МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине «Введение в нереляционные базы данных»

Тема: Онлайн-кинотеатр с каталогом фильмов

Студентка гр. 2384 _____ Пчелинцева К. Р.

Студент гр. 2384 _____ Дамакин Р. П.

Студентка гр. 2384 _____ Матеюк Д. С.

Студент гр. 2384 _____ Тимченко Д. А.

Студент гр. 2384 _____ Шурыгин Д. Л.

Преподаватель _____ Заславский М. М.

Санкт-Петербург

2025

ЗАДАНИЕ

Студентка Пчелинцева К. Р.

Студент Дамакин Р. П.

Студентка Матеюк Д. С.

Студент Тимченко Д. А.

Студент Шурыгин Д. Л.

Группа 2384

Тема: Онлайн-кинотеатр с каталогом фильмов

Исходные данные:

Необходимо создать веб-приложение для онлайн-кинотеатра с базой данных фильмов и сериалов, возможностью их добавления, редактирования, поиска и просмотра с использованием СУБД MongoDB.

Содержание пояснительной записки:

«Содержание»

«Введение»

«Сценарии использования»

«Модель данных»

«Разработанное приложение»

«Выводы»

«Приложения»

«Список использованных источников»

Предполагаемый объем пояснительной записки:

Не менее 15 страниц.

Дата выдачи задания: 5.02.2025

Дата сдачи реферата: 20.05.2025

Дата защиты реферата: 22.05.2025

Студентка гр. 2384		Пчелинцева К. Р.
Студент гр. 2384		Дамакин Р. П.
Студентка гр. 2384		Матеюк Д. С.
Студент гр. 2384		Тимченко Д. А.
Студент гр. 2384		Шурыгин Д. Л.
Преподаватель		Заславский М. М.

АННОТАЦИЯ

В рамках проекта разработано веб-приложение онлайн-кинотеатра. Приложение включает функционал для добавления, редактирования и удаления фильмов и сериалов, а также их просмотра. Реализована система поиска и сортировки контента, возможность просмотра информации об актерах и режиссерах. Поддерживается экспорт и импорт данных для резервного копирования.

В процессе разработки были использованы технологии HTML/CSS/JavaScript для клиентской части, Python (Flask) для серверной части, СУБД MongoDB для хранения данных и Docker.

Исходный код доступен по ссылке: nosql1h25-cinema

SUMMARY

The project developed a web application for an online cinema. The application includes functionality for adding, editing and deleting films and TV series, as well as viewing them. A system for searching and sorting content, the ability to view information about actors and directors has been implemented. Export and import of data for backup is supported.

During the development process, HTML/CSS/JavaScript technologies were used for the client part, Python (Flask) for the server part, MongoDB DBMS for data storage and Docker.

The source code is available at the link: nosql1h25-cinema

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1.1. Актуальность проблемы

С ростом популярности онлайн-кинотеатров и увеличением спроса на цифровой контент возрастает потребность в удобных и функциональных платформах для управления медиатекой. Пользователи часто сталкиваются с проблемами неудобного поиска контента, отсутствия структурированной информации о фильмах и сериалах, а также сложностями в администрировании собственных медиабиблиотек. Существующие решения не всегда предлагают гибкие инструменты для каталогизации, сортировки и анализа данных, а также не обеспечивают удобного взаимодействия с информацией об актерах и режиссерах.

1.2. Постановка задачи.

Задача проекта заключается в разработке веб-приложения, которое позволяет администраторам и пользователям:

- Добавлять, редактировать и удалять фильмы и сериалы, включая их метаданные (название, год выпуска, описание и др.);
- Организовывать структуру сериалов по сезонам и эпизодам;
- Просматривать информацию об актерах и режиссерах, участвовавших в создании контента;
- Осуществлять поиск и сортировку фильмов и сериалов по различным критериям;
- Экспортировать и импортировать данные для резервного копирования;

Кроме того, приложение должно обеспечить:

- Удобный интерфейс для просмотра контента;
- Надежное хранение данных в СУБД MongoDB;
- Высокую производительность и отказоустойчивость системы.

1.3. Предлагаемое решение.

Для реализации создается веб-приложение с использованием HTML/CSS/JavaScript для клиентской части, Flask для серверной части, MongoDB для хранения данных и Docker для контейнеризации и развертывания. Приложение обеспечивает удобное управление каталогом фильмов и сериалов, включая их просмотр и администрирование.

1.4. Качественные требования к решению.

Разрабатываемое веб-приложение должно быть удобным и интуитивно понятным для пользователей, обеспечивать выполнение операций фильтрации, поиска и просмотра медиаконтента, легко масштабироваться и расширяться за счет модульной архитектуры, поддерживать надежное хранение данных в MongoDB, а также обеспечивать быстрое развертывание на различных платформах благодаря использованию Docker.

2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

2.1. Макет UI

Ниже представлен макет приложения (рис. 1-29)

