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

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине «Введение в нереляционные базы данных»

Тема: Каталог литературных цитат

Студент гр.1304 _____ Басыров В.А.

Студент гр.1304 _____ Маркуш А.Е.

Студентка гр.1304 _____ Ярусова Т.В.

Преподаватель _____ Заславский М.М.

Санкт-Петербург

2024

ЗАДАНИЕ

НА ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

Студент Басыров В.А.

Студент Маркуш А.Е.

Студентка Ярусова Т.В.

Группа 1304

Тема работы: каталог литературных цитат

Исходные данные:

Необходимо реализовать веб-приложение для каталога литературных цитат

Содержание пояснительной записки:

«Содержание», «Введение», «Сценарии использования», «Модель данных», «Разработанное приложение», «Заключение», «Список использованных источников»

Предполагаемый объем пояснительной записки:

Не менее 20 страниц.

Дата выдачи задания: 05.09.2024

Дата сдачи реферата: 9.12.2024

Дата защиты реферата: 9.12.2024

Студент	_____	Басыров В.А.
Студент	_____	Маркуш А.Е.
Студент	_____	Ярусова Т.В.
Преподаватель	_____	Заславский М.М.

АННОТАЦИЯ

В рамках ИДЗ разработано веб приложение “Каталог литературных цитат”, в котором представлены цитаты из произведений различных авторов. Приложение включает возможность просмотра цитат, авторов, произведений и героев, а также их фильтрация. Также есть возможность импортировать и экспортировать цитаты, просматривать статистику по цитатам, авторам, произведениям и героям.

Приложение реализовано с помощью технологий React, TypeScript, NodeJs и MongoDb.

Исходный код представлен по ссылке: https://github.com/moevm/nosql2h24-citations

SUMMARY

Within the framework of the IDZ, a web application “Catalog of literary quotations” has been developed, which contains quotations from the works of various authors. The application includes the ability to view quotes, authors, works and characters, as well as their filtering. It is also possible to import and export quotes, view statistics on quotes, authors, works and heroes.

The application is implemented using React, TypeScript, NodeJS and MongoDB technologies.

The source code is provided at the link: https://github.com/moevm/nosql2h24-citations

СОДЕРЖАНИЕ

1.	Введение	6
1.1.	Актуальность решаемой проблемы	6
1.2.	Постановка задачи	6
1.3.	Предлагаемое решение	6
1.4.	Качественные требования к решению	6
2.	Сценарии использования	7
2.1.	Макет UI	7
2.2.	Сценарии использования для импорта данных	15
2.3.	Сценарии использования для предоставления данных	16
2.4.	Сценарии использования для анализа данных	25
2.5.	Сценарии использования для экспорта данных	26
3.	Модель данных	29
3.1.	Нереляционная модель данных	29
3.2.	Аналог модели данных для SQL СУБД	32
3.3.	Сравнение моделей	37
4.	Разработанное приложение	39
5.	Выводы	41
6.	Приложения	43
7.	Литература	44

1.ВВЕДЕНИЕ

1.1 Актуальность проблемы

Для всестороннего развития личности важно предоставить возможность человеку читать классическую литературу. Поэтому необходимо популяризировать классическую литературу и завлекать читательскую аудиторию. Один из таких способов - создать каталог литературных цитат, в котором пользователь будет искать яркие цитаты и узнавать о новых интересных книгах и повышать свой интерес к литературе .

1.2 Постановка задачи

Задача проекта заключается в:

1) Сделать каталог цитат.
2) Сделать карточки цитат с подробной информацией об авторе, герое, произведении и самой цитате.
3) Предоставить возможность фильтровать цитаты по годам, авторам, произведениям и героям.
4) Сделать страницу статистики для цитат, авторов, произведений и героев.
5) Добавить возможность импорта и экспорта цитат.

1.3. Предлагаемое решение.

Приложение реализовано с помощью технологий React, TypeScript, NodeJs и MongoDb.

1.4. Качественные требования к решению.

Приложение должно быть простым в использовании, масштабируемым и производительным.

2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

2.1. Макет UI

Ниже представлены страницы макета приложения (рис. 1 - 9).

Рисунок 1 - Главная страница приложения.

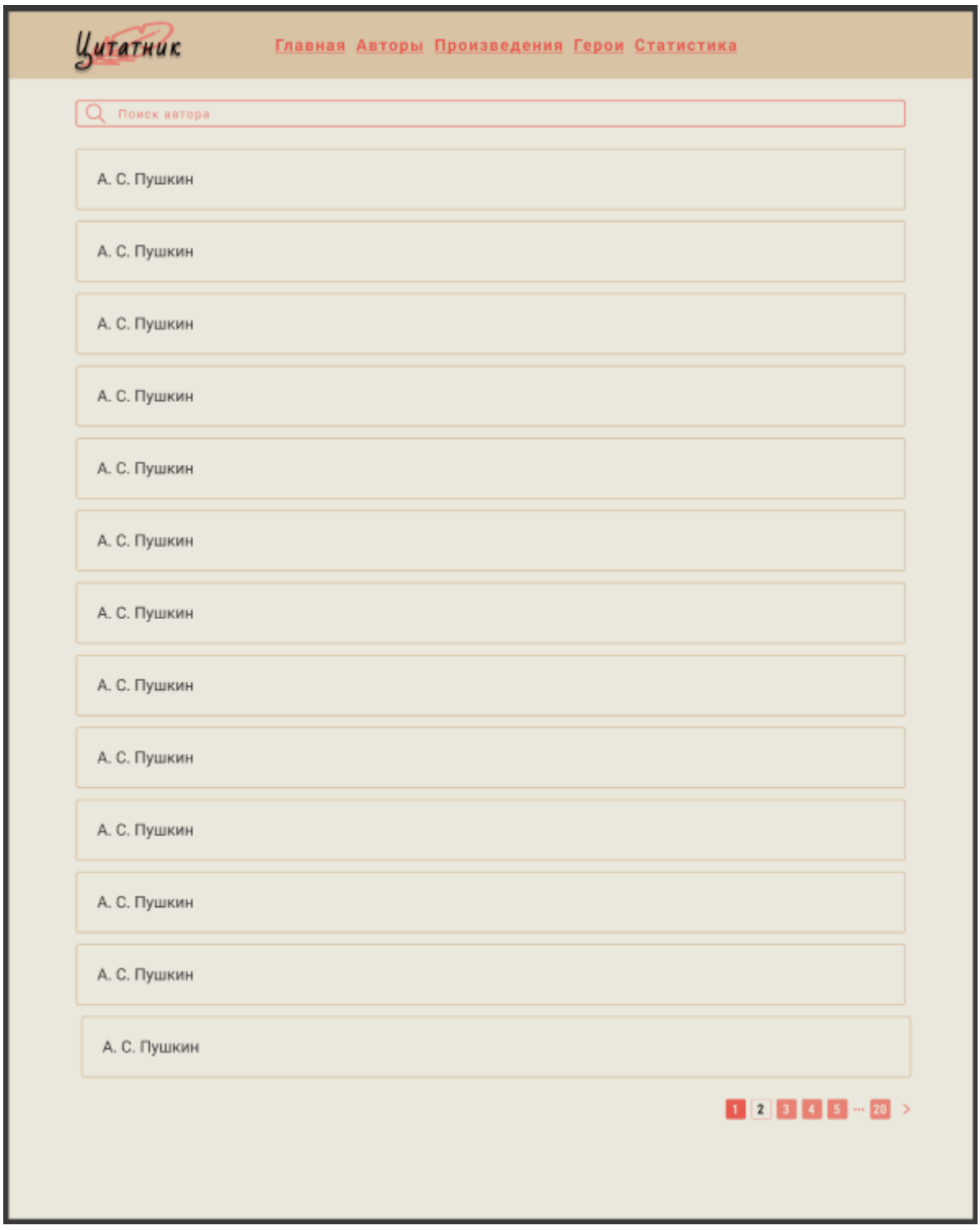


Рисунок 2 - Страница авторов.

