

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

КУРСОВАЯ РАБОТА
по дисциплине «Введение в нереляционные базы данных»
Тема: Инструмент сбора данных о научных публикаций

Студенты гр. 9383

Преподаватель

Хотяков Е.П.
Лапина А.А.
Лихашва А.Д.

Заславский М.М.

Санкт-Петербург

2022

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студенты: Хотяков Е.П., Лапина А.А., Лихашва А.Д.

Группа: 9383.

Тема проекта: Инструмент сбора данных о научных публикаций.

Исходные данные: Необходимо реализовать веб-инструмент для импорта данных из источников по сложным запросам (фио / организации / года / издания ...), для агрегации и генерации отчетов, экспорта. СУБД — MongoDB. Набор данных/API - Google Scholar / elibrary / ORCID / Publons API.

Содержание пояснительной записки:

Введение, Качественные требования к решению, Сценарии использования, Модель данных, Нереляционная модель данных, Аналог модели данных для SQL СУБД, Сравнение моделей, Разработанное приложение, Выводы, Приложения, Литературы.

Предполагаемый объем пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания: 10.10.2022

Дата сдачи реферата: 21.12.2022

Дата защиты реферата: 21.12.2022

Студенты

Хотяков Е.П.
Лапина А.А.
Лихашва А.Д.

Преподаватель

Заславский М.М.

АННОТАЦИЯ

В рамках данного курса предполагалось разработать какое-либо приложение в команде на одну из поставленных тем. Была выбрана тема создания инструмента сбора данных о научных публикациях с использованием MongoDB, так как мы хотели получить опыт работы с данной СУБД, а также принести пользу университету, потому что наше приложение может облегчить поиск научных статей для преподавателей. Найти исходный код и всю дополнительную информацию можно по ссылке: <https://github.com/moevm/nosql2h22-publications>.

SUMMARY

As part of this course, it was supposed to develop an application in a team on one of the topics set. The topic of creating a data collection tool for scientific publications using MongoDB was chosen, as we wanted to gain experience with this DBMS, as well as benefit the university, because our application can facilitate the search for scientific articles for teachers. You can find the source code and all additional information at the link: <https://github.com/moevm/nosql2h22-publications>.

СОДЕРЖАНИЕ

1.	Введение	6
2.	Качественные требования к решению	6
3.	Сценарии использования	6
4.	Модель данных	10
4.1	Нереляционная модель данных	10
4.2.	Аналог модели данных для SQL СУБД	14
4.3.	Сравнение моделей	18
5.	Разработанное приложение	18
6.	Выводы	19
7.	Приложения	20
8.	Литературы	23

1. Введение

В настоящее время занятие научной деятельностью становится все более популярным, но к сожалению, получение структурированной информации по какому либо критерию, например, по автору становится довольно трудоемкой задачей, так как необходимо посетить множество сайтов для получения информации, где она может быть представлена не в самом удобном виде. Поэтому выбранная тема обладает высокой актуальностью, так как с помощью разрабатываемого инструмента исследователи смогут находить научные публикации по определенным критериям.

Цель работы – разработать инструмент сбора данных о научных публикаций.

Было решено разработать веб-приложение, которое позволит хранить в электронном виде данные о научных публикациях, при этом позволяющее удобно с ними взаимодействовать.

2. Качественные требования к решению

Требуется разработать инструмент сбора данных о научных публикаций с использованием СУБД – MongoDB для удобного поиска информации о публикациях с возможностью импорта/экспорта данных.

3. Сценарии использования

Макеты UI

Полную схему макета можно увидеть здесь:

<https://www.figma.com/file/c3mDE8NkMgXGYxpv3288Cq/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA-%D0%BD>

%D0%B0%D1%83%D1%87%D0%BD%D1%8B%D1%85-%D0%BF
%D1%83%D0%B1%D0%BB%D0%B8%D0%BA
%D0%B0%D1%86%D0%B8%D0%B9?node-id=0%3A1

1. Экран Начальный экран:

Рисунок 1 — Начальный экран

2. Экран при получении данных при поиске по параметрам (в данном случае автор):

Название	Авторы	Год публикации	Издание	API	Краткое описание
Опыт создания программы автоматической генерации веб-приложений по формальным требованиям	Беляев С.А., Корытов П.В.,	2020	Cloud of Science	CyberLeninka	В статье рассматривается проблема автоматизированного создания веб-приложений по формальным требован...
Разработка программы ведения полевой геологической документации	Беляев С.А.,	2018	Cloud of Science	CyberLeninka	Полевая документация в большинстве случаев ведется в бумажном виде с использованием карандаша, в ред...
ПРИМЕНЕНИЕ ВЕРОЯТНОСТНЫХ И ВРЕМЕННЫХ АВТОМАТОВ В ПРОГРАММАХ УПРАВЛЕНИЯ МНОГОАГЕНТНЫХ СИСТЕМ	Беляев С.А.,	2020	Наукоемкие технологии в космических исследованиях Земли	CyberLeninka	Рассмотрены классические модели временного и вероятностного автомата, построенного с использованием ...
Проблемы профессиональной подготовки специалистов для эксплуатации сложных технических объектов в современных условиях	Юрий Борисович Остапченко, Сергей Алексеевич Кудряков, ВВ Романцев, СА Беляев,	2014	Известия Санкт-Петербургского государственного электротехнического университета ЛЭТИ	Google Scholar	По мере развития и усложнения техники менялась и роль человека в ее эксплуатации. Человек становился...

Рисунок 2 - Экран при получении данных при поиске по параметрам

3. Экран при отсутствии данных при поиске по параметрам (в данном случае автор):

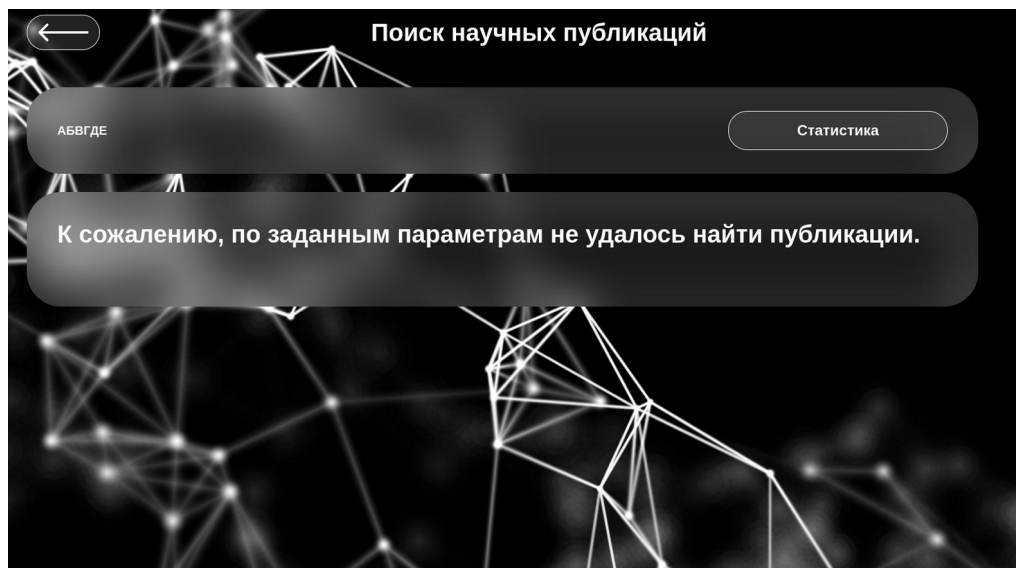


Рисунок 3 - Экран при отсутствии данных при поиске по параметрам

Описание сценариев использования

Действующее лицо: Пользователь.