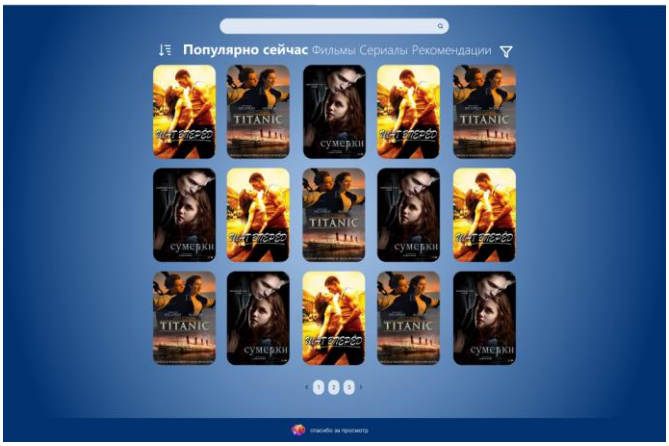

Рисунок 1. Главная страница с подборкой фильмов

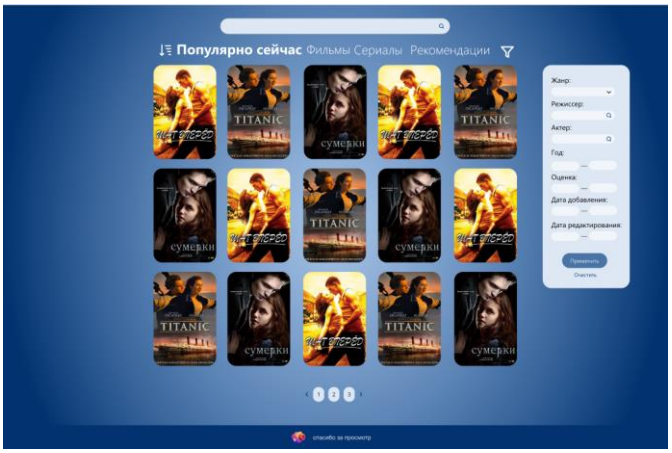

Рисунок 2. Интерфейс фильтрации контента

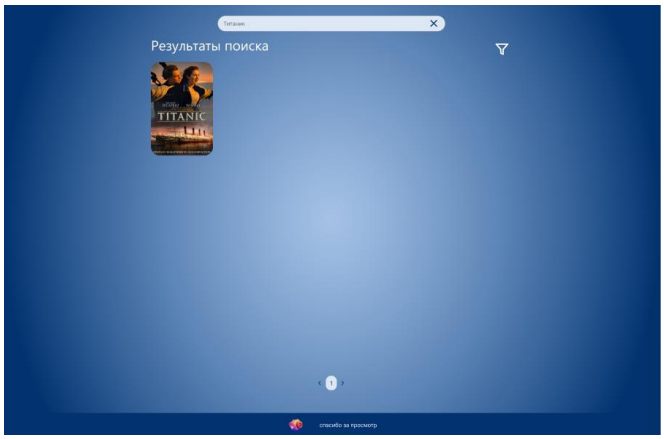

Рисунок 3. Результаты поиска фильма

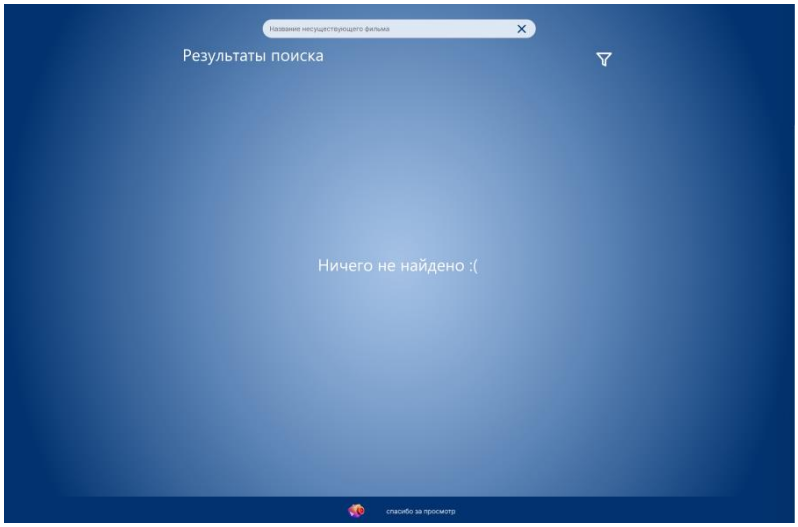

Рисунок 4. Сообщение "Ничего не найдено" при неудачном поиск

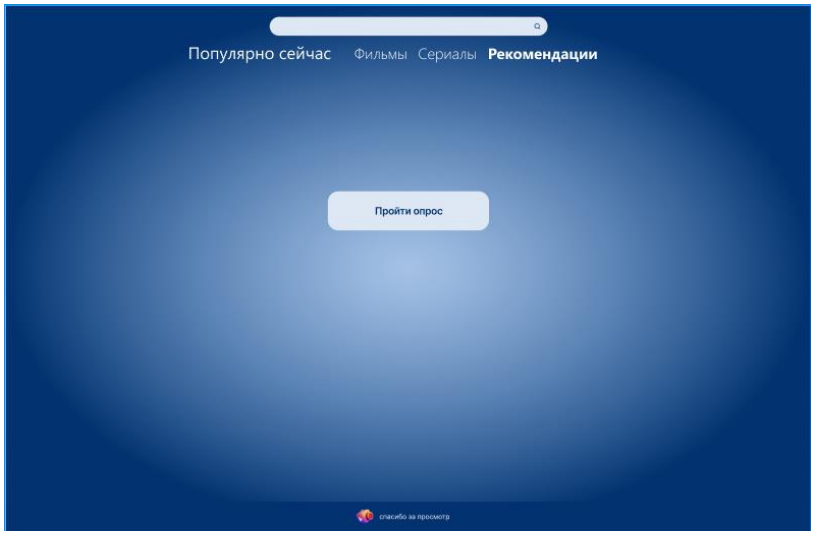

Рисунок 5. Форма рекомендаций с кнопкой "Пройти опрос"

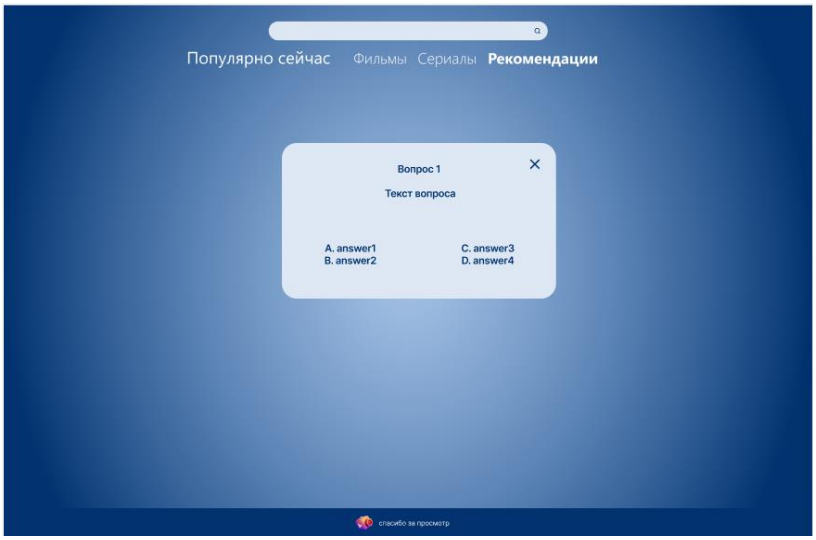

Рисунок 6. Интерфейс опроса для формирования рекомендаций

Рисунок 7. Персонализированная подборка фильмов после опроса

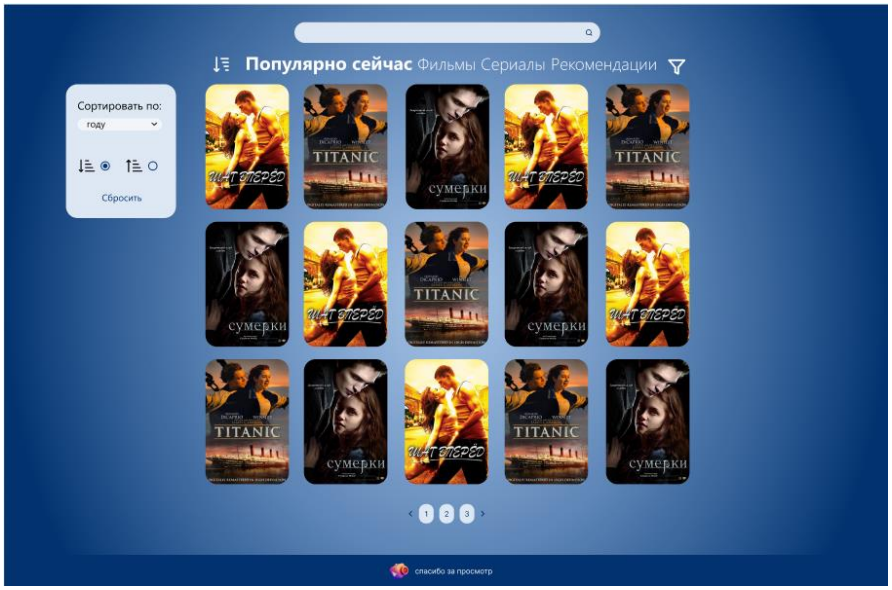

Рисунок 8. Интерфейс сортировки фильмов

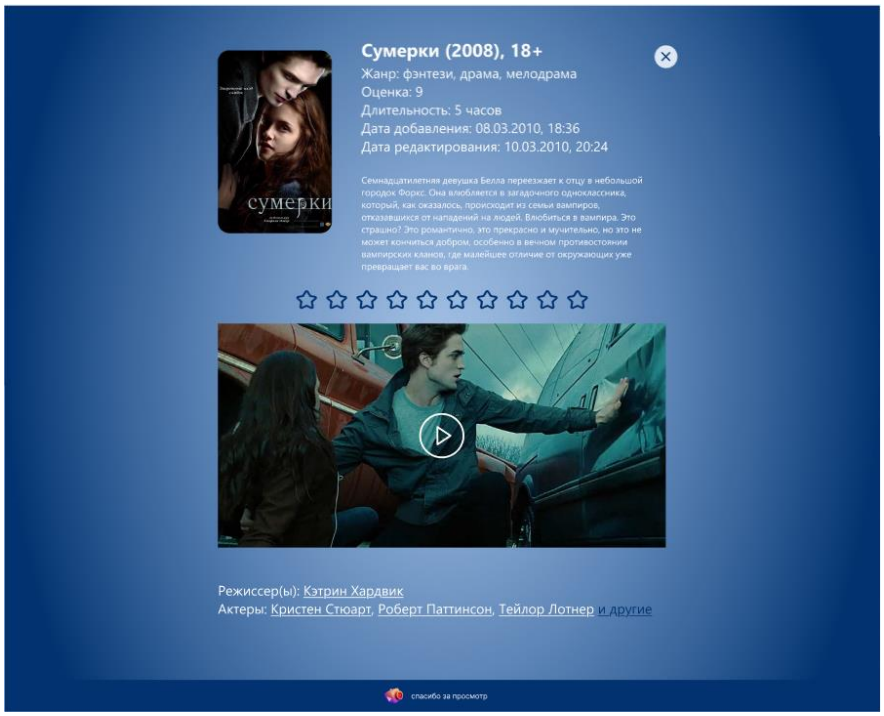

Рисунок 9. Страница фильма с основными данными

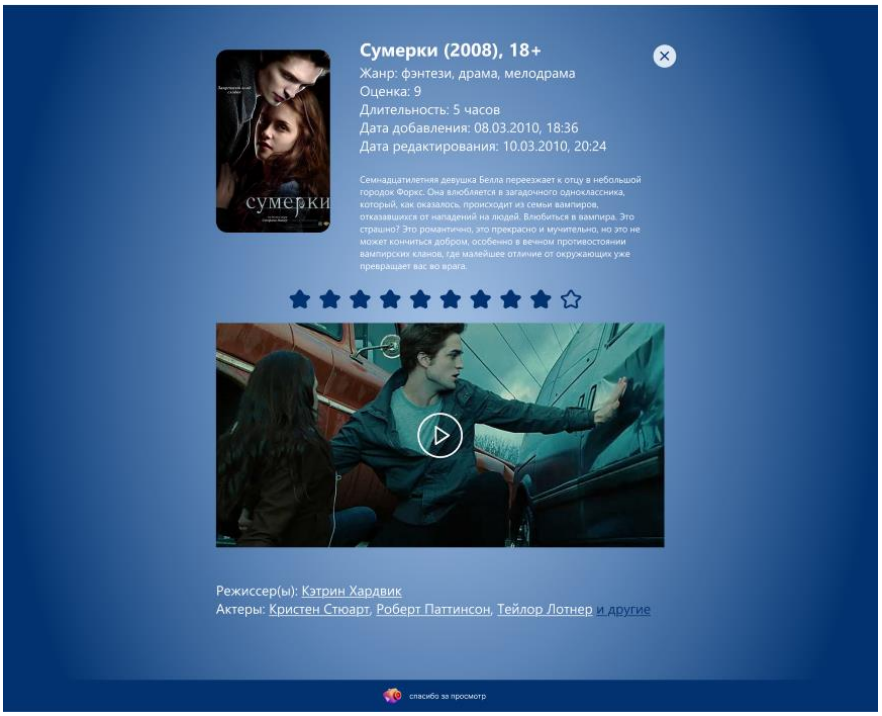

Рисунок 10. Система оценки фильма (звёздный рейтинг)