Рисунок 3 - Страница героев.

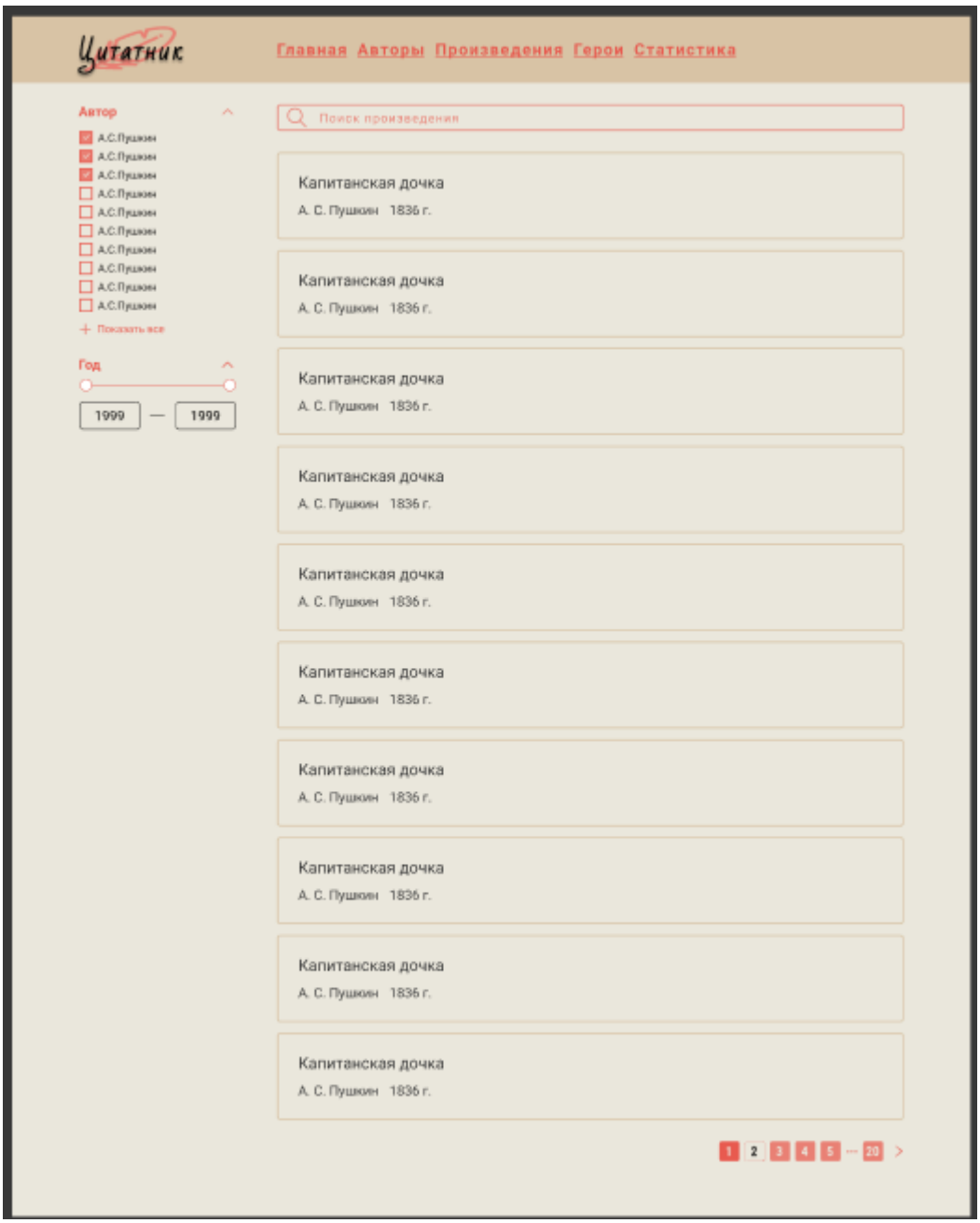


Рисунок 4 - Страница произведений.