Основной сценарий:

Цель: Поиск научных публикаций

1. Пользователь заходит на начальный экран, где отображены 4 поля: «ФИО», «Организация», «Год публикации», «Название публикации», где пользователь заполняет их (обязательно должно быть заполнено хотя бы 1 поле).
2. Пользователь нажимает на кнопку «Поиск» .
3. Пользователь видит полученные данные:
 - Если они нашлись, они отображаются. Пользователь может экспортировать найденные данные.

- Если при поиске не нашлось подходящих публикаций, пользователь увидит сообщение.
4. Пользователь хочет получить новые данные, возвращается на начальный экран, изменяет содержимое полей на желаемое.
 5. Переход на шаг 2.

Альтернативный сценарий 1:

Цель: Импорт БД

1. Пользователь заходит на начальный экран.
2. Нажимает кнопку «Импорт БД».

Альтернативный сценарий 2:

Цель: Экспорт БД

1. Пользователь заходит на начальный экран.
2. Нажимает кнопку «Экспорт БД».

Альтернативный сценарий 3:

Цель: Просмотр с подробным описанием публикации

1. Пользователь получает список публикаций на экране 2.
2. Нажимает на строку с нужной публикацией.
3. Открывается экран 4 с описанием публикации.

Альтернативный сценарий 4:

Цель: Получение статистики

1. Пользователь получает список публикаций на экране 2.

2. Нажимает на кнопку “Статистика”.
3. Открывается экран 5.
4. Пользователь выбирает тип издания для статистики.
5. Получение статистики.

4. Модель данных

4.1 Нереляционная модели данных

Графическое представление



Рисунок 4 — Графическое представление нереляционной модели

Описание назначений коллекций, типов данных и сущностей

Всего в проекте необходимо 3 сущности - Author - автор, Organization - организация и Publication - публикация. (В MongoDB латинские буквы и символы кодируются 1 байтом, а кириллические буквы – 2 байтами).

Ниже представлено описание коллекций: Database

author: // коллекция, содержащая информацию об авторе
(итого 84 байта)

```
{
  id_author: ObjectId,           // идентификатор автора (12 байт)
  FIO: String,                   // ФИО автора (до 30 символов или 60 байт)
  id_organization: ObjectId,     // идентификатор организации (12 байт)
  id_publications: [ObjectId, ..., ObjectId] // публикации (от 12 байт)
}
```

organization: // коллекция, содержащая информацию об
организации (итого 76 байта)

```
{
  id_organization: ObjectId,     // идентификатор организации (12 байт)
  name_organization: String,     // название организации (до 20 символов или 40
байт)
  id_authors: [ObjectId, ..., ObjectId] // авторы (от 12 байт)
  id_publications: [ObjectId, ..., ObjectId] // публикации (от 12 байт)
}
```

publication: // коллекция, содержащая информацию о публикации
(итого 445 байта)

```
{
  id_publication: ObjectId,      // идентификатор публикации (12 байт)
  name_publication: String,      // название публикации (до 20 символов или 40
байт)
  year_publication: 64-Bit Integer, // год публикации (4 байта)
  link_publication: String,      // ссылка на публикацию (до 50 символов или 50
байт)
  API: String,                   // API (до 15 символов или 15 байт)
  edition: String,               // издание (до 30 символов или 60 байт)
  description: String            // краткое описание публикации (до 130 символов
или 260 байт)
  id_organization: ObjectId,     // идентификатор организации (12 байт)
  id_authors: [ObjectId, ..., ObjectId] // авторы (от 12 байт)
}
```

Пример данных

Автор:

```
{
```

```

    id_author: ObjectId("1111a1e112df7f8644c2cea2"),
    FIO: "Лапина Анастасия Андреевна",
    id_organization: ObjectId("2222a1e112df7f8644c2cea2"),
    id_publications: [ObjectId("3333a1e112df7f8644c2cea2"),
ObjectId("aaaaa1e112df7f8644c2cea2")]
}

```

Организация:

```

{
    id_organization: ObjectId("2222a1e112df7f8644c2cea2"),
    name_organization: "ЛЭТИ",
    id_authors: [ObjectId("1111a1e112df7f8644c2cea2")]
    id_publications: [ObjectId("3333a1e112df7f8644c2cea2"),
ObjectId("aaaaa1e112df7f8644c2cea2")]
}
}

```

Публикация:

```

{
    id_publication: ObjectId("3333a1e112df7f8644c2cea2"),
    name_publication: "Имя публикации",
    year_publication: 2022,
    link_publication: "https://etu.ru",
    API: "Google Scholar",
    edition: "Научное издание номер 1",
    description: "В публикации ...",
    id_organization: ObjectId("2222a1e112df7f8644c2cea2"),
    id_authors: [ObjectId("1111a1e112df7f8644c2cea2")]
}

```

Оценка удельного объема информации, хранимой в модели (сколько потребуется памяти, чтобы сохранить объекты, как объем зависит от количества объектов)

Будем считать, что у нас X_a - авторов, X_o - организаций и X_p - публикаций. Следовательно, объем информации можно найти так: $84 * X_a + 76 * X_o + 445 * X_p$, X_o и X_p можно считать словарями, и значит, их размеры берем за константу, тогда получаем оценку удельного объема информации: $605 * X_a$

Избыточность модели

Избыточными полями в нашей бд являются: id_author, id_organization, id_publication. Тогда суммарный объем избыточных данных равен 36, а чистых 569*Ха Следовательно, избыточность модели равна: $(605/569)*\text{Ха} \sim 1,06$

Направление роста модели

Рассматривая модель данных и полученные результаты, можно сделать вывод о том, что модель растёт с линейной скоростью.

