 2017. — С. 428–431.

9. Administrative documents of the Don Cossack Host in the 18th–19th centuries: the issue of the creation of a linguistic corpus / O. Gorban, M. Kosova, E. Sheptukhina, A. Svetlov, A. Komendantov, A. Matveev, D. Filimonov // Journal: Scripta & e-Scripta. — 2021. — № 21. — P. 139–150.

10. Segalovich, I. A Fast Morphological Algorithm with Unknown Word Guessing Induced by a Dictionary for a Web Search Engine. / I. Segalovich // Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications. — Las Vegas : CSREA Press, 2003. — P. 273–280.

11. Wilde, E. REST: From Research to Practice / E. Wilde, C. Pautasso. — New York : Springer, 2011. — 528 p. — DOI: https://doi.org/10.1007/978-1-4419-8303-9.

REFERENCES

1. Filimonov D.Yu., Svetlov A.V., Gorban O.A., Kosova M.V., Sheptukhina E.M. Avtomatizatsiya protsessa metarazmetki arkhivnykh dokumentov [Automation of Archival Documents Meta Tagging]. *Matematicheskaya fizika i kompyuternoe modelirovanie* [Mathematical Physics and Computer Simulation], 2020, vol. 23, no. 4, pp. 56-68. DOI: https://doi.org/10.15688/mpcm.jvolsu.2020.4.6.

2. Balyasova E.S., Sheptukhina E.M. Regionalnye arkhivnye dokumenty XVIII veka v aspekte korpusnoy lingvistiki [The 18th Century Regional Archival Documents in the Aspect of Corpus Linguistics]. *Kommunikativnye aspekty sovremennoy lingvistiki i lingvodidaktiki: materialy mezhdunar. nauch. konf.* Volgograd, Izd-vo VolGU, 2017, pp. 31-37.

3. Gorban O.A., Kosova M.V., Sheptukhina E.M. Chernovoy tekst kak osnova rekonstruktsii rechemyslitelnoy deyatelnosti (na materiale regionalnykh dokumentov XVIII v.) [Draft Text As a Basis for Reconstructing Mental and Speech Activity (Exemplified with Regional Documents of the 18th Cen.)]. *Vestnik Volgogradskogo gosudarstvennogo universiteta. Seriya 2, Yazykoznanie* [Science Journal of Volgograd State University. Linguistics], 2018, vol. 17, no. 4, pp. 40-54. DOI: https://doi.org/10.15688/jvolsu2.2018.4.4.

4. Ispolzovanie mystem [Using MyStem]. *Tekhnologii Yandeksa*. URL: https://yandex.ru/dev/mystem/doc/index.html.

5. Komendantov A.S., Matveev A.G., Svetlov A.V. Avtomatizatsiya morfologicheskoy razmetki arkhivnykh dokumentov [Automation of Archival Documents Morphological Tagging]. *Matematicheskaya fizika i kompyuternoe modelirovanie* [Mathematical Physics and Computer Simulation], 2019, vol. 22, no. 4, pp. 53-63. DOI: https://doi.org/10.15688/mpcm.jvolsu.2019.4.4.

6. Svetlov A.V., Komendantov A.S. Avtomatizatsiya protsessa polucheniya lingvisticheskoy informatsii: sovremennye vozmozhnosti [Automation of the Process for Obtaining Linguistic Information: State-of-the-Art Capabilities]. *Vestnik Volgogradskogo gosudarstvennogo universiteta. Seriya 2, Yazykoznanie* [Science Journal of Volgograd State University. Linguistics], 2017, vol. 16, no. 2, pp. 39-46. DOI: https://doi.org/10.15688/jvolsu2.2017.2.4.

7. Sheptukhina E.M., Gorban O.A. Voyskovye gramoty serediny XVIII veka v aspekte kategorii modalnosti [Don Cossack Army Charters of the Mid 18th Century Via the Category of Modality]. *Vestnik Volgogradskogo gosudarstvennogo universiteta. Seriya 2, Yazykoznanie* [Science Journal of Volgograd State University. Linguistics], 2015, no. 5 (29), pp. 7-18. DOI: http://dx.doi.org/10.15688/jvolsu2.2015.5.1.