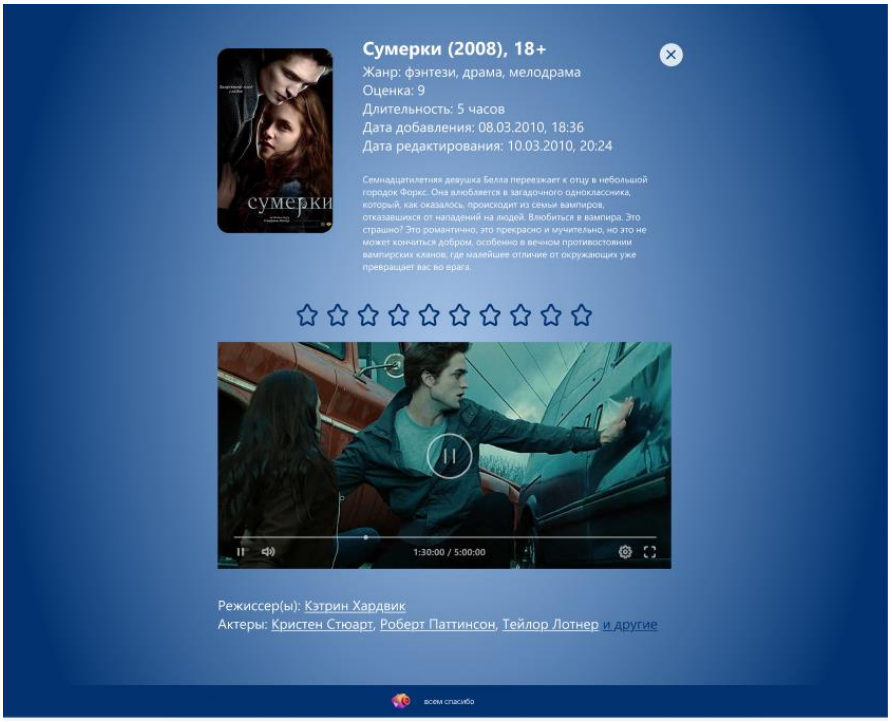

Рисунок 11. Проигрыватель в режиме воспроизведения

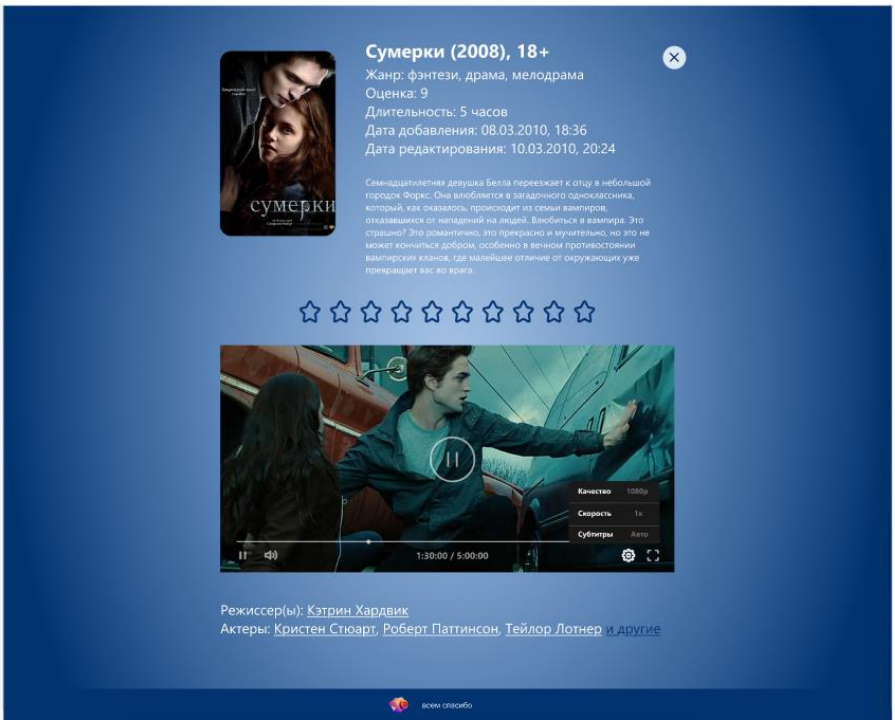

Рисунок 12. Меню настроек проигрывателя

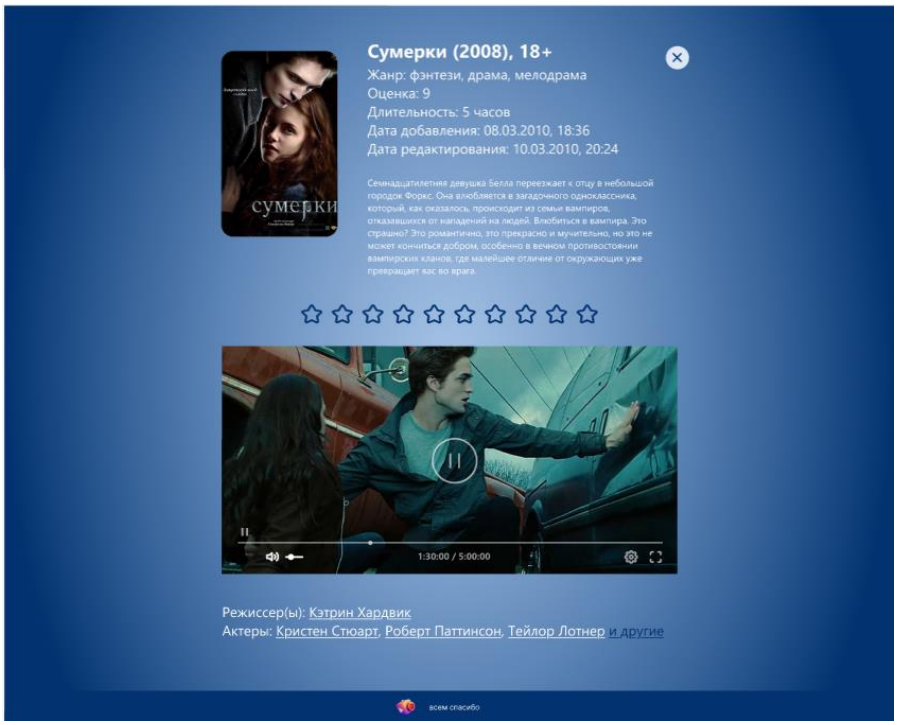

Рисунок 13. Регулировка громкости в проигрывателе

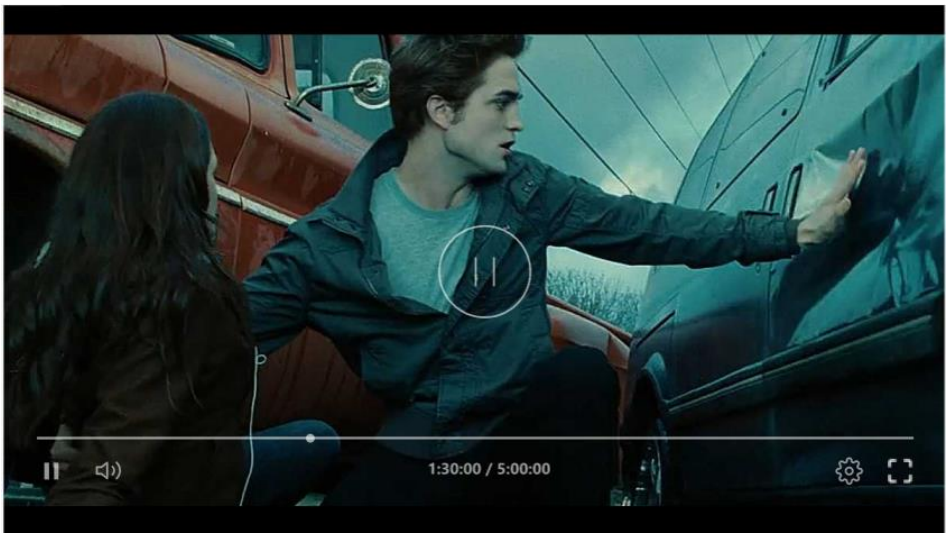

Рисунок 14. Проигрыватель в полноэкранном режиме

Рисунок 15. Перемотка видео с помощью ползунка таймера

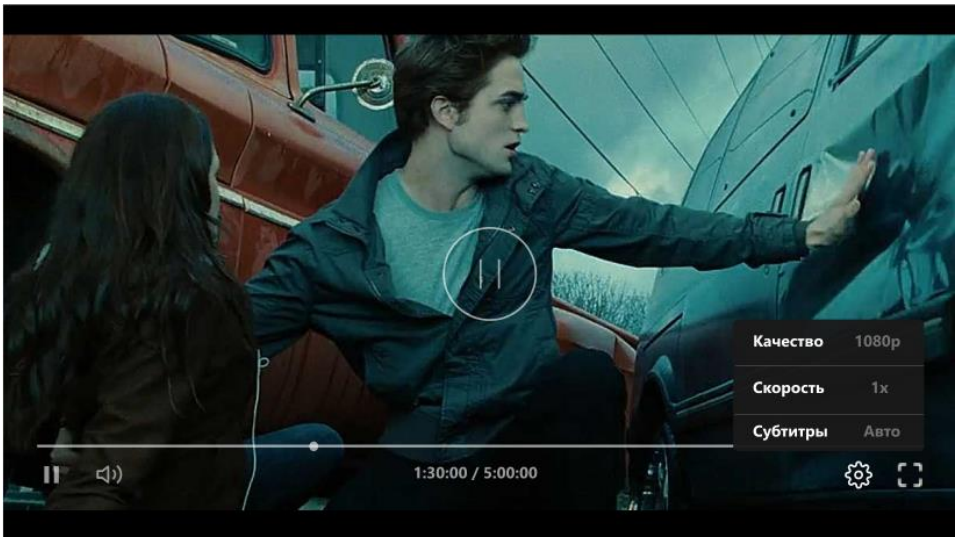

Рисунок 16. Меню настроек в полноэкранном режиме проигрывателя