Запросы к модели, с помощью которых реализуются сценарии использования

•Поиск автора

```
let FIO_ = "Иванов Иван Иванович";
```

```
db.author.find({FIO: FIO_});
```

•Поиск организации

```
let organization = "ЛЭТИ";
```

```
db.organization.find({name_organization: organization});
```

•Поиск публикации

```
let publication = "Какое-то название";
```

```
db.publication.find({name_publication: publication});
```

4.2 Аналог модели данных для SQL СУБД - характеризуется аналогично нереляционной

Графическое представление

Разработана аналогичная схема реляционной базы данных:

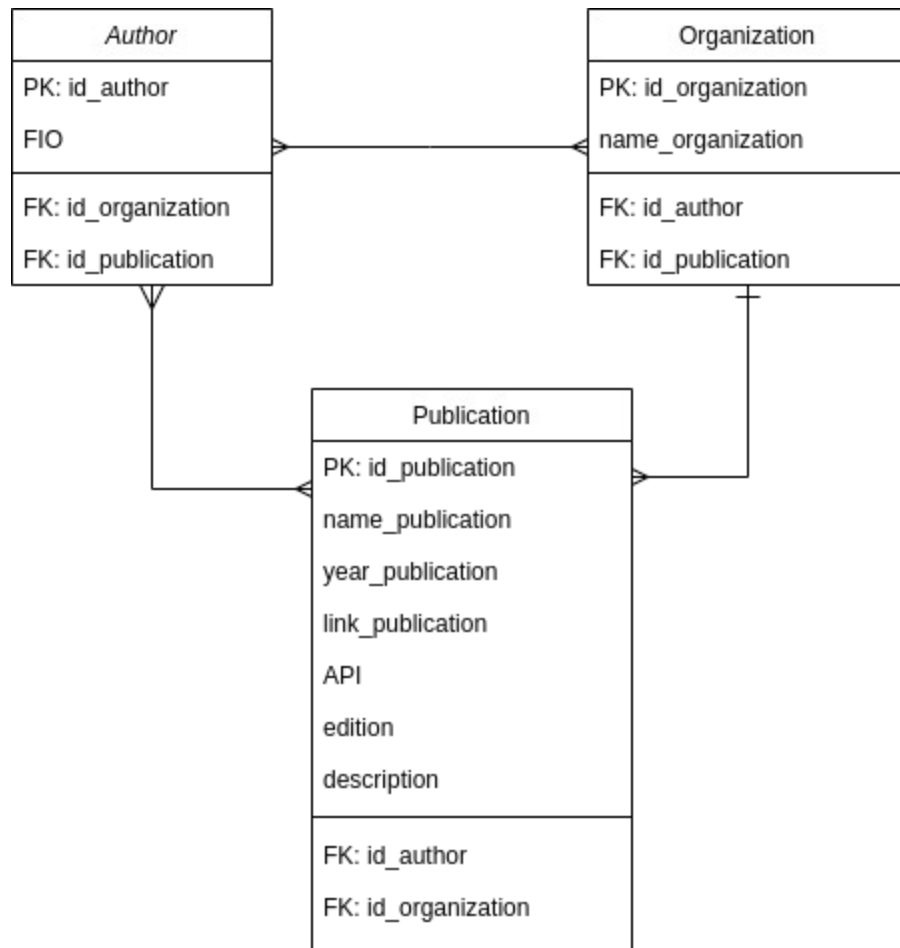


Рисунок 5 — Графическое представление реляционной модели

Описание назначений коллекций, типов данных и сущностей

Всего в проекте необходимо 3 сущности - Author - автор, Organization - организация и Publication - публикация.

Ниже представлено описание коллекций: Будем использовать Unicode, то есть символы кодируются 2 байтами Database:

```

author:                // коллекция, содержащая информацию об авторе (итого 96
байта)
{
    id_author: ObjectId,    // идентификатор автора (12 байт)
    FIO: String,           // ФИО автора (до 30 символов или 60 байт)
    id_organization: ObjectId, // идентификатор организации (12 байт)
}
  
```

```

        id_publication: ObjectId      // идентификатор публикации (12 байт)
    }

organization:                        // коллекция, содержащая информацию об организации
(итого 76 байта)
{
    id_organization: ObjectId,        // идентификатор организации (12 байт)
    name_organization: String,        // название организации (до 20 символов или 40
байт)
    id_author: ObjectId,              // идентификатор автора (12 байт)
    id_publication: ObjectId          // идентификатор публикации (12 байт)
}

publication:                         // коллекция, содержащая информацию о публикации
(итого 530 байта)
{
    id_publication: ObjectId,          // идентификатор публикации (12 байт)
    name_publication: String,          // название публикации (до 20 символов или 40
байт)
    year_publication: 64-Bit Integer   // год публикации (4 байта)
    link_publication: String           // ссылка на публикацию (до 50 символов или 100
байт)
    API: String                       // API (до 15 символов или 30 байт)
    edition: String                   // издание (до 30 символов или 60 байт)
    description: String               // краткое описание публикации (до 130 символов
или 260 байт)
    id_author: ObjectId,              // идентификатор автора (12 байт)
    id_organization: ObjectId          // идентификатор организации (12 байт)
}

```

Пример данных

Автор:

```

{
    id_author: ObjectId("1111a1e112df7f8644c2cea2"),
    FIO: "Лапина Анастасия Андреевна",
    id_organization: ObjectId("2222a1e112df7f8644c2cea2"),
    id_publication: ObjectId("3333a1e112df7f8644c2cea2")
}

```

Организация:

```

{
    id_organization: ObjectId("2222a1e112df7f8644c2cea2"),

```

```

    name_organization: "ЛЭТИ",
    id_author: ObjectId("1111a1e112df7f8644c2cea2"),
    id_publication: ObjectId("3333a1e112df7f8644c2cea2")
}

```

Публикация:

```

{
    id_publication: ObjectId("3333a1e112df7f8644c2cea2"),
    name_publication: "Имя публикации",
    year_publication: 2022,
    link_publication: "https://etu.ru",
    API: "Google Scholar",
    edition: "Научное издание номер 1",
    description: "В публикации ...",
    id_author: ObjectId("1111a1e112df7f8644c2cea2"),
    id_organization: ObjectId("2222a1e112df7f8644c2cea2")
}

```

Оценка удельного объема информации, хранимой в модели (сколько потребуется памяти, чтобы сохранить объекты, как объем зависит от количества объектов)

Будем считать, что у нас X_a - авторов, X_o - организаций и X_p - публикаций. Следовательно, объем информации можно найти так: $96 * X_a + 76 * X_o + 530 * X_p$, X_o и X_p можно считать словарями, и значит, их размеры берем за константу, тогда получаем оценку удельного объема информации: $702 * X_a$

Избыточность модели

Избыточными полями в нашей бд являются: `id_author`, `id_organization`, `id_publication`. Тогда суммарный объем избыточных данных равен 36, а чистых $666 * X_a$ Следовательно, избыточность модели равна: $(702/666) * X_a \sim 1,05$

Направление роста модели

Рассматривая модель данных и полученные результаты, можно сделать вывод о том, что модель растёт с линейной скоростью.