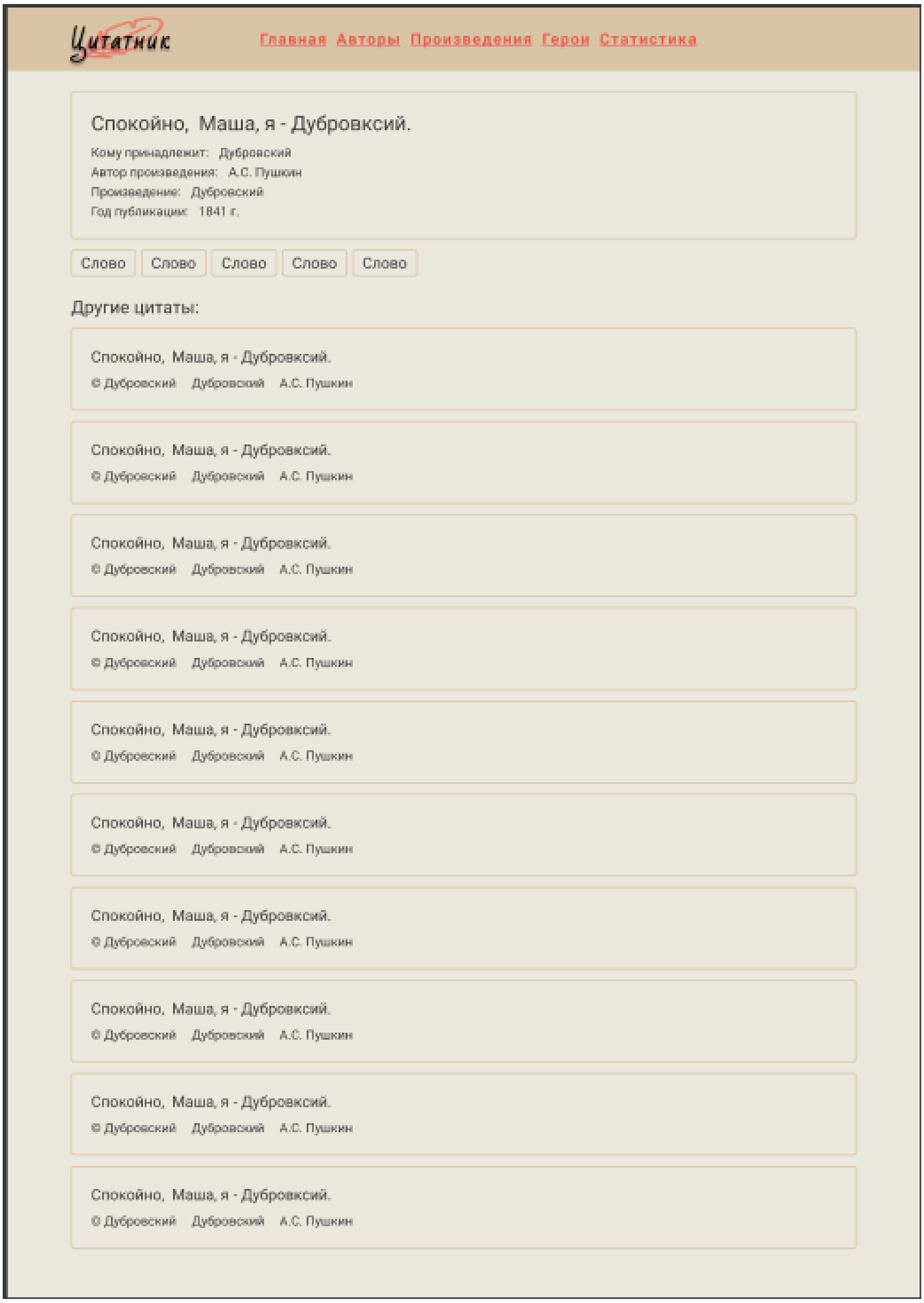
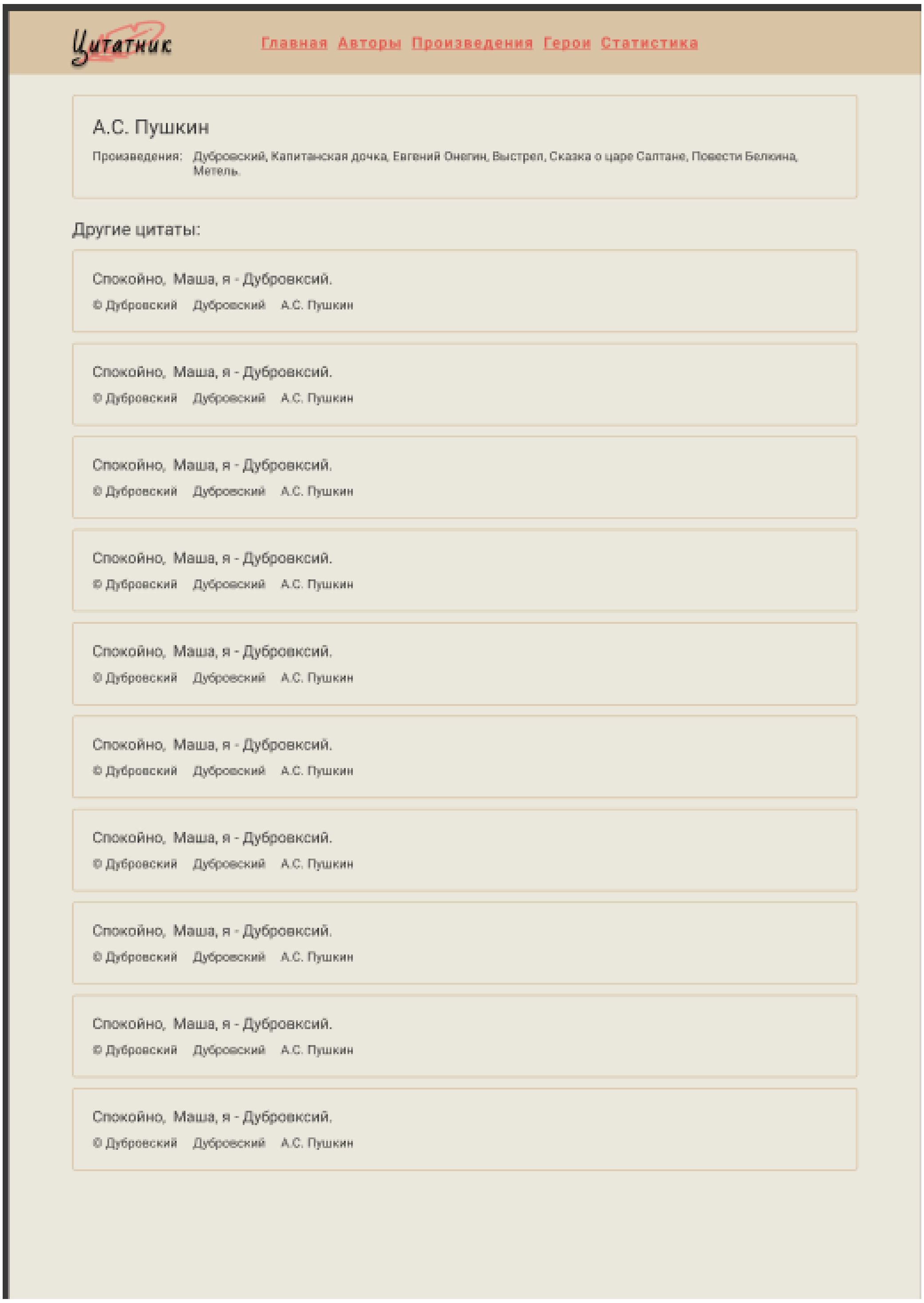


Рисунок 5 - Страница детальной информации о цитате.



Страница 6 - Страница детальной информации об авторе.

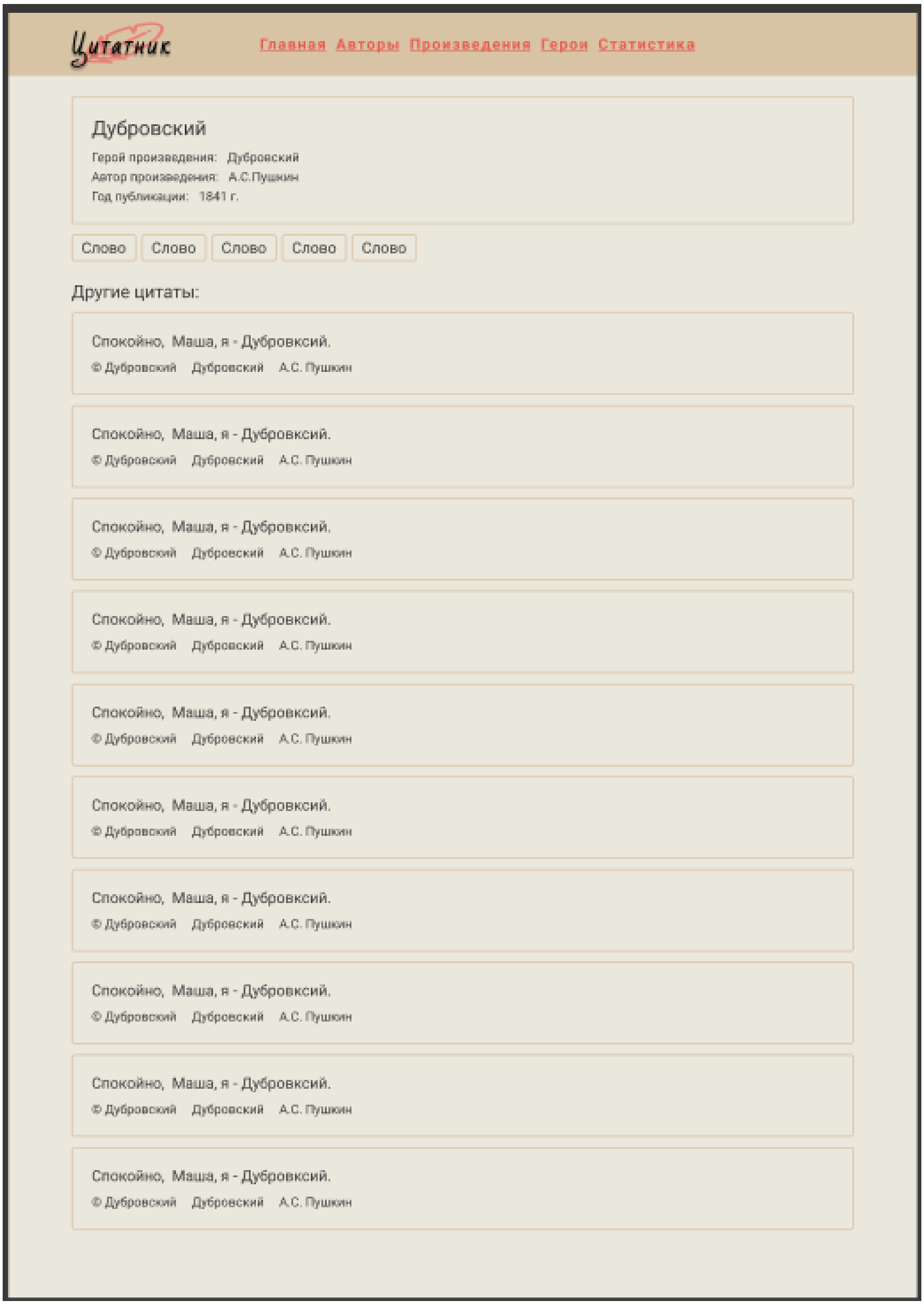


Рисунок 7 - Страница детальной информации о герое.