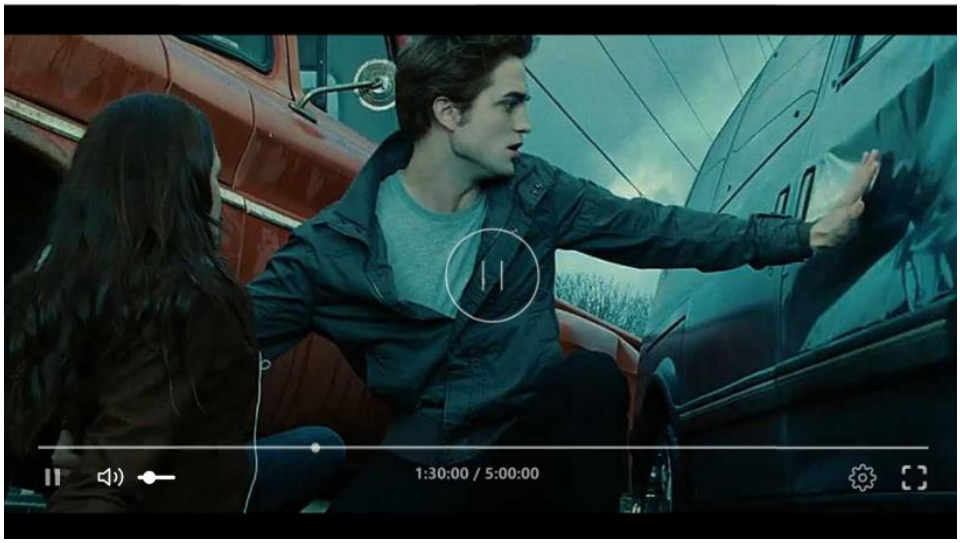

Рисунок 17. Регулировка громкости в полноэкранном режиме

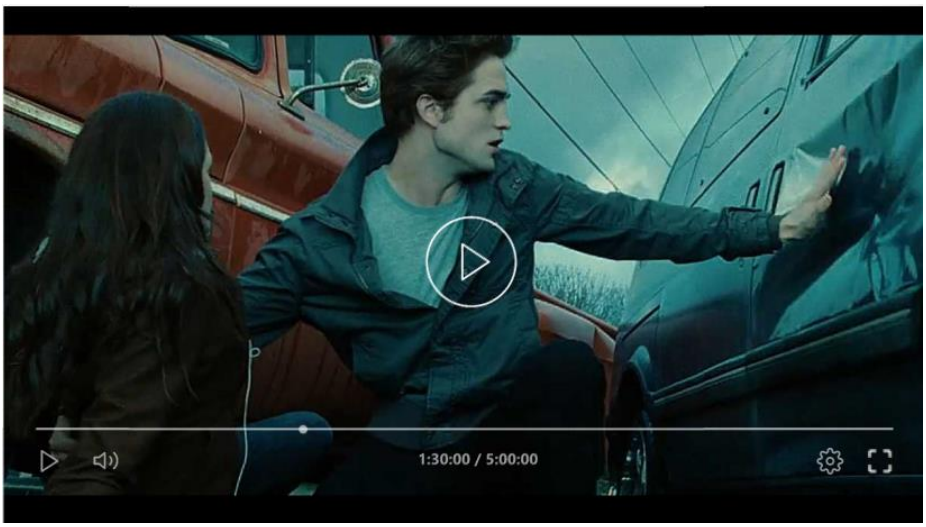

Рисунок 18. Проигрыватель в режиме паузы (полноэкранный режим)

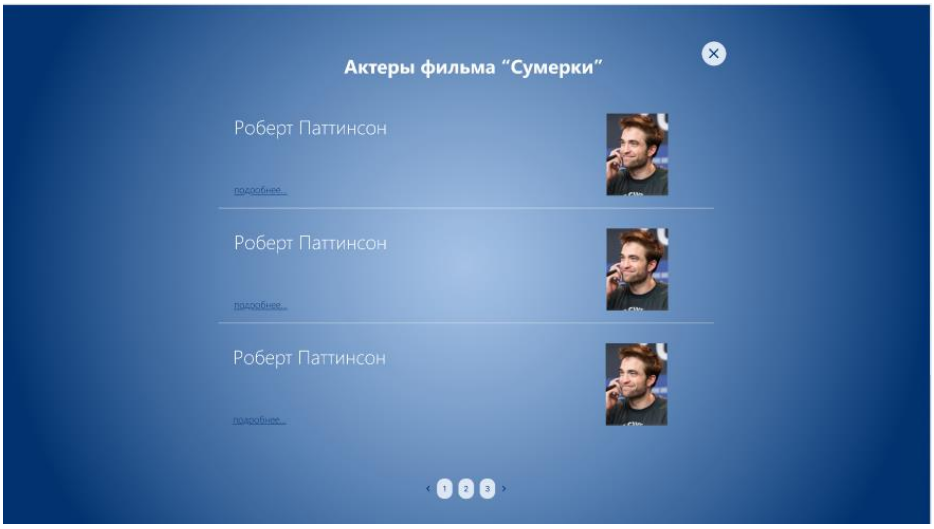

Рисунок 19. Подробная страница с информацией об актёрах фильма

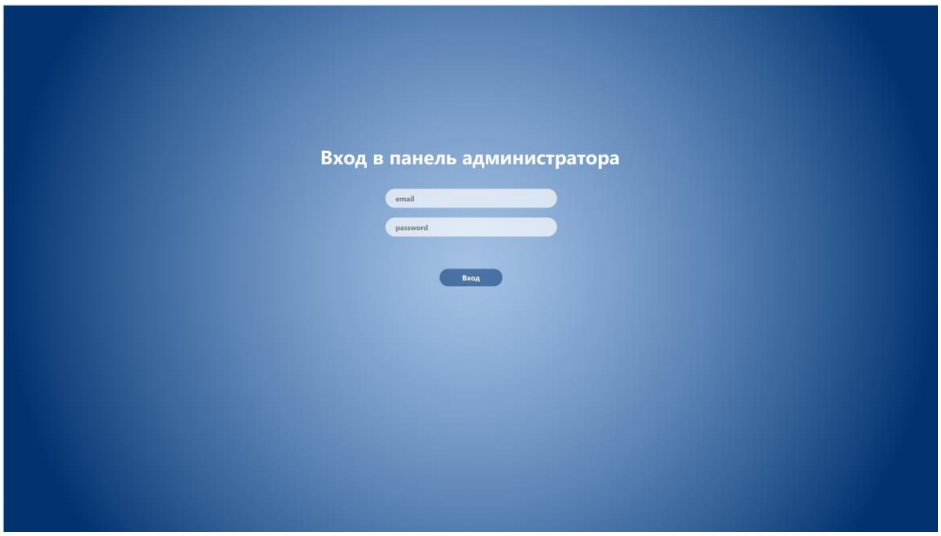

Рисунок 20. Интерфейс авторизации в админ-панель

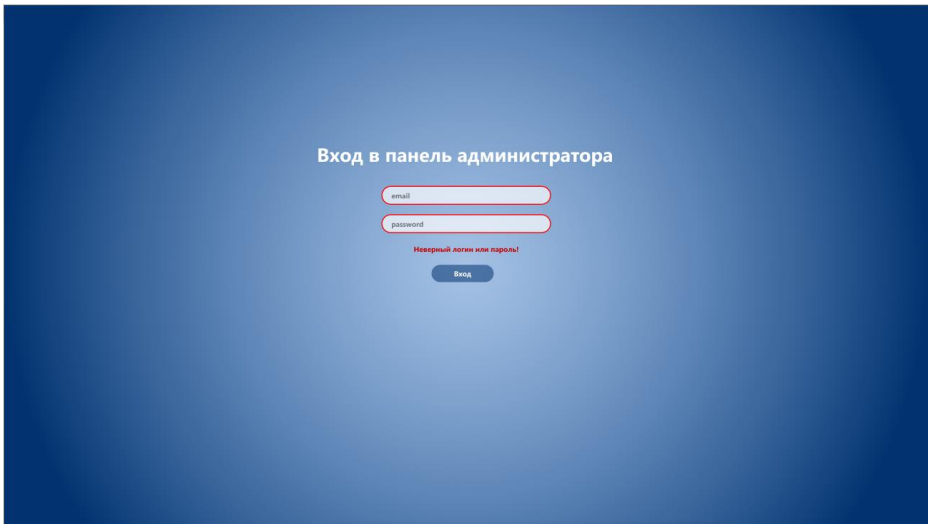

Рисунок 21. Сообщение об ошибке при авторизации (неверные данные)