Запросы к модели, с помощью которых реализуются сценарии использования

•Поиск по автору

```
SELECT publication.publication_name, author.FIO, publication.year_publication,  
publication.edition, publication.API, publication.description  
FROM publication  
JOIN publication on publication.id_publication = author.id_publication  
WHERE author.FIO = "Иванов Иван Иванович";
```

•Поиск по организации

```
SELECT publication.publication_name, author.FIO, publication.year_publication,  
publication.edition, publication.API, publication.description  
FROM publication  
JOIN publication on publication.id_publication = author.id_publication  
JOIN organization on organization.id_organization = publication.id_organization  
WHERE organization.name_organization = "ЛЭТИ";
```

•Поиск по публикациям

```
SELECT publication.publication_name, author.FIO, publication.year_publication,  
publication.edition, publication.API, publication.description  
FROM publication  
JOIN publication on publication.id_publication = author.id_publication  
WHERE publication.name_publication = "Публикация номер 1";
```


4.3 Сравнение моделей

Сравнение реляционной и нереляционной моделей представлено в следующей таблице:

	NoSQL	SQL
Удельный объем информации	$605 * X_a$	$702 * X_a$
Избыточность	1.06	1.05
Количество коллекций	3	3

Анализируя данные, можно сделать вывод, что нереляционная база данных лучше подходит для данной задачи, нежели реляционная, так как удельный объем информации у нереляционной меньше, при этом показатель избыточности у нее незначительно больше, поэтому им можно пренебречь.

5. Разработанное приложение

Краткое описание

Инструмент сбора данных о научных публикациях реализован на языке программирования - JavaScript. С помощью данного приложения можно совершать поиск научных статей по различным критериям таким как: ФИО, название публикации, год публикации, организация, тип издания, название издания, количество цитат и index. В случае, если данные по заданному запросу не найдены, то об этом сообщается пользователю. В качестве СУБД использована MongoDB.

Схема экранов приложения и/или схема интерфейса командной строки

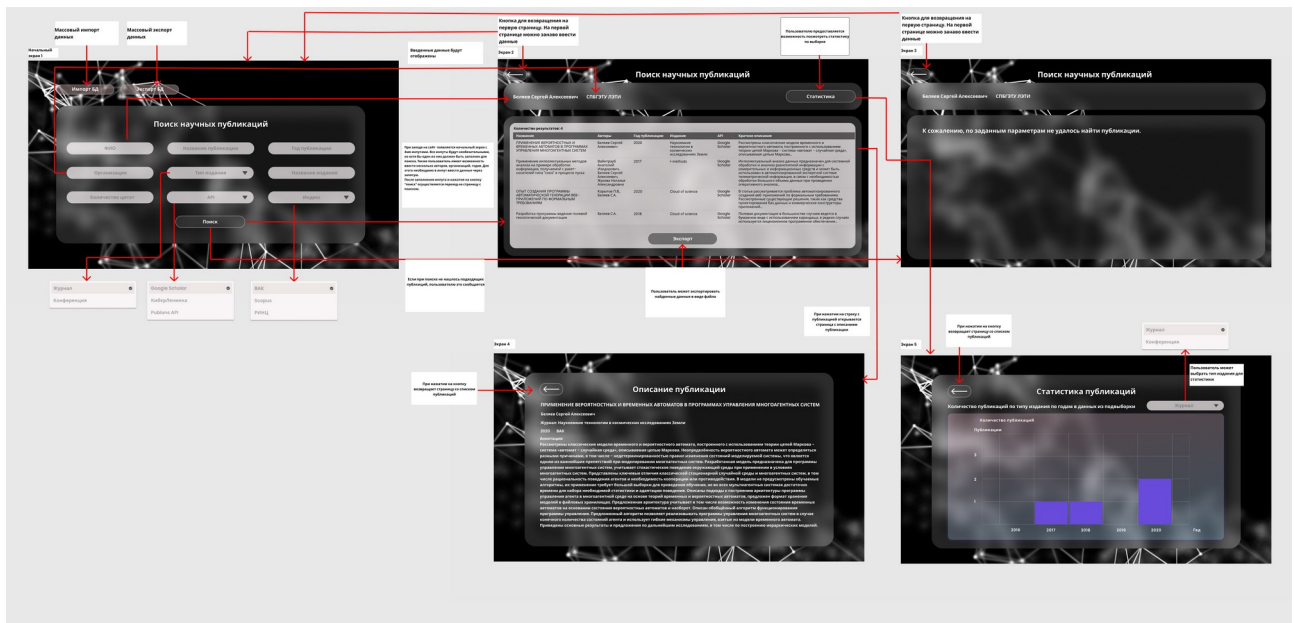


Рисунок 6 — Схема экранов приложения

Посмотреть схему в более удобном виде можно здесь:
<https://www.figma.com/file/c3mDE8NkMgXGYxpv3288Cq/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA-%D0%BD%D0%B0%D1%83%D1%87%D0%BD%D1%8B%D1%85-%D0%BF%D1%83%D0%B1%D0%BB%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B9?node-id=0%3A1>

Использованные технологии

БД: MongoDB,

Back-end, Front-end: JavaScript, HTML, CSS.

Ссылки на Приложение

Ссылка на github: <https://github.com/moevm/nosql2h22-publications>

6. Выводы

Достигнутые результаты

В ходе работы был разработан инструмент для поиска научных публикаций, который позволяет пользователям быстро и удобно находить

информацию о научных публикациях по критериям: таким как: ФИО, название публикации, год публикации, организация, тип издания, название издания, количество цитат и index. Также создан докер-контейнер, который облегчает развертывание приложения.

Недостатки и пути для улучшения полученного решения

Пока приложение находится на стадии развития и не реализован раздел со статистикой.

Будущее развитие решения

В будущем планируем реализовать раздел с описанием статьи и сделать статистику.