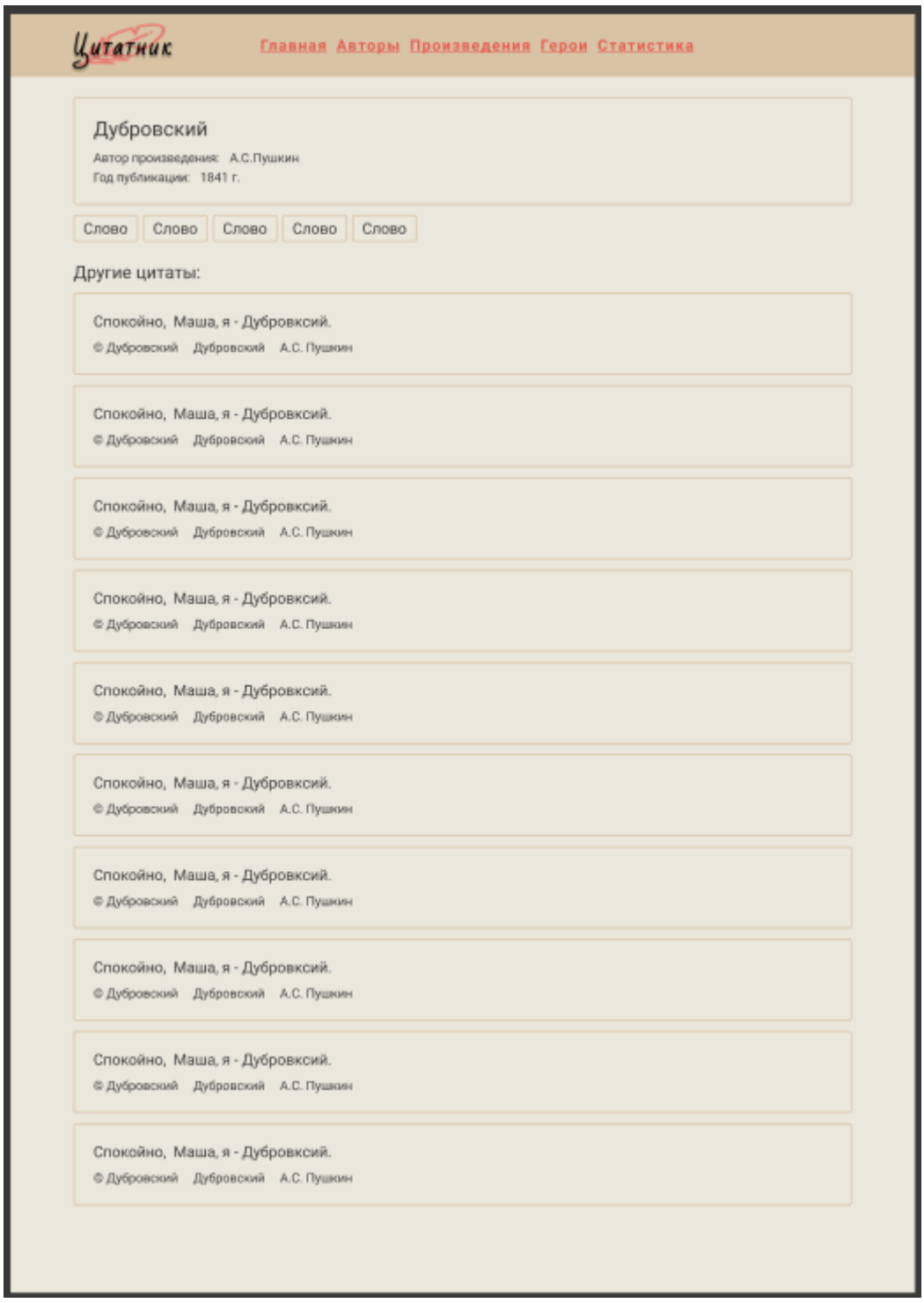


Рисунок 8 - Страница детальной информации о произведении.

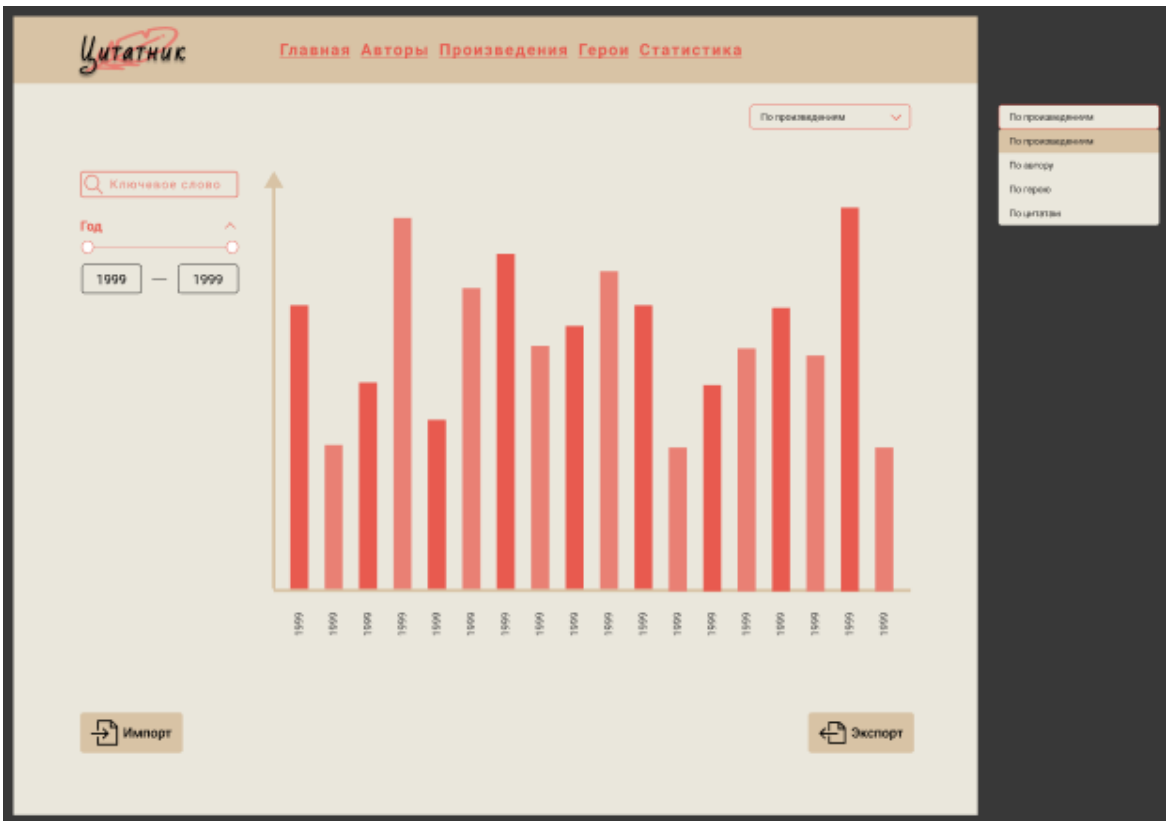


Рисунок 9 - Страница статистики.