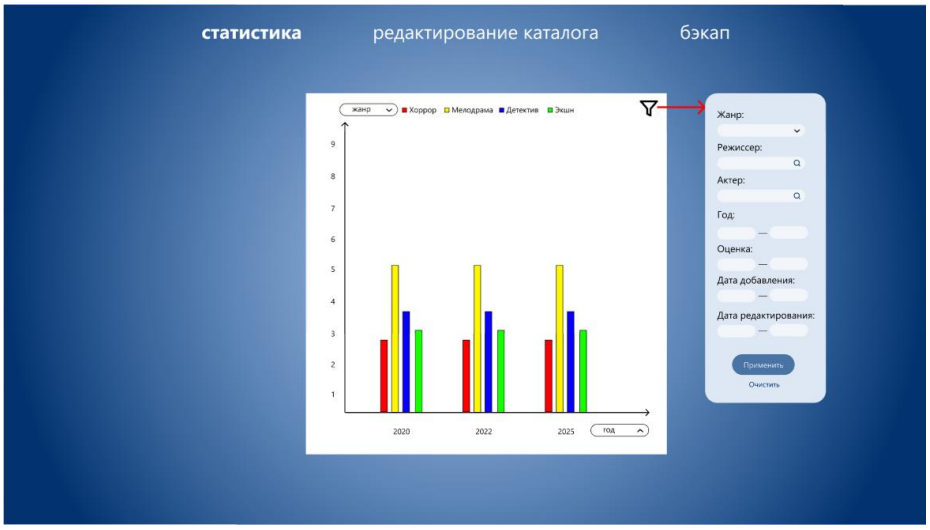

Рисунок 22. Страница статистики

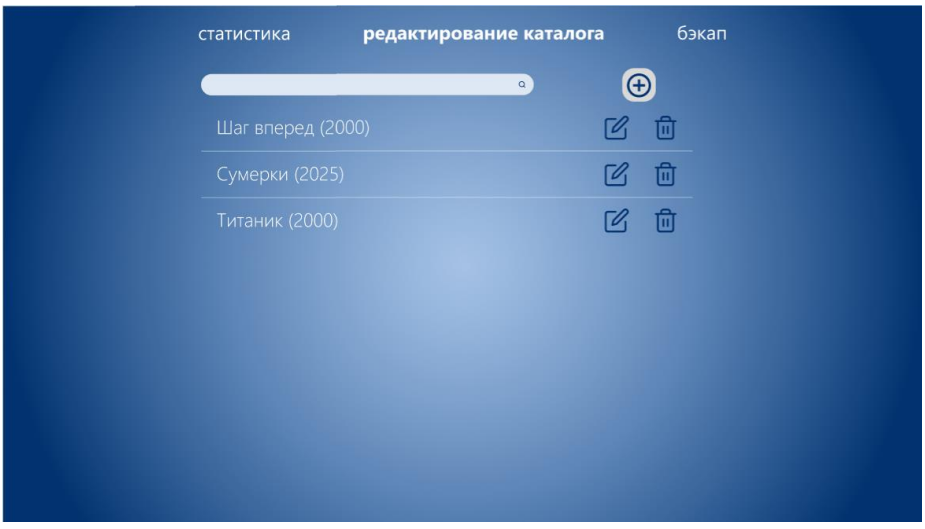

Рисунок 23. Интерфейс редактирования каталога фильмов

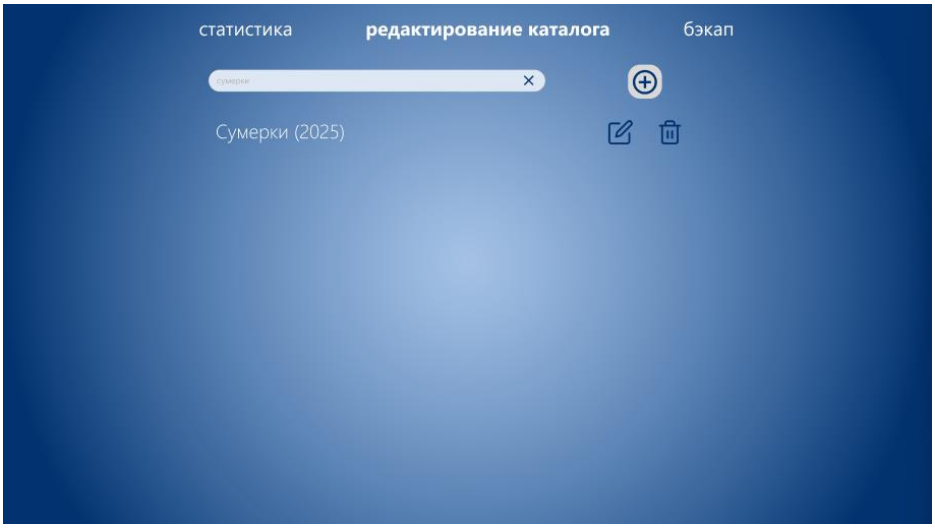

Рисунок 24. Поиск фильмов в режиме редактирования каталога

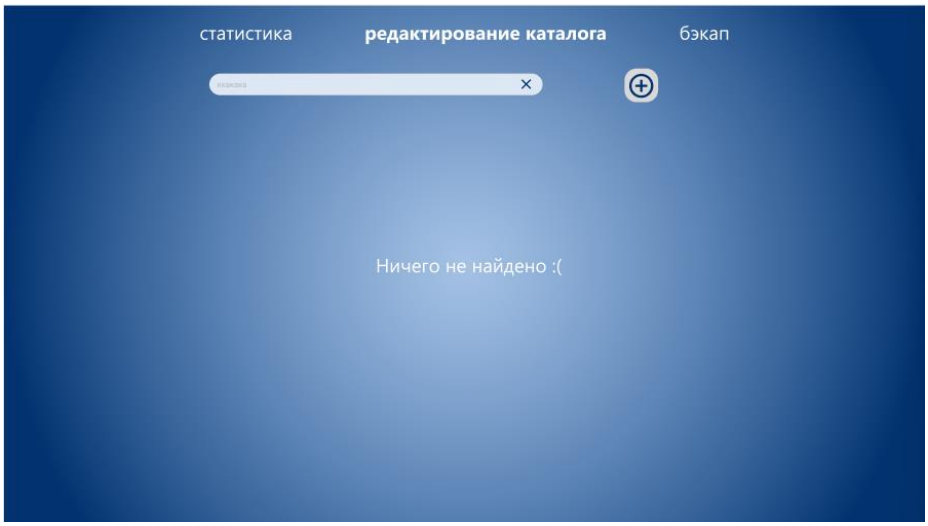

Рисунок 25. Сообщение "Ничего не найдено" при поиске в каталоге

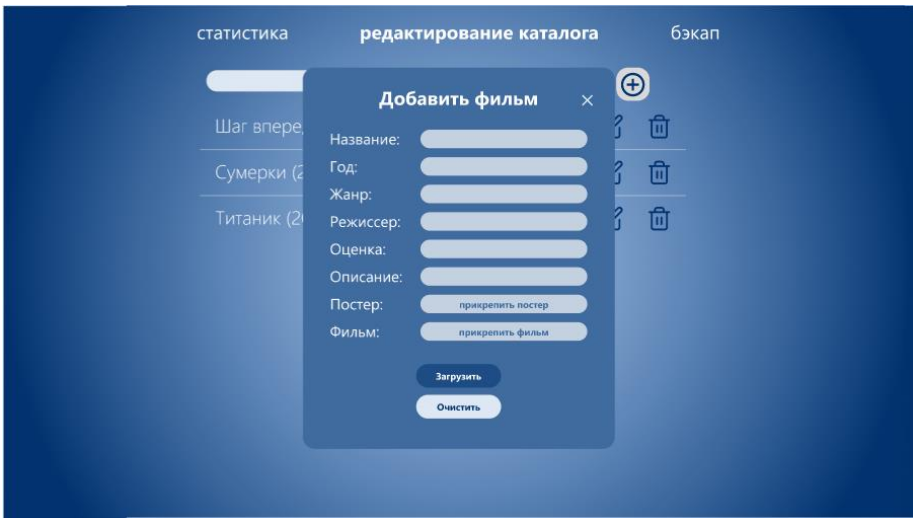

Рисунок 26. Форма добавления нового фильма

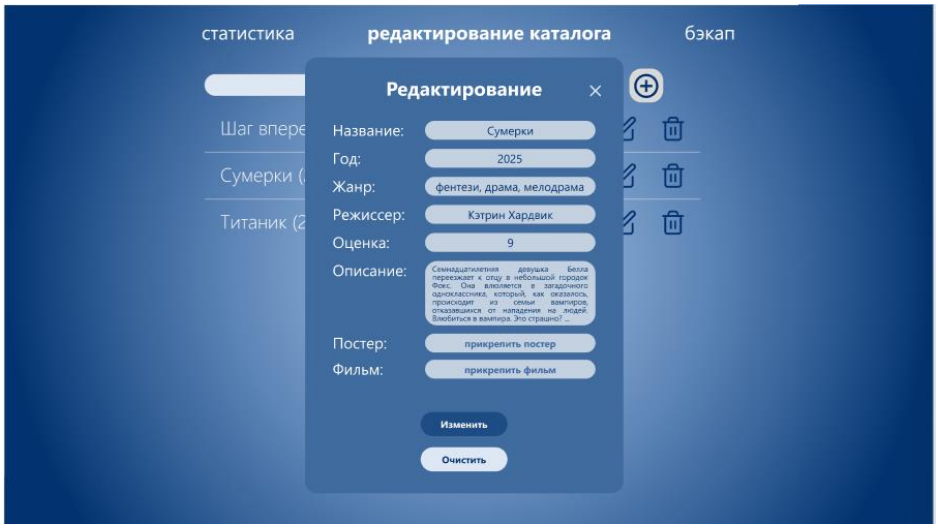

Рисунок 27. Форма редактирования данных о фильме

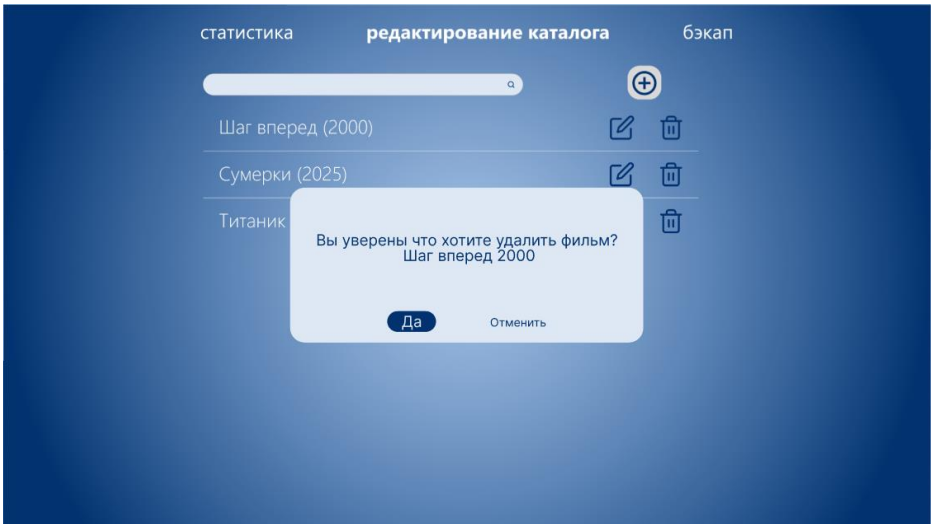

Рисунок 28. Диалог подтверждения удаления фильма

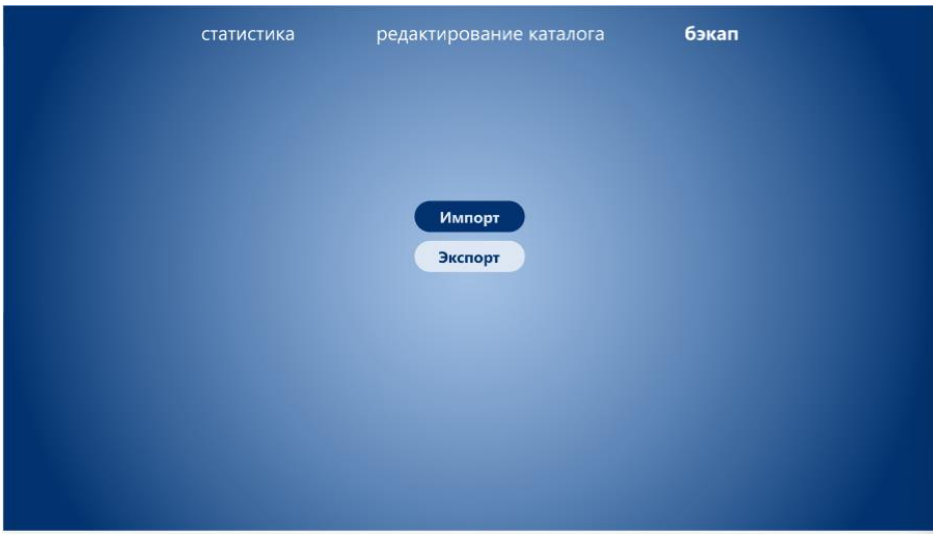

Рисунок 29. Интерфейс управления резервными копиями (бэкапами)

2.2. Сценарий использования: Главная/подборка.

Цель: Пользователь хочет найти и выбрать фильм для просмотра.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь заходит на сайт.
2. Видит список фильмов в разделе "Популярно сейчас".
3. Выбирает фильм из списка, нажимая на его карточку, переходит на страницу фильма.

Альтернативные сценарии:

- Пользователь хочет найти конкретный фильм — вводит название фильма в строку поиска и нажимает кнопку "Найти". Система отображает результаты поиска. Пользователь выбирает фильм и переходит на его страницу.
- Пользователь хочет выбрать фильм по критериям — нажимает кнопку "Фильтр".
- Пользователь хочет просмотреть больше фильмов на главной странице — нажимает на стрелку "вперед" или "назад" для перелистывания либо выбирает номер страницы для быстрого перехода.
- Пользователь хочет просмотреть список всех фильмов или сериалов:
    - Нажимает на вкладку "Фильмы", "Сериалы" или "Рекомендации".
    - Система отображает соответствующий контент.
    - Пользователь выбирает нужный фильм/сериал.
- Если по запросу или фильтрам фильмов не найдено, система выводит сообщение "Фильмы не найдены. Попробуйте изменить параметры поиска".

Результат сценария: Пользователь выбрал фильм для просмотра.

2.3. Сценарий использования: Сортировка фильмов.

Цель: Пользователь хочет отсортировать фильмы по определенному признаку

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку "сортировка".
2. В появившемся окне пользователь выбирает параметр сортировки и "от меньшего к большему" либо "от большего к меньшему".

Альтернативные сценарии:

- Пользователь передумал насчет параметра сортировки и нажал на кнопку "сбросить", чтобы заново ввести нужный параметр.