7. Приложения

Документация по сборке и развертыванию приложения

1. Склонировать проект из репозитория (указан в ссылках на приложение)
2. Запустить докер-контейнер с помощью `docker-compose build --no-cache`
3. Открыть приложение в браузере по адресу `http://localhost:3000`

Инструкция для пользователя

Для получения информации о публикации:

- 1) Зайти на главный экран
- 2) Ввести необходимые параметры для поиска
- 3) Увидеть результат

Для импорта:

- 1) Зайти на главный экран
- 2) Нажать кнопку «Импорт БД»

Для экспорта:

- 1) Зайти на главный экран
- 2) Нажать кнопку «Экспорт БД»

Снимки экрана приложения

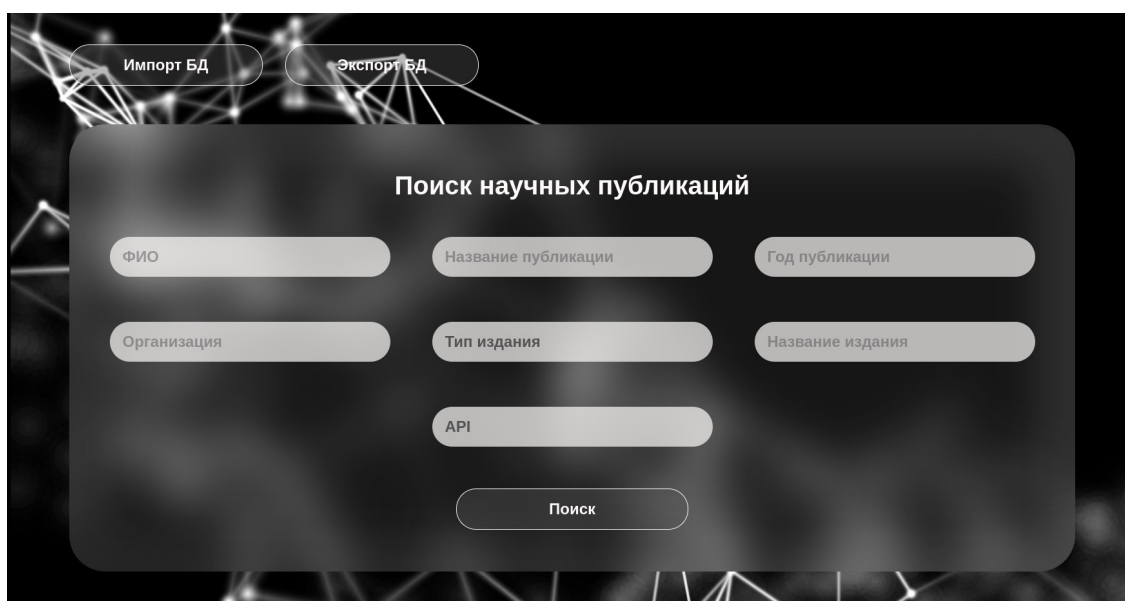


Рисунок 7 — Главный экран

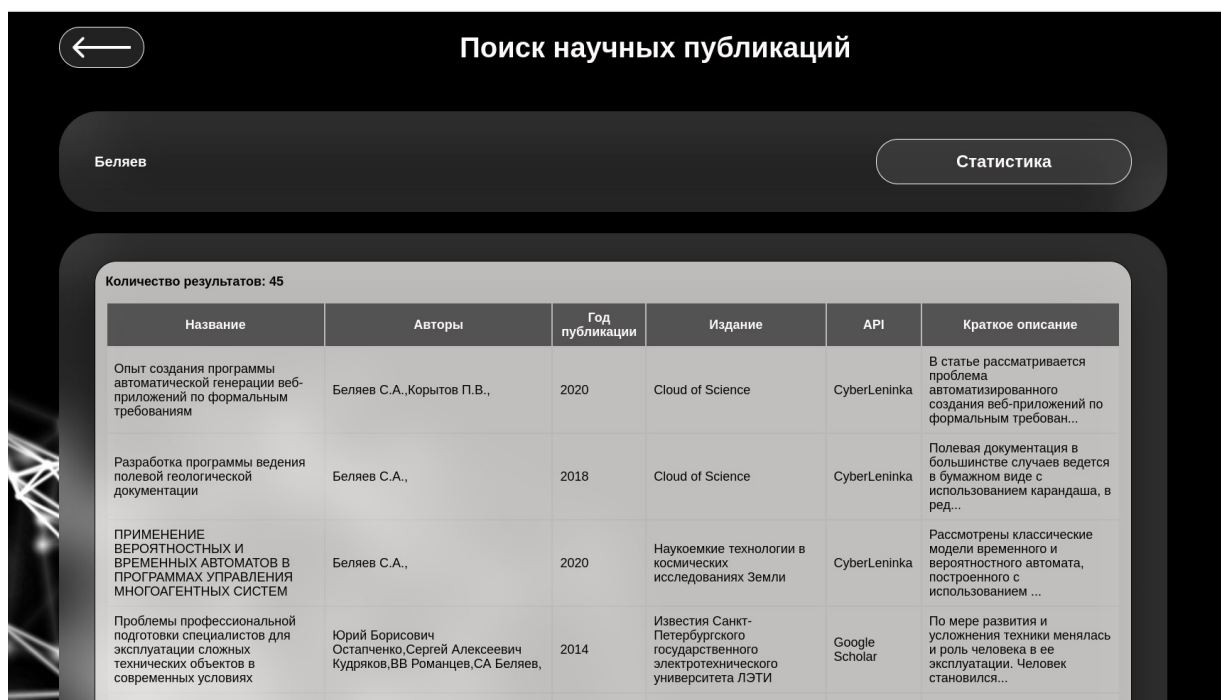


Рисунок 8 - Экран при получении данных при поиске по параметрам

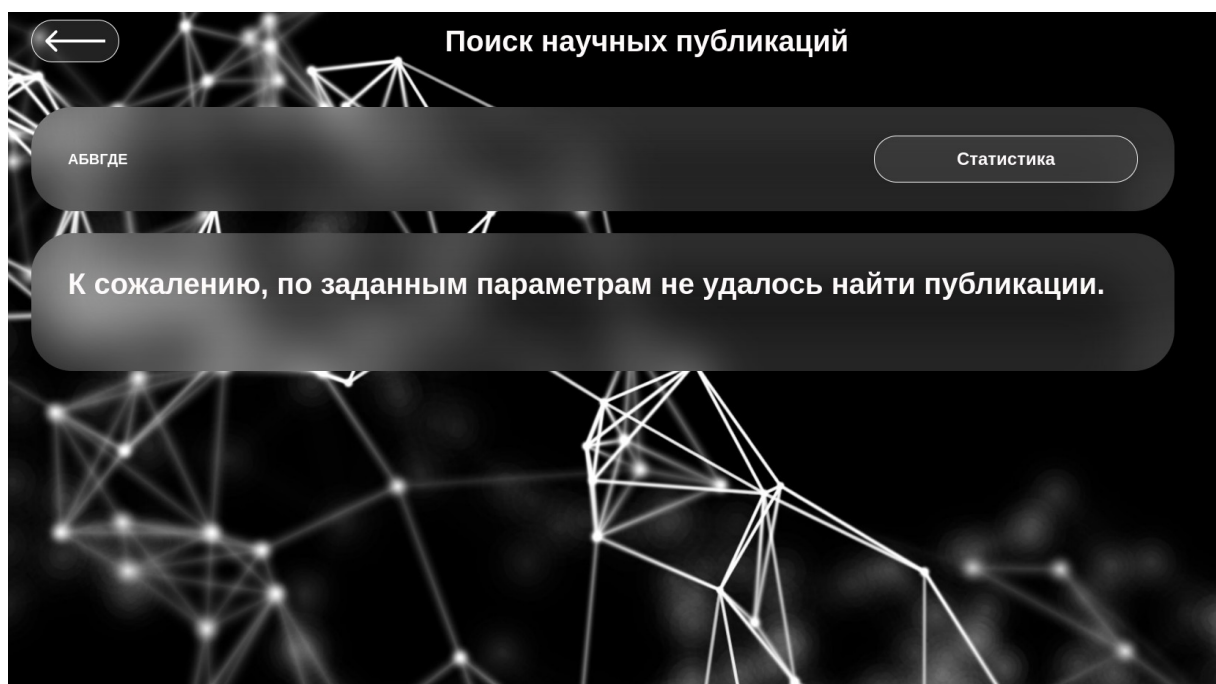


Рисунок 9 - Экран при отсутствии данных при поиске по параметрам

Поиск по критериям: ФИО, год, организация:

Импорт БД Экспорт БД

Поиск научных публикаций

Беляев Название публикации 2018

ЛЭТИ Тип издания Название издания

Количество цитат API index

Поиск

Рисунок 10 — Заполняем данные

Поиск научных публикаций

Беляев 2018 ЛЭТИ Статистика

Количество результатов: 1

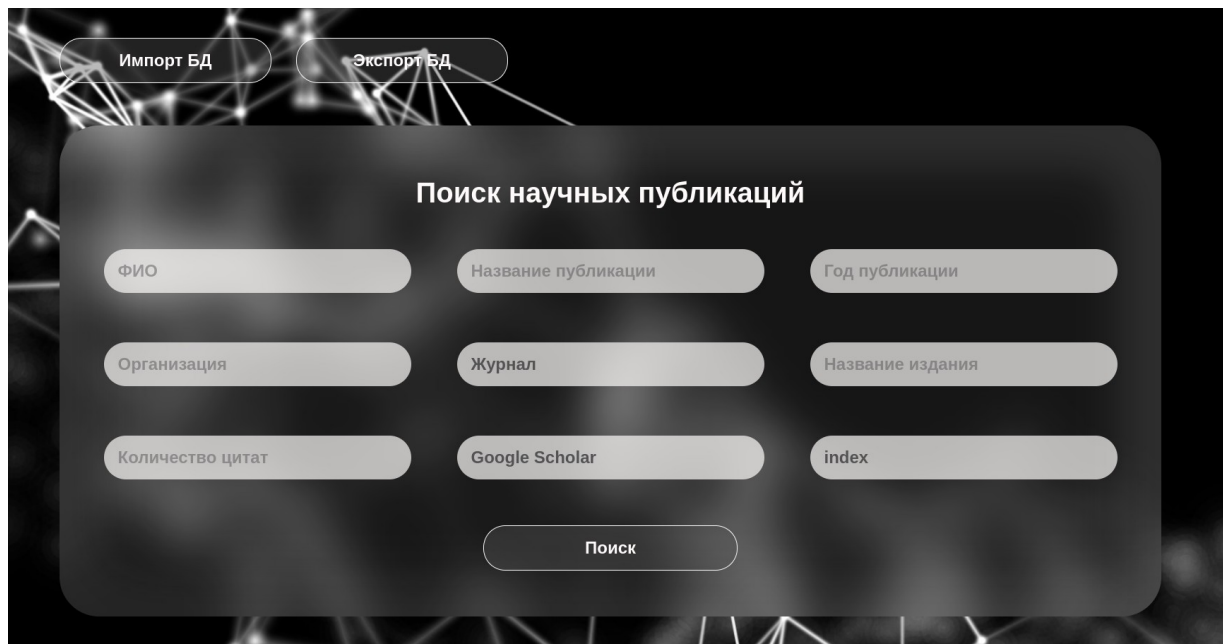
Название	Авторы	Год публикации	Издание	API	Краткое описание
Разработка программы ведения полевой геологической документации	Беляев С.А.,	2018	Cloud of Science	CyberLeninka	Полевая документация в большинстве случаев ведется в бумажном виде с использованием карандаша, в ред...