2.2. Сценарии использования для импорта данных

Описание:

Пользователь загружает данные в базу.

Уровень:

Подфункция.

Триггер:

Алерт об успешной загрузке данных.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь нажимает на вкладку "Статистика".
2. Пользователь нажимает на кнопку "Импортировать".
3. Пользователь выбирает файл, который необходимо импортировать.

Исключения:

Отсутствуют.

Постусловия:

Вариант успешного завершения:

Пользователь получает сообщение об успешно выполненной операции

Минимальные гарантии:

Отсутствуют.

Вариант неудачного завершения:

Алерт, сообщающий об ошибке.

Исключения и следующие шаги:

Исключения:

- Файл, не соответствует необходимому формату данных

Следующие шаги:

- Взаимодействие с приложением

2.3. Сценарии использования для представления данных

2.3.1. Сценарий использования: Открытие главной страницы

Описание:

Пользователь открывает приложение или же нажимает на вкладку "Главная".

Уровень:

Цель пользователя.

Триггер:

Пользователь переходит на главную страницу платформы.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь открывает приложение или нажимает на вкладку "Главная".

Исключения:

Отсутствуют.

Постусловия:

Вариант успешного завершения:

Пользователь перенаправляется на главную страницу приложения.

Минимальные гарантии:

Отсутствуют.

Вариант неудачного завершения:

Отсутствует.

Исключения и следующие шаги:

Исключения:

- Отсутствуют

Следующие шаги:

- Взаимодействие с приложением

2.3.2. Сценарий использования: Поиск цитаты по слову

Описание:

Пользователь на главной странице в поисковую строку вводит слово (предложение), по которому будет показан список цитат.

Уровень:

Подфункция.

Триггер:

На главной странице отображается список цитата, в которых содержится слово (предложение), которое ввел пользователь.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь заходит на главную страницу
2. В поисковую строку вводит слово (предложение).

Исключения:

Отсутствуют. В случае, если не будет найдена ни одна цитата, будет возвращен пустой список.

Постусловия:

Вариант успешного завершения:

На главной странице отображается список цитат, в которых есть введенное пользователем слово (предложение).

Минимальные гарантии:

Пользователю вернется пустой список.

Вариант неудачного завершения:

Пользователю вернется пустой список.

Исключения и следующие шаги:

Исключения:

- Отсутствуют

Следующие шаги:

- Взаимодействие с приложением

2.3.3. Сценарий использования: Поиск автора/ произведения/ героя

Описание:

Пользователь на странице поиск автора / поиск произведения / поиск героя в поисковую строку вводит автора / произведение / героя, по которому будет показан искомый автор / произведение / герой.

Уровень:

Подфункция.

Триггер:

На странице поиск автора / поиск произведения / поиск героя отображается искомый автор / произведение / герой, который(ое) ввел пользователь.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь заходит на главную страницу
2. Нажимает на вкладку Автор / Произведение / Герой
3. В поисковую строку вводит автора / произведение / героя .

Исключения:

Отсутствуют. В случае, если не будет найден автор / произведение / герой, будет возвращен пустой список.

Постусловия:

Вариант успешного завершения:

На странице поиск автора / поиск произведения / поиск героя отображается искомый автор / произведение / герой, который(ое) ввел пользователь.

Минимальные гарантии:

Пользователю вернется пустой список.

Вариант неудачного завершения:

Пользователю вернется пустой список.

Исключения и следующие шаги:

Исключения:

- Отсутствуют

Следующие шаги:

- Взаимодействие с приложением

2.3.4. Сценарий использования: Настройка фильтров на главной странице

Описание:

Пользователь отмечает интересующие его параметры, по которым будут отфильтрованы цитаты.

Уровень:

Подфункция.

Триггер:

На главной странице отображается список цитата, отфильтрованный по выбору пользователя.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь заходит на главную страницу
2. Выбирает необходимые фильтры.

Исключения:

Отсутствуют. В случае, если не будет найдена ни одна цитата, будет возвращен пустой список.

Постусловия:

Вариант успешного завершения:

На главной странице отображается список цитат, отфильтрованный по выбору пользователя.

Минимальные гарантии:

Пользователю вернется пустой список.

Вариант неудачного завершения:

Пользователю вернется пустой список.

Исключения и следующие шаги:

Исключения:

- Отсутствуют

Следующие шаги:

- Взаимодействие с приложением

2.3.5. Сценарий использования: Просмотр детальной информации о цитате, об авторе цитаты, произведения, в котором содержится цитата, о герое, который сказал выбранную цитату

Описание:

Пользователь имеет возможность просматривать детальную информацию о каждой цитате, об авторе цитаты, произведения, в котором содержится цитата, о герои, который сказал выбранную цитату, а также похожие цитаты.

Уровень:

Подфункция.

Триггер:

Нажатие на карточку цитаты/на автора цитаты на карточке цитаты/на произведение, в котором содержится цитата, на карточке цитаты/на героя, который сказал цитату, на карточке цитаты.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь нажимает на карточку цитаты/на автора цитаты на карточке цитаты/на произведение, в котором содержится цитата, на карточке цитаты/на героя, который сказал цитату, на карточке цитаты.

Исключения:

Исключение отсутствуют.

Постусловия:

Вариант успешного завершения:

Пользователь видит страницу с детальной информацией о цитате, авторе, произведении или герое, а также похожие цитаты.

Минимальные гарантии:

Пользователь видит страницу с детальной информацией о цитате, авторе, произведении или герое, а также похожие цитаты.

Исключения и следующие шаги:

Исключения:

- Отсутствуют

Следующие шаги:

- Взаимодействие с приложением

2.4. Сценарии использования для анализа данных

Описание:

Пользователь нажимает на вкладку "Статистика".

Уровень:

Подфункция.

Триггер:

Пользователь переходит на страницу статистики платформы.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь нажимает на вкладку "Статистика".

Исключения:

Отсутствуют.

Постусловия:

Вариант успешного завершения:

Пользователь перенаправляется на страницу статистики приложения.

Минимальные гарантии:

Отсутствуют.

Вариант неудачного завершения:

Отсутствует.

Исключения и следующие шаги:

Исключения:

- Отсутствуют

Следующие шаги:

- Взаимодействие с приложением

2.5. Сценарии использования для экспорта данных

Описание:

Пользователь выгружает данные из базы.

Уровень:

Подфункция.

Триггер:

Алерт об успешной загрузке данных.

Действующее лицо:

Основным действующим лицом является пользователь.

Второстепенные лица:

Отсутствуют.

Предусловия:

Пользователь [Действующее лицо] должен иметь активное интернет-соединение.

Основной успешный сценарий:

1. Пользователь нажимает на вкладку "Статистика".
2. Пользователь нажимает на кнопку "Экспортировать".

Исключения:

Отсутствуют.

Постусловия:

Вариант успешного завершения:

Пользователь получает сообщение об успешно выполненной операции.

Минимальные гарантии:

Отсутствуют.

Вариант неудачного завершения:

Алерт, сообщающий об ошибке.

Исключения и следующие шаги:

Исключения:

- Отсутствует

Следующие шаги:

- Взаимодействие с приложением

3. МОДЕЛЬ ДАННЫХ

3.1. Нереляционная модель данных.