Результат сценария: Пользователь получил список фильмов, отсортированный по определенному параметру.

2.4. Сценарий использования: Получение рекомендаций.

Цель: Пользователь хочет получить персональные рекомендации по фильмам и сериалам.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь заходит на сайт и выбирает вкладку "Рекомендации".
2. Система предлагает пройти опрос из N вопросов, где пользователь выбирает один из нескольких вариантов ответа на каждый вопрос.
3. После завершения опроса система анализирует ответы и формирует персональный список рекомендаций.
4. Пользователь видит список рекомендованных фильмов и сериалов и может выбрать один для просмотра.

Альтернативные сценарии:

- Пользователь хочет выйти из опроса, не заполнив его — нажимает кнопку "Отмена" или закрывает страницу.
- Пользователь не отвечает на все вопросы, система выводит сообщение "Пожалуйста, ответьте на все вопросы, чтобы получить точные рекомендации".

Результат сценария: Пользователь получил персональные рекомендации и выбрал фильм или сериал для просмотра.

2.5. Сценарий использования: Страница фильма.

Цель: Пользователь хочет посмотреть выбранный ранее фильм.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь оказывается на странице.
2. Видит описание фильма (год, жанр, оценку, режиссер, длительность, актеры, которые в нем снимались, дата добавления, дата редактирования), проигрыватель.
3. Нажимает на кнопку "плэй" и смотрит фильм.
4. Если хочет, выставляет оценку, выбирая количество звезд от 1 до 10.
5. Система фиксирует оценку, обновляет средний рейтинг и выводит сообщение "Ваша оценка сохранена".

Альтернативные сценарии:

- Пользователь не хочет смотреть фильм и, нажимая на крестик, возвращается на главную страницу.
- Пользователь уже оценивал фильм — система обновляет его предыдущую оценку.
- Пользователь хочет посмотреть информацию об актере/режиссере для этого нажимает по его фамилии в списке ниже проигрывателя.

- Если произошла ошибка при отправке оценки, система выводит сообщение "Ошибка сохранения оценки. Попробуйте позже".

Результат сценария: Пользователь ознакомился с фильмом.

2.6. Сценарий использования: Фильтры.

Цель: Пользователь хочет получить список фильмов, подобранных по введенным параметрам.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку "Фильтр".
2. Пользователь видит окно с фильтрами.
3. Пользователь вводит свои требования к параметрам в соответствующие поля и нажимает на кнопку "Применить".
4. Пользователь получает список фильмов по выбранным параметрам.