8. Sheptukhina E.M., Gorban O.A. Etapy sozdaniya lingvisticheskogo korpusa voyskovykh gramot XVIII—XIX vv. arkhivnogo fonda «Mikhaylovskiy stanichnyy ataman» GAVO [Stages of Creating the Linguistic Corpus of Don Cossack Army Charters of the 18^{th}–19^{th} Centuries Archive Fund "Mikhailovsky Stanichny Ataman"]. *Gumanitarnoe obrazovanie i nauka v tekhnicheskom vuze: sb. dokl. Vseros. nauch.-prakt. konf. s mezhdunar. uchastiem*. Izhevsk, Izd-vo Izhev. gos. tekhn. un-ta im. M.T. Kalashnikova Publ., 2017, pp. 428-431.

9. Gorban O., Kosova M., Sheptukhina E., Svetlov A., Komendantov A., Matveev A., Filimonov D. Administrative Documents of the Don Cossack Host in the 18^{th}–19^{th} Centuries: the Issue of the Creation of a Linguistic Corpus. *Journal: Scripta & e-Scripta*, 2021, no. 21, pp. 139-150.

10. Segalovich I. A Fast Morphological Algorithm with Unknown Word Guessing Induced by a Dictionary for a Web Search Engine. *Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications*. Las Vegas, CSREA Press, 2003, pp. 273–280.

11. Wilde E., Pautasso C. *REST: From Research to Practice*. New York, Springer, 2011. 528 p. DOI: https://doi.org/10.1007/978-1-4419-8303-9.

ON DEVELOPMENT OF WEB APPLICATION FOR CORPUS OF ARCHIVAL DOCUMENTS

Artur V. Pavlov

Student, Institute of Mathematics and IT,
Volgograd State University
art.val.pavlov@yandex.ru, matf@volsu.ru
Prosp. Universitetsky, 100, 400062 Volgograd, Russian Federation

Yuliya D. Sapich

Student, Institute of Mathematics and IT,
Volgograd State University
yds2402@yandex.ru, matf@volsu.ru
Prosp. Universitetsky, 100, 400062 Volgograd, Russian Federation

Andrey V. Svetlov

Candidate of Physical and Mathematical Sciences, Associate Professor,
Department of Mathematical Analysis and Function Theory,
Volgograd State University
a.v.svetlov@gmail.com, andrew.svetlov@volsu.ru, matf@volsu.ru
https://orcid.org/0000-0002-8764-6132
Prosp. Universitetsky, 100, 400062 Volgograd, Russian Federation

Anatoly S. Komendantov

Software Engineer,
OOO "SET"
anatoly.komendantov@gmail.com
Smolyachkova St, 4/2, 194044 Saint Petersburg, Russian Federation

Abstract. This work is a part of the project on creation the linguistic corpus of the fund "Mikhailovsky stanitsa ataman" documents. This fund contains historically valuable administrative documents of the Don Cossacks Army of the 18^{th}–19^{th} centuries, stored in the state archives of Volgograd Region. To introduce it to scientific society, a lot of preliminary work to digitize them was done by group of scientists from Volgograd State University headed by Professor O.A. Gorban. In their current form, these documents are suitable for computer processing. The only significant problem is outdated vocabulary and graphics, but it was generally solved in our previous works. At the current stage, the main task is to develop the technical and software parts of the corpus. In fact, this means the creation of an "engine" for a document corpus, that is, software for storing a database of marked-up texts, executing queries to this database, and also providing user-friendly interface that does not require special IT-skills. At the same time, in the process of working on the previous tasks, we decided to integrate the document markup tool into the general corpus software. Thus, the present work is devoted to the development of a REST service that allows you to perform automated morphological analysis of texts, save a special form of processed documents in a database, search in database by a query with morphological features of elements in the texts. The software also provides a function for manual correction of errors that occur in automated analysis of Old Slavonic texts with obsolete characters.

Key words: linguistic corpus of documents, web service, automation of morphological analysis, MyStem tool, corpus-based linguistics.