Графическое представление модель:

Пример хранения данных в бд для модели с героем

```
{ _id: ObjectId('67044b0f86eb0f319342b9dc'), quote: 'Нельзя надеяться на женскую верность; счастлив, кто смотрит на это равнодушно.', hero: 'Корсаков', authorName: 'Александр Сергеевич Пушкин', book: { name: 'Арап Петра Великого', year: 1913 } }
```

Пример хранения данных в бд для модели без героя

```
{ _id: ObjectId('67044b0f86eb0f319342b9bd'), quote: '... Что ум, любя простор, теснит,\n' + 'Что слишком часто разговоры\n' + 'Принять мы рады за дела...', book: { name: 'Евгений Онегин', year: 1859 }, authorName: 'Александр Сергеевич Пушкин' }
```

Оценка объема информации, хранимой в моделях:

Приведем объем в памяти:

1) Размер всей коллекции (в коллекции 628 элементов): 333325 байт.
2) Средний размер одного объекта: 530 байт.

Так как модель денормализованна, то фактически зависимость объема пропорциональна количеству объектов. Т.е если принять следующие обозначения

n - количество объектов

$V(n)$ - общий объем коллекции

avgDoc - средний объем документа

Получим формулу:

$V(n) = n * avgDoc$

Так как *avgDoc* равен 530 байт, формула зависимости примет вид:

$V(n) = 530 * n$

Примеры запросов для совершения сценариев использования

1. **5Открытие главной страницы:**

```
const quotes = await Quote.find().limit(pageLimit).exec();
```

2. Поиск цитаты по ключевому слову:

```
const quotes = await Quote.find({ quote: new RegExp(keyword, 'i')
}).exec();
```

3. Поиск автора / произведение / героя:

3.1 Поиск автора:

```
const author = await Quote.find(
      { authorName: new RegExp(authorName, 'i') }, // Фильтр по имени
автора (регистронезависимый)
      { authorName: 1 } // Выводим только имя автора
    ).exec();
```

3.2. Поиск книги:

```
const filter: any = {
      'book.name': new RegExp(bookName, 'i'), // Фильтр по названию
книги (регистронезависимый)
      ...(bookName && { 'book.name': bookName, 'i') }), // Фильтр по
названию книги, если оно передано
      ...(bookYear && { 'book.year': bookYear }) // Фильтр по году
публикации, если он указан
    };

const books = await Quote.find(filter, {
      'book.name': 1, // Вывести название книги
      'book.year': 1, // Вывести год публикации
      authorName: 1 // Вывести имя автора
    }).exec();
```

3.3. Поиск героя:

```
const filter: any = {
      heroName: new RegExp(heroName, 'i'), // Фильтр по имени героя
      ...(bookName && { 'book.name': bookName, 'i') }), // Фильтр по
названию книги, если оно передано
      ...(authorName && { authorName: authorName, 'i') }) // Фильтр
по автору, если оно передано
    };

const hero = await Quote.find(filter, {
      heroName: 1, // Вывести имя героя
      'book.name': 1, // Вывести название книги
      authorName: 1 // Вывести имя автора
    }).exec();
```

4. Настройка фильтров на главной странице:

```
const filter = {
      ...(authorName && { authorName }),              // Фильтр по
автору, если он указан
      ...(bookName && { 'book.name': bookName }),   // Фильтр по
названию книги, если указан
      ...(bookYear && { 'book.year': bookYear }),  // Фильтр по году
книги, если указан
      ...(heroName && { heroName })                // Фильтр по имени
героя, если указан
    };
const quotes = await Quote.find(filter).exec();
```

5. Просмотр детальной информации:

5.1. Просмотр информации об авторе:

```
const result = await Quote.aggregate([
      { $match: { authorName: new RegExp(`^${authorName}$`, 'i') }
},
      {  $group:  {  _id:  "$authorName",  books:  {  $addToSet:
"$book.name" } } }
    ]).exec();
```

5.2. Просмотр информации о книге:

```
const result = await Quote.findOne(
      { 'book.name': bookName },
      { 'book.name': 1, 'book.year': 1, authorName: 1 }
    ).exec();
```

5.3. Просмотр информации о герое:

```
const result = await Quote.findOne(
      { heroName: heroName },
      {
        heroName: 1,
        'book.name': 1,
        'book.year': 1,
        authorName: 1
        }
    ).exec();
```

6. Открытие страницы статистики статистика по авторам:

```
const result = await Quote.aggregate([
      { $group: { _id: "$authorName", count: { $sum: 1 } } }  //
Группировка по авторам и подсчет количества цитат
```

```
    ]).exec();
статистика по книгам
    const result = await Quote.aggregate([
      { $group: { _id: "$book.name", count: { $sum: 1 } } }   //
Группировка по названию книг и подсчет цитат
    ]).exec();
статистика по героям
    const result = await Quote.aggregate([
      { $group: { _id: "$heroName", count: { $sum: 1 } } }   //
Группировка по героям и подсчет цитат
    ]).exec();
статистика по цитатам
    const result = await Quote.aggregate([
      { $match: { quote: new RegExp(word, 'i') } },  // Фильтрация
по слову в цитате (регистронезависимый поиск)
      { $group: { _id: "$book.year", count: { $sum: 1 } } }   //
Группировка по годам и подсчет количества цитат
    ]).exec();
```

7. Импортирование данных:

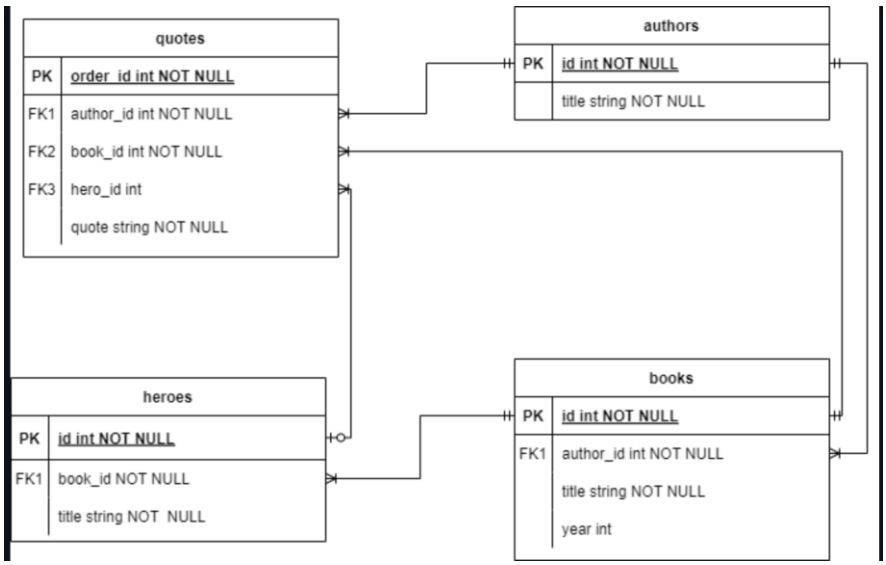
```
const newQuote = new Quote({
      authorName: authorName,
      book: { name: bookName, year: bookYear },
      quote: quoteText
    });

await newQuote.save();
```

8. Экспортирование данных:

```
const quotes = await Quote.find().exec();
```

3.2. Аналог модели для SQL СУБД.

Графическое представление:



Описание удельного объема информации, хранимой в модели:

Приведем объем в памяти

Размер всей коллекции (в коллекции 628 элементов): 434176 байт.

Средний размер одного объекта с полной информацией: 694 байт.