Альтернативные сценарии:

- Пользователь хочет закрыть фильтры и нажимает на кнопку "Фильтр", всплывающее окно исчезает.
- Пользователь хочет убрать все фильтры и нажимает на кнопку "Очистить фильтры", затем нажимает "Применить", получая исходный список фильмов.

Результат сценария: Пользователь получил желаемый список фильмов по определенным параметрам.

2.7. Сценарий использования: Поиск фильма.

Цель: Пользователь хочет найти конкретный фильм.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь вводит название фильма в строку поиска и нажимает на кнопку "Поиск".
2. Пользователь получает список фильмов, названия которых соответствуют введенному тексту.

Альтернативные сценарии:

- Пользователь ввел несуществующий фильм, система выводит сообщение "Ничего не найдено :(".
- Пользователь хочет вернуться на главную, очищает строку поиска и нажимает кнопку "Поиск".

Результат сценария: Пользователь нашел необходимый фильм.

2.8. Сценарий использования: Вход в админ-панель.

Цель: Администратор хочет войти в систему для управления сайтом.

Действующие лица: Администратор

Основной сценарий:

1. Администратор открывает браузер и вводит в адресную строку site.com/admin.
2. Загружается страница авторизации с полями "Email" и "Password".
3. Администратор вводит учетные данные.
4. Нажимает кнопку "Войти".
5. Если данные введены корректно, система перенаправляет администратора в админ-панель.

Альтернативные сценарии:

- Если введены неверные данные, система выводит сообщение "Неверный email или пароль" и предлагает повторить ввод.

- Если сервер временно недоступен, система выводит сообщение об ошибке и предлагает попробовать позже.

Результат сценария: Администратор успешно вошел в админ-панель и получил доступ к управлению сайтом.

2.9. Сценарий использования: Импорт базы данных.

Цель: Администратор хочет скачать базу данных.

Действующие лица: Администратор

Основной сценарий:

1. Администратор находится в админ панели во вкладке "бэкап".
2. Администратор нажимает на кнопку "Импорт".
3. Открывается проводник для скачивания .json файла.

Альтернативные сценарии:

- Ошибка при выгрузке файла:
    - Если произошла техническая ошибка (например, сервер не отвечает), система выводит сообщение "Ошибка! Не удалось загрузить файл. Попробуйте позже."
    - Администратор пробует повторить выгрузку через некоторое время.

Результат сценария: Администратор выгрузил базу данных.

2.9. Сценарий использования: Экспорт базы данных.

Цель: Администратор хочет загрузить данные в базу данных.

Действующие лица: Администратор

Основной сценарий:

1. Администратор находится в админ панели во вкладке "бэкап".
2. Администратор нажимает на кнопку "Экспорт".
3. Открывается проводник для загрузки .json файла.

Альтернативные сценарии:

- Ошибка при загрузке файла:
    - Если произошла техническая ошибка (например, сервер не отвечает), система выводит сообщение "Ошибка! Не удалось загрузить файл. Попробуйте позже.".
    - Администратор пробует повторить выгрузку через некоторое время.

Результат сценария: Администратор загрузил данные в базу данных.

2.10. Сценарий использования: Редактирование каталога.

Цель: Администратор хочет управлять каталогом фильмов: редактировать информацию, удалять фильмы, искать их в базе или добавлять новые.

Действующие лица: Администратор

Основной сценарий:

1. Администратор находится в админ панели во вкладке "Редактирование каталога".
2. Видит список всех фильмов, доступных в системе.
3. При необходимости использует поиск, вводя название фильма.
    a. Если фильм найден, отображается карточка фильма.
    b. Если фильма нет в базе, система выводит сообщение "Ничего не найдено :(".
4. Администратор может:
    a. Выбрать фильм для редактирования, изменить его данные и сохранить изменения.
    b. Нажать на кнопку "+", чтобы добавить новый фильм (переходит на страницу "Добавление фильма").
    c. Удалить фильм, нажав кнопку "Удалить".

Альтернативные сценарии:

- Удаление фильма:

- Администратор нажимает кнопку "Удалить".
- Появляется подтверждающее окно: "Вы уверены, что хотите удалить фильм?".
- Если администратор выбирает "Да", фильм удаляется из каталога.
- Если нажимает "Отменить", система закрывает окно без удаления.

Результат сценария: Администратор успешно отредактировал каталог, удалил ненужные фильмы, добавил новый фильм или нашел нужный через поиск.

2.11. Сценарий использования: Добавить фильм.

Цель: Администратор хочет добавить новый фильм на сайт.

Действующие лица: Администратор

Основной сценарий:

1. Администратор нажимает кнопку "+" и переходит в раздел "Добавить фильм".
2. Вводит данные:
    a. Название
    b. Год
    c. Жанр
    d. Режиссер
    e. Оценка
    f. Описание
3. Прикрепляет постер фильма.
4. Прикрепляет файл с фильмом.
5. Нажимает кнопку "Загрузить".
6. Система проверяет заполненность всех обязательных полей.

a. Если все данные корректны, фильм успешно добавляется в базу и отображается в списке фильмов.

Альтернативные сценарии:

- Если администратор хочет очистить введенные данные, он нажимает кнопку "Очистить", и все поля сбрасываются.
- Если какие-то обязательные поля не заполнены, система выводит сообщение об ошибке, указывая, какие именно поля необходимо заполнить.
- Если произошла ошибка при загрузке файлов (постер или фильм), система уведомляет администратора и предлагает повторить загрузку.
- Если администратор передумал добавлять фильм, то нажимает на крестик и выходит из режима "Добавить фильм".

Результат сценария: Фильм успешно добавлен в базу данных и доступен пользователям на сайте.

2.12. Сценарий использования: Изменить фильм.

Цель: Администратор хочет изменить данные о существующем фильме.

Действующие лица: Администратор

Основной сценарий:

1. Администратор находится в админ-панели во вкладке "Редактирование каталога".
2. Вводит название фильма в поиск и нажимает кнопку "Найти".
3. Получает список фильмов с введенным словом в названии.
4. Нажимает на кнопку "Изменить".
5. Переходит на страницу изменения фильма.
6. Вводит необходимые изменения в соответствующие поля и нажимает кнопку "Применить".

7. Нажимает на крестик, чтобы вернуться на главную страницу админ-панели.

Альтернативные сценарии:

- Если администратор передумал изменять фильм, он нажимает на крестик для возвращения на главную страницу админ-панели.
- Ошибка при сохранении изменений:
    - Если данные заполнены некорректно, система выводит сообщение "Ошибка! Заполните все обязательные поля".

Результат сценария: Определенные поля фильма были изменены.

2.13. Сценарий использования: Просмотр фильма в проигрывателе.

Цель: Пользователь хочет посмотреть конкретный фильм.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку "Плэй".
2. Проигрыватель начинает показывать фильм.

Альтернативные сценарии:

- Пользователь хочет посмотреть фильм в полноэкранном режиме, тогда он нажимает на кнопку "Развернуть экран".
- Пользователь хочет поставить фильм на паузу – для этого он еще раз жмет на кнопку "Плэй".

Результат сценария: Пользователь просматривает фильм.

2.14. Сценарий использования: Настройки проигрывателя.

Цель: Пользователь хочет изменить параметры просмотра.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку "Настройки".
2. Выбирает необходимый параметр.
3. Меняет параметр на один из предложенных вариантов.

Альтернативные сценарии:

- Пользователь передумал менять настройки и закрыл меню.

Результат: Пользователь изменил параметры просмотра.

2.15. Сценарий использования: Просмотр фильма в полноэкранном режиме.

Цель: Пользователь хочет посмотреть фильм в полноэкранном режиме

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку “развернуть экран”
2. Пользователь нажимает на кнопку “плэй”

Альтернативные сценарии:

- Пользователь не хочет смотреть фильм в полноэкранном режиме, тогда он нажимает на кнопку “развернуть экран” еще раз.

Результат сценария: Пользователь смотрит фильм в полноэкранном режиме.

2.16. Сценарий использования: Настройки просмотра в полноэкранном режиме.

Цель: Пользователь хочет изменить параметры просмотра

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку “настройки”.

2. Пользователь выбирает необходимый параметр во всплывающем меню.
3. Пользователь меняет параметр один из предложенных вариантов.

Альтернативные сценарии:

- Пользователь передумал менять настройки и нажал на кнопку “настройки” еще раз, чтобы закрыть всплывающее меню.

Результат сценария: Пользователь изменил параметры просмотра в полноэкранном режиме

2.17. Сценарий использования: Настройка громкости проигрывателя.

Цель: Пользователь хочет изменить громкость проигрывателя

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку “звук”.
2. Пользователь двигает появившийся ползунок на нужное значение.

Альтернативные сценарии:

- Пользователь передумал менять настройки звука и нажал на кнопку “звук” еще раз, чтобы убрать ползунок.

Результат сценария: Пользователь изменил громкость проигрывателя.

2.18. Сценарий использования: Настройка громкости проигрывателя в полноэкранном режиме.