0

Экспорт

Рисунок 11 — Результат поиска

Поиск журналов в Google Scholar



Импорт БД Экспорт БД

Поиск научных публикаций

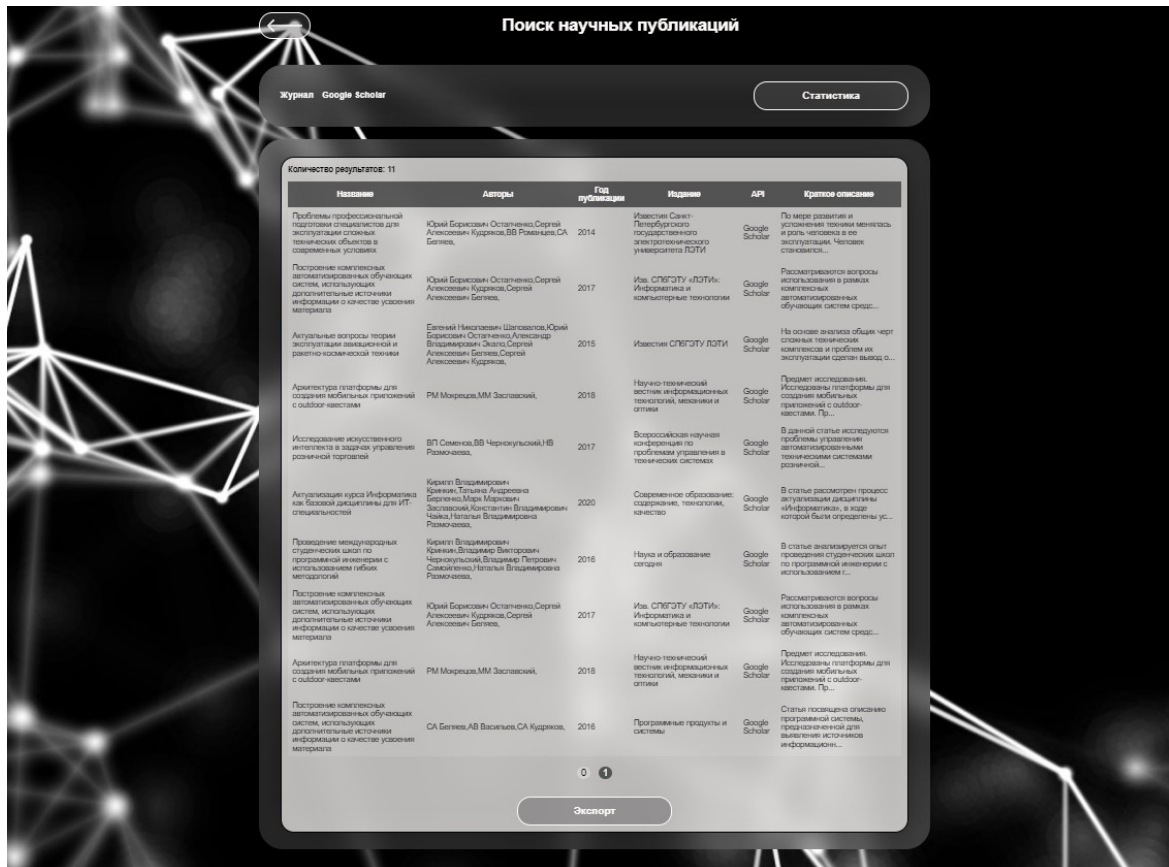
ФИО Название публикации Год публикации

Организация Журнал Название издания

Количество цитат Google Scholar index

Поиск

Рисунок 12 — Заполняем данные



Журнал Google Scholar Статистика

Количество результатов: 11

Название	Авторы	Год публикации	Издание	АИ	Краткое описание
Проблемы профессиональной подготовки специалистов для эксплуатации станций технического обслуживания в современных условиях	Юрий Борисович Остапенко, Сергей Александрович Курдюков, ВВ Романца, СА Бегиев	2014	Известия Санкт-Петербургского государственного электротехнического университета ЛЭТИ	Google Scholar	По мере развития и усложнения техники меняется и роль человека в ее эксплуатации. Человек становится...
Построение комплексных автоматизированных обучающих систем, использующих дополнительные источники информации о качестве усвоения материала	Юрий Борисович Остапенко, Сергей Александрович Курдюков, Сергей Александрович Бегиев	2017	Изв. СПбГЭТУ «ЛЭТИ». Информатика и компьютерные технологии	Google Scholar	Рассматриваются вопросы использования в рамках комплексной автоматизированной обучающей системы сред...
Актуальные вопросы теории эксплуатации авиационной и ракетно-космической техники	Евгений Николаевич Шаповалов, Юрий Борисович Остапенко, Александр Владимирович Окало, Сергей Александрович Бегиев, Сергей Александрович Курдюков	2015	Известия СПбГЭТУ ЛЭТИ	Google Scholar	На основе анализа общих черт станций технического обслуживания и проблем их эксплуатации сделан вывод о...
Архитектура платформы для создания мобильных приложений с outdoor-частями	РМ Морозов, ММ Заславский	2018	Научно-технический вестник информационных технологий, механики и оптики	Google Scholar	Предмет исследования. Исследованы платформы для создания мобильных приложений с outdoor-частями. Пр...
Исследование искусственного интеллекта в задачах управления релейной топологией	ВЛ Семенин, ВВ Чернуцкий, НВ Романцова	2017	Воронежская научная конференция по проблемам управления в технических системах	Google Scholar	В данной статье исследуются проблемы управления автоматизированными техническими системами различной...
Анализ курса Информатика базовой дисциплины для ИТ-специалистов	Кирити Владимирович Крикин, Татьяна Андреевна Бегиева, Марк Маркович Заславский, Константин Владимирович Чалка, Наталья Владимировна Романцова	2020	Современное образование: содержание, технологии, качество	Google Scholar	В статье рассмотрен процесс актуализации дисциплины «Информатика», в ходе которой были определены ус...
Прохождение международных студенческих школ по программной инженерии с использованием гибких методологий	Кирити Владимирович Крикин, Владимир Викторович Чернуцкий, Владимир Петрович Самойленко, Наталья Владимировна Романцова	2016	Наука и образование сегодня	Google Scholar	В статье анализируется опыт проведения студенческих школ по программной инженерии с использованием...
Построение комплексных автоматизированных обучающих систем, использующих дополнительные источники информации о качестве усвоения материала	Юрий Борисович Остапенко, Сергей Александрович Курдюков, Сергей Александрович Бегиев	2017	Изв. СПбГЭТУ «ЛЭТИ». Информатика и компьютерные технологии	Google Scholar	Рассматриваются вопросы использования в рамках комплексной автоматизированной обучающей системы сред...
Архитектура платформы для создания мобильных приложений с outdoor-частями	РМ Морозов, ММ Заславский	2018	Научно-технический вестник информационных технологий, механики и оптики	Google Scholar	Предмет исследования. Исследованы платформы для создания мобильных приложений с outdoor-частями. Пр...
Построение комплексных автоматизированных обучающих систем, использующих дополнительные источники информации о качестве усвоения материала	СА Бегиев, АВ Васильев, СА Курдюков	2016	Прогрессивные продукты и системы	Google Scholar	Статья посвящена описанию программной системы, предназначенной для выделения источников информации...

0 1

Экспорт

Рисунок 13 — Результат поиска

Импорт БД

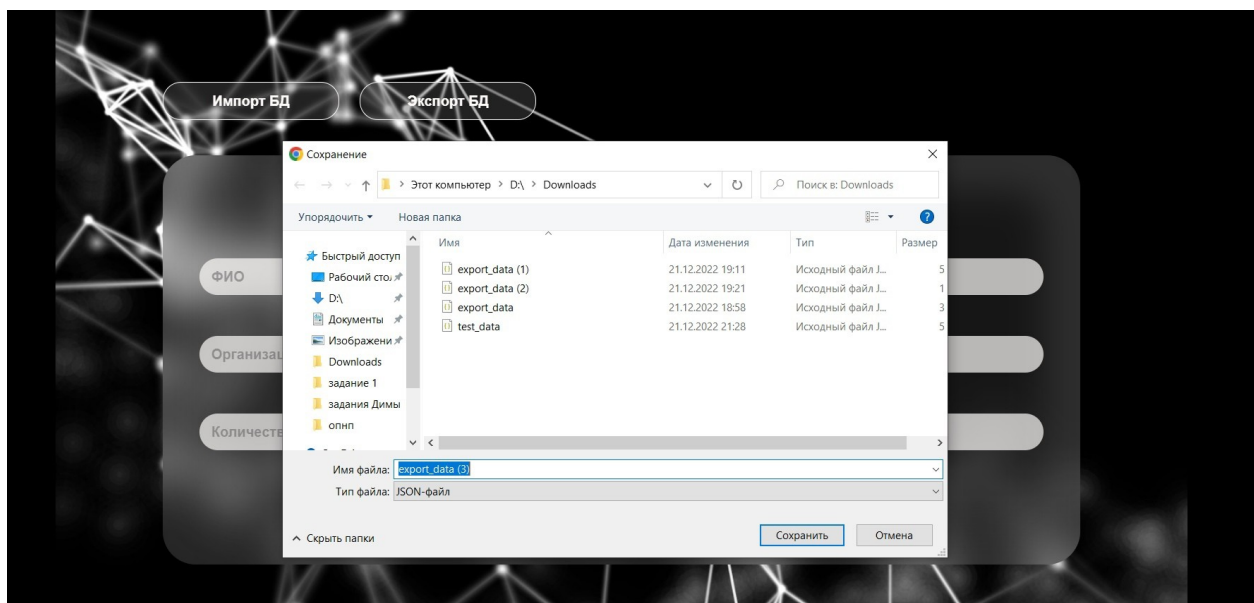


Рисунок 14 — Предложение выбрать названия файла при нажатии на «Импорт БД»