Так как модель нормализована, то объем зависит от того, насколько много у сущностей(автора, книги, героя) цитат.

n - количество объектов

V(n) - общий объем коллекции

avgRow - средний объем одного объекта с полной информацией

Получим формулу *V(n) = n * avgRow*

Зная avgRow, получим формулу

*V(n) = 694 * n*

Примеры запросов:

1. Открытие главной страницы

```
SELECT q.quote, a.title AS author, h.title AS hero, b.title AS book
FROM quotes q
LEFT JOIN authors a ON q.author_id = a.id
LEFT JOIN heroes h ON q.hero_id = h.id
LEFT JOIN books b ON q.book_id = b.id
```

```
ORDER BY q.id DESC
LIMIT @page_limit;
```

2. Поиск цитаты по слову

```
SELECT q.quote AS quote, a.title AS author, h.title AS hero, b.title
AS book
FROM quotes q
LEFT JOIN authors a ON q.author_id = a.id
LEFT JOIN heroes h ON q.hero_id = h.id
LEFT JOIN books b ON q.book_id = b.id
WHERE q.quote ILIKE @quote_regex
```

где quote_regex регулярное выражения вида '%keyword%', где keyword - строка введённая пользователем.

3. Поиск автора / произведение / героя

3.1. Поиск автора

```
SELECT id, title
FROM authors
WHERE title ILIKE @author_regex;
```

где author_regex регулярное выражения вида '%author%', где author- - строка введенная пользователем.

3.2. Поиск героя

```
SELECT h.id AS id, h.title AS hero_name, b.title AS book_name,
a.title AS author_name
FROM heroes h
JOIN books b ON h.book_id = b.id
JOIN authors a ON b.author_id = a.id
WHERE h.title ILIKE @hero_regex
  AND b.id = @book_id
  AND a.id = @author_id;
```

где hero_regex регулярное выражения вида '%name%', где name - строка введенная пользователем.

3.3.Поиск книги

```
SELECT b.id AS id, b.title AS book_name, a.title AS author_name,
b.year as year
FROM books b
JOIN authors a ON b.author_id = a.id
WHERE b.title ILIKE @book_regex'
  AND a.id = @author_id;
```

где book_regex регулярное выражения вида '%book%', где book - строка введенная пользователем.

4. Настройка фильтров на главной странице

```
SELECT q.id AS id, q.quote AS qoute, a.title AS author, h.title AS
hero, b.title AS book
FROM quotes q
LEFT JOIN authors a ON q.author_id = a.id
LEFT JOIN heroes h ON q.hero_id = h.id
LEFT JOIN books b ON q.book_id = b.id
WHERE a.id = @author_id AND b.year = @year AND h.id = @hero_id;
```

5. Просмотр детальной информации

5.1. Просмотр детальной информации о цитате

```
SELECT q.quote AS quote, a.title AS author, h.title AS hero, b.title
AS book
FROM quotes q
LEFT JOIN authors a ON q.author_id = a.id
LEFT JOIN heroes h ON q.hero_id = h.id
LEFT JOIN books b ON q.book_id = b.id
WHERE q.id = @quote_id;
```

5.2.Просмотр детальной информации об авторе

```
SELECT a.title AS author_name, array_agg(b.title) AS books
FROM authors a
JOIN books b ON a.id = b.author_id
WHERE a.id = @author_id
```

5.3.Просмотр детальной информации о герое

```
SELECT h.title AS hero_name, b.title AS book_name, a.title AS
author_name, b.year AS year
FROM heroes h
JOIN books b ON h.book_id = b.id
JOIN authors a ON b.author_id = a.id
WHERE b.id = @author_id
```

5.4.Просмотр детальной информации о книге

```
SELECT b.title AS book_name, a.title AS author_name, b.year AS year
FROM books b
JOIN authors a ON b.author_id = a.id
WHERE b.id = @book_id
```

6. Открытие страницы статистики

6.1. Статистика по авторам

```
SELECT a.title AS author, COUNT(q.id) AS quotes_count
FROM quotes q
JOIN authors a ON q.author_id = a.id
GROUP BY a.title;
```

6.2. Статистика по книгам

```
SELECT b.title AS book, COUNT(q.id) AS quotes_count
FROM quotes q
JOIN books b ON q.book_id = b.id
GROUP BY b.title;
```

6.3. Статистика по героям

```
SELECT h.title AS hero, COUNT(q.id) AS quotes_count
FROM quotes q
JOIN heroes h ON q.hero_id = h.id
GROUP BY h.title;
```

6.4.Статистика по цитатам

```
SELECT book.year, COUNT(*) AS quote_count
FROM quotes
WHERE quote ILIKE @qoute_regex
GROUP BY book.year
ORDER BY book.year;
```

где qoute_regex регулярное выражения вида '%qoute%', где qoute - строка введенная пользователем.

7. Импорт данных для автора

```
INSERT INTO authors (title) VALUES (@title);
для книги
INSERT INTO books (title, year, author_id) VALUES (@title, @year, @author_id);
для героя
INSERT INTO heroes (title, book_id) VALUES (@title, @book_id);
для цитаты
INSERT INTO quotes (quote, author_id, hero_id, book_id) VALUES (@quote, @author_id, @hero_id, @book_id);
```

8. Экспортирование данных

```
SELECT q.quote AS qoute, a.title AS author, h.title AS hero, b.title AS book
FROM quotes q
LEFT JOIN authors a ON q.author_id = a.id
LEFT JOIN heroes h ON q.hero_id = h.id
LEFT JOIN books b ON q.book_id = b.id;
```

3.3. Сравнение моделей

Удельный объем хранения

Хотя коэффицент избыточности у нереляционной базы данных больше, однако на текущей выборке количество памяти в MongoDb больше, чем в PostgreSQl. Можно сделать вывод, что при малом количестве авторов(героев, книг) и большом количестве цитат выгоднее использовать реляционную модель, в противном случае - нереляционную

Запросы по отдельным юзкейсам

Сравнения количества запросов

Открытие главной страницы: 1 SQL / 1 NoSQL

Поиск цитаты по слову: 1 SQL / 1 NoSQL

Поиск автора / произведение / героя: 1 SQL / 1 NoSQL (для всех случаев)

Настройка фильтров на главной странице: 1 SQL / 1 NoSQL (для всех случаев)

Просмотр детальной информации автора / произведение / героя / цитаты : 1 SQL / 1 NoSQL (для всех случаев)

Открытие страницы статистики: 1 SQL / 1 NoSQL (для всех случаев)

Импортирование данных: 4 SQL / 1 NoSQL

Экспортирование данных: 1 SQL / 1 NoSQL

Количество задействованных коллекций

Открытие главной страницы: 4 SQL / 1 NoSQL

Поиск цитаты по слову: 4 SQL / 1 NoSQL

Поиск автора / произведение / героя: 1,2,3 соответственно SQL / 1 NoSQL

Настройка фильтров на главной странице: 4 SQL / 1 NoSQL (для всех случаев)

Просмотр детальной информации: 1 SQL / 1 NoSQL (для всех случаев)

Открытие страницы статистики автора / произведение / героя / цитаты : 1,2,3,4 соответственно SQL / 1 NoSQL (для всех случаев)

Импортирование данных: 4 SQL / 1 NoSQL

Экспортирование данных: 4 SQL / 1 NoSQL

4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

4.1. Краткое описание

Код приложения разделен на две части: back-end и front-end.

Back-end реализован с использованием фреймворка NestJS. На back-end части реализовано обращение к базе данных и передача данных на front-end часть проекта.

Front-end обращается к back-end части и отображает полученные данные в виде списка карточек.

При разработке использовался архитектурный паттерн MVC. Model - это база данных, view - отображение на front-end части и controller - обработка данных на back-end части.

В проекте выделены страницы и основные компоненты отображаемых элементов.

4.2. Использованные технологии

БД: MongoDB.

Back-end: TypeScript, NestJS.

Front-end: React.

4.3. Схема экрана приложения

Схема экранов приложения представлена на рисунке 10.