Экспорт БД

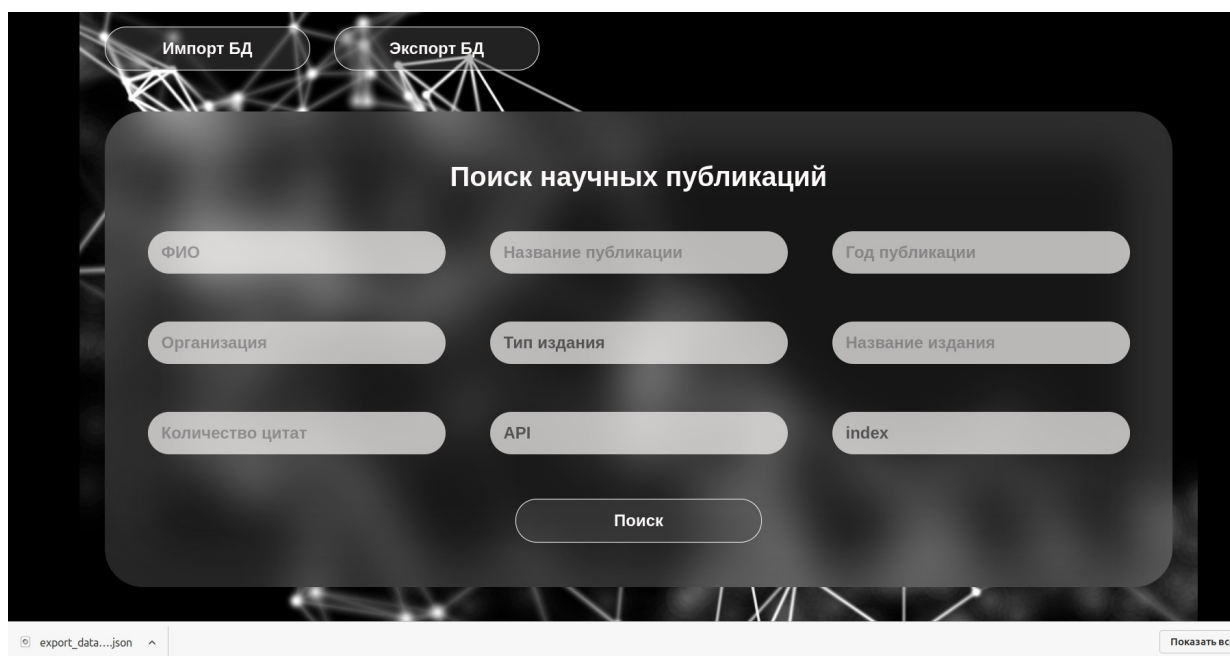


Рисунок 15 — Скачивается json при нажатии «Экспорт БД»



```
1 [{"_id":"63a330f50e079def52cba49b","id_publication":"1","FIO":["Беляев  
С.А.", "Корытов П.В."], "name_publication": "Опыт создания программы автоматической  
генерации веб-приложений по формальным требованиям", "year_publication": 2020, "name_organization": "Санкт-Петербургский государственный электротехнический  
университет «ЛЭТИ»", "type_edition": "Журнал", "name_edition": "Cloud of  
Science", "number_quotes": 10, "API": "CyberLeninka", "index": "BAK", "description": "В  
статье рассматривается проблема автоматизированного создания веб-приложений по  
формальным требованиям. Рассмотрены существующие решения, такие как средства  
проектирования баз данных и коммерческие конструкторы приложений, либо не  
предоставляют достаточную функциональность для создания работающего приложения,  
либо не работают с формальными моделями и малорасширяемы. Разработанное авторами  
решение предоставляет возможность создания клиент-серверного веб-приложения по  
модели «сущность-связь». Статья описывает алгоритм генерации кода по этой модели и  
некоторые детали создания системы, независимой от предметной области. Решение  
позволяет автоматизировать создание простых учетных систем и упростить создание  
более сложных."}, {"_id":"63a330f50e079def52cba49c","id_publication":"2","FIO":  
["Беляев С.А."], "name_publication": "Разработка программы ведения полевой  
геологической документации", "year_publication": 2018, "name_organization": "Санкт-  
Петербургский государственный электротехнический университет  
«ЛЭТИ»", "type_edition": "Журнал", "name_edition": "Cloud of Science", "number_quotes": 5, "API": "CyberLeninka", "index": "BAK", "description": "Полевая документация в  
большинстве случаев ведется в бумажном виде с использованием карандаша, в редких  
случаях используется лицензионное программное обеспечение. Это связано с высокой  
стоимостью оборудования и основными финансовыми вложениями именно в оборудование и  
технологии бурения, а не средства документирования. В работе предлагается описание  
построения универсальной программы ведения полевой геологической документации,  
которая решает существенную часть задач своих конкурентов в части ведения полевого  
журнала. В работе описаны основные требования к программе ведения геологической  
документации, сформулированы ее ограничения, предложены решения по каждой из  
ключевых задач разработки, приведены принципиальные фрагменты программного кода на  
Java. Описаны подходы к формированию текстового описания для каждого интервала с  
использованием скриптового языка JavaScript, в том числе с использованием данных  
по подынтервалам."}, {"_id":"63a330f50e079def52cba49d","id_publication":"3","FIO":  
["Блеес Э.И.", "Заславский М.М."], "name_publication": "Критерии соответствия текста  
научному стилю", "year_publication": 2019, "name_organization": "Санкт-Петербургский  
государственный электротехнический университет  
«ЛЭТИ»", "type_edition": "Журнал", "name_edition": "Научно-технический вестник  
информационных технологий, механики и оптики", "number_quotes": 25, "API": "CyberLeninka", "index": "BAK", "description": "Приведены результаты  
экспериментального исследования критериев соответствия текста научному стилю.  
Исследованы показатель повторений в текстовом документе ключевых слов и фраз,  
процентное соотношение стопслов и общего числа слов в тексте, отклонение графика  
частоты слов в тексте от идеального графика по Ципфу. Исследование проведено с  
применением сценария, проверяющего текст по нескольким критериям. В результате  
экспериментального исследования на выборке из 2500 статей, опубликованных в  
источниках ВАК/РИНЦ, получены распределения значений критериев, которые проверены  
на нормальность по нескольким критериям, а также на корреляцию между собой. В
```

Рисунок 16 — Содержание экспортированного

8. Литература

- 1) Документация MongoDB: <https://docs.mongodb.com/manual/>
- 2) Документация к Docker: <https://www.docker.com/>