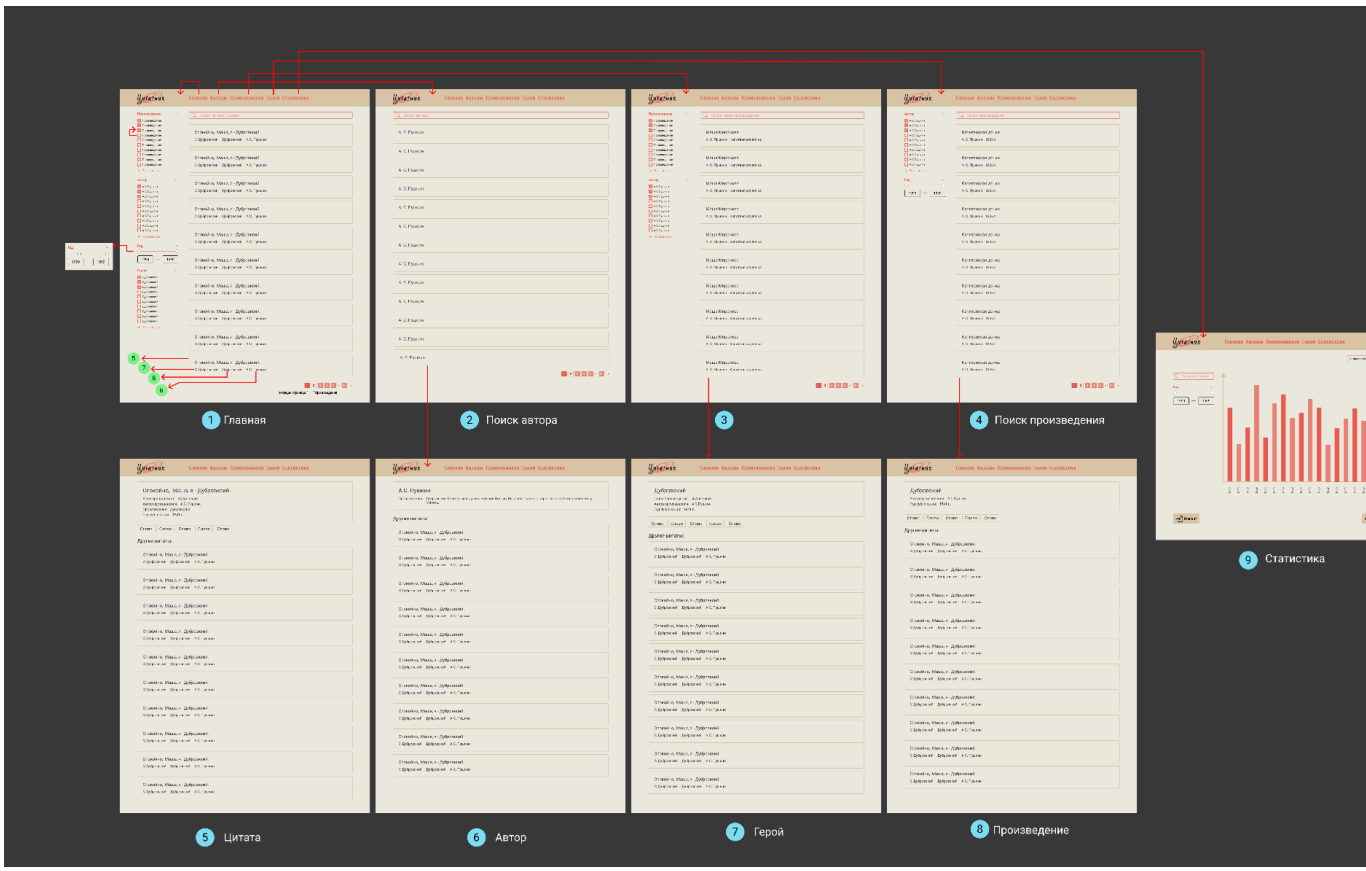


Рисунок 10 - Схема экранов приложения.

5. ВЫВОДЫ

5.1. Достигнутые результаты

В ходе работы реализовано приложение “Цитатник”, представляющее собой каталог литературных цитат, который позволяет пользователям просматривать цитаты и детальною информацию о них.

А также можно отфильтровать цитаты по авторам, героям, произведениям и годам публикации. Посмотреть информацию детальную информацию об авторах, о произведениях и о героях.

Кроме того, приложение предоставляет возможность просмотра статистики, импорта и экспорта данных.

Если говорить о выбранном решении хранении данных, то основываясь на оценке, проведенной выше, можно сделать следующие заключения:

По удельному объёму реляционная модель будет оптимальнее чем нереляционная.

По количеству запросов к бд модели примерно равны. Различие наблюдается только при импорте данных в пользу нереляционной модели.

По числу используемых коллекций лучший результат показывает нереляционная модель.

MongoDB является более гибким вариантом и позволяет легче делать добавление новых объектов, однако она значительно увеличивает количество избыточных данных. SQL модель является более эффективной в использовании памяти и сложных запросов, а также её легче поддерживать при масштабируемости приложения. Учитывая простоту структур, используемых в приложении и его функционал, MongoDB является оптимальным вариантом за счёт лёгкости и гибкости операций.

5.2. Недостатки и пути для улучшения

На данный момент в приложении отсутствует возможность регистрации и авторизации, что не позволяет пользователям оставлять комментарии и выбирать понравившиеся цитаты.

5.3. Будущее развитие решения

Планируется добавление личного кабинета пользователя, чтобы каждый пользователь мог собирать собственный каталог понравившихся цитат и оставлять комментарии в общем каталоге.

7. ПРИЛОЖЕНИЕ

7.1. Документация по сборке и развертыванию приложения

1) Скачать репозиторий (указан в ссылках на приложение).
2) В корне проекта запустить комманду *docker-compose up*
3) Открыть приложение по адресу http://127.0.0.1:3000

7.2. Инструкция для пользователя

- **Просмотр информации**

Для того, чтобы просмотреть цитаты, необходимо перейти на главный экран приложения. Чтобы увидеть определенные цитаты, можно выбрать определенные фильтры или ввести слово в поисковую строку.

Также можно посмотреть детальную информацию о каждой цитате. Необходимо нажать на текст цитаты. Откроется страница с детальной информацией о цитате. По такому же алгоритму можно посмотреть список авторов, произведений и героев, а также детальную информацию о них.

- **Экспортирование данных**

Необходимо перейти на страницу статистики и нажать на кнопку “Экспорт”. При нажатии на кнопку скачается файл в формате .json.

- **Импортирование данных**

Необходимо перейти на страницу статистики и нажать на кнопку “Импорт”. При нажатии на кнопку приложение предложит выбрать файл. Файл должен быть в формате .json и соответствовать схеме.

- **Просмотр статистики**

Перейти на экран статистики и выбрать интересующий параметр для просмотра статистики.

7. ЛИТЕРАТУРА

1. Ссылка на GitHub. - [Электронный ресурс]. - URL: https://github.com/moevm/nosql2h24-citations

2. Документация MongoDB. - [Электронный ресурс]. - URL: https://www.mongodb.com/ (дата обращения: 9.11.2024)

3. Документация React. - [Электронный ресурс]. - URL: https://ru.react.js.org/docs/getting-started.html (дата обращения: 9.11.2024)

4. Chart.js: Bar Chart. - [Электронный ресурс]. - URL: https://proglib.io/p/sozdavayte-diagrammy-na-react-s-pomoshchyu-biblioteki-chart-js (дата обращения: 9.11.2024)

5. Документация ReactSlider. - [Электронный ресурс]. - URL: https://www.npmjs.com/package/react-slider (дата обращения: 9.11.2024)