Цель: Пользователь хочет изменить громкость проигрывателя

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь нажимает на кнопку “звук”.
2. Пользователь двигает появившийся ползунок на нужное значение.

Альтернативные сценарии:

- Пользователь передумал менять настройки звука и нажал на кнопку “звук” еще раз, чтобы убрать ползунок.

Результат сценария: Пользователь изменил громкость проигрывателя в полноэкранном режиме.

2.19. Сценарий использования: Использование ползунка таймера.

Цель: Пользователь хочет перемотать фильм на нужную точку времени

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь наводится на точку на таймере.
2. Пользователь перетаскивает точку в нужное ему место, отслеживая время слева.

Результат сценария: Пользователь перемотал фильм на необходимую ему точку времени.

2.20. Сценарий использования: Использование ползунка таймера в полноэкранном режиме.

Цель: Пользователь хочет перемотать фильм на нужную точку времени

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь наводится на точку на таймере.
2. Пользователь перетаскивает точку в нужное ему место, отслеживая время слева.

Результат сценария: Пользователь перемотал фильм на необходимую ему точку времени в полноэкранном режиме.

2.21. Сценарий использования: Просмотр статистики.

Цель: Администратор хочет посмотреть статистику каталога фильмов.

Действующие лица: Администратор

Основной сценарий:

1. Администратор находится в админ-панели во вкладке "статистика".
2. Администратор нажимает на кнопку "фильтр".
3. Администратор вводит желаемые параметры для кастомизации статистики и нажимает применить.
4. Администратор видит диаграмму, отображающую кастомизированную статистику.

Альтернативные сценарии:

- Администратор передумал насчет всех параметров кастомизации и нажал на кнопку "очистить", чтобы заново ввести нужные параметры.

Результат сценария: Администратор получил кастомизированную статистику.

2.22. Сценарий использования: Просмотр актеров, участвующих в фильме.

Цель: Пользователь хочет узнать список актеров, участвующих в конкретном фильме.

Действующие лица: Пользователь

Основной сценарий:

1. Пользователь находится на странице фильма.
2. Пользователь нажимает на гиперссылку "и другие" и попадает на страницу со списком актеров в данном фильме.
3. Пользователь видит полный список актеров с их именем и фамилией, фотографией и ссылкой на более подробную информацию.

Альтернативные сценарии:

- Пользователь может поискать конкретного актера в фильме, для этого он перейдет на следующую страницу, нажав на цифру внизу.
- Пользователь нашел захотел вернуться обратно на страницу фильма и нажал на крестик.

Результат сценария: Пользователь узнал какие актеры участвовали в фильме.

2.23. Вывод

На основании описанных сценариев использования можно сделать вывод, что операции чтения будут преобладать над операциями записи в данном приложении. Пользователи смогут просматривать каталог фильмов и сериалов, искать и фильтровать контент, ознакомляться с информацией о фильмах и актерах, а также смотреть видео в проигрывателе. Администраторы могут просматривать и анализировать статистику, осуществлять мониторинг всего контента и проверять данные о персонах. Пользователю доступны следующие операции записи: выставление оценок фильмам. Администратор же имеет возможность добавлять и редактировать контент, а также управлять каталогом. Таким образом, основная функция приложения — обеспечение удобного доступа к медиаконтенту и его эффективное администрирование.

3. МОДЕЛЬ ДАННЫХ.

3.1. Нереляционная модель данных.

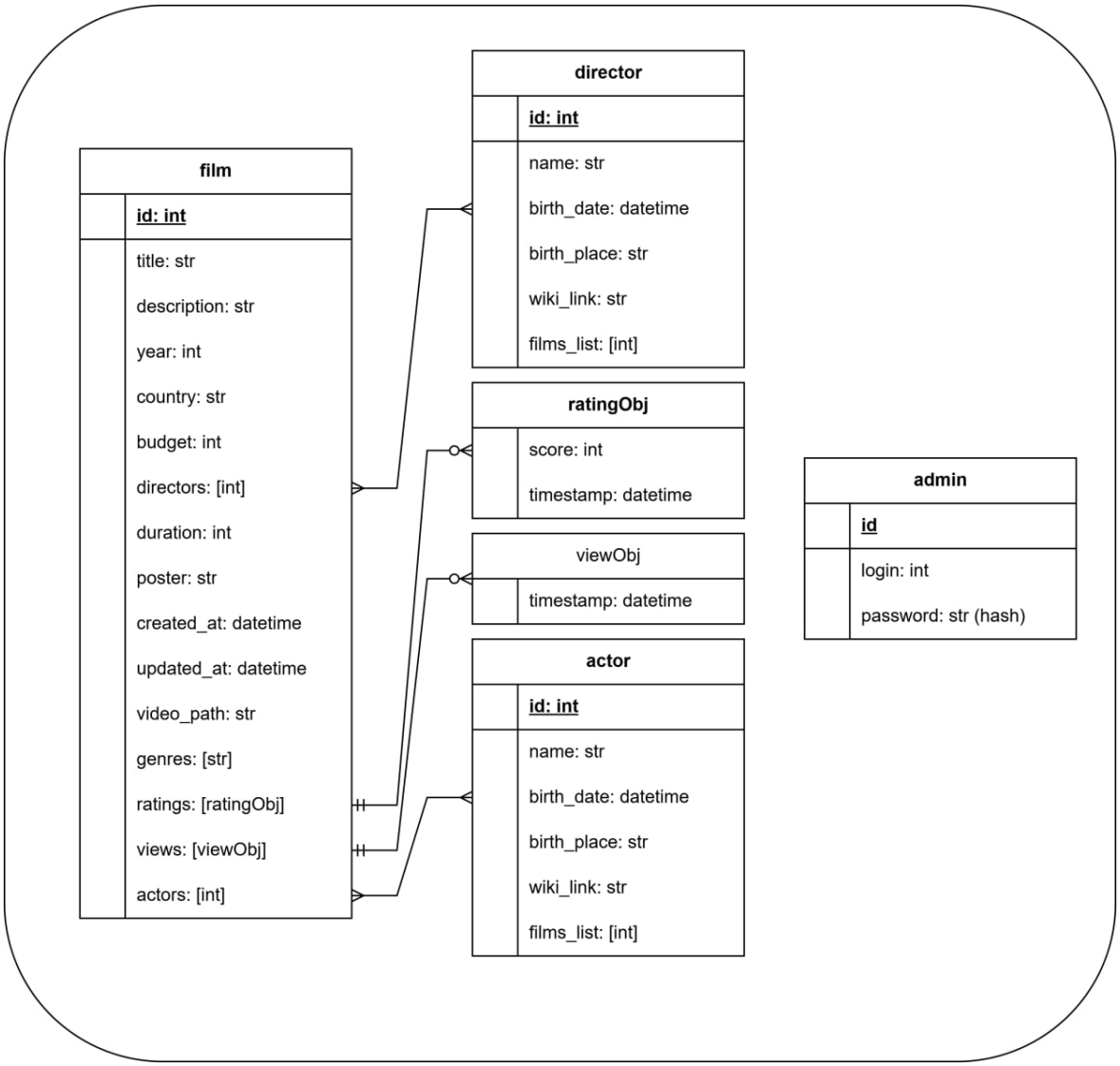

Рисунок 30. Графическое представление нереляционной модели.

1. Коллекция film

```
{

    "id": "number",
    "title": "string",
    "description": "string",
    "year": "number",
    "country": "string",
    "budget": "number",
    "directors": ["number"],
    "duration": "number",
    "poster": "string",
    "created_at": "Date (ISO)",
```

```
    "updated_at": "Date (ISO)",
    "video_path": "string",
    "genres": ["string"],
    "ratings": [ { "score": "number",
    "timestamp": "Date (ISO)" } ],
    "views": [ { "timestamp": "Date (ISO)" } ],
    "actors": ["number"]

}
```

2. Коллекция director

```
{

    "_id": "number",
    "name": "string",
    "birth_date": "Date (ISO)",
    "birth_place": "string",
    "wiki_link": "string",
    "films_list": ["number"]

}
```

3. Коллекция actor

```
{

    "_id": "number",
    "name": "string",
    "birth_date": "Date (ISO)",
    "birth_place": "string",
    "wiki_link": "string",
    "films_list": ["number"]

}
```

4. Коллекция admin

```
{

    "login": "string",
    "password": "string (хеш)"

}
```

Модель данных

1. Коллекция film

Поля и размеры:

- id: 4 байта
- title: 50 байт (50 символов)
- description: 1000 байт (1000 символов)
- year: 4 байта

- country: 50 байт (50 символов)
- budget: 8 байт (64-битное число)
- directors: 40 байт (10 элементов по 4 байта)
- duration: 4 байта
- poster: 50 байт (URL)
- created_at/updated_at: 24 байта (ISO timestamp)
- video_path: 50 байт (URL)
- genres: 150 байт (3 строки по 50 символов)
- ratings: 1200 байт (100 оценок × 12 байт)
- views: 8000 байт (1000 просмотров × 8 байт)
- actors: 40 байт (10 элементов по 4 байта)

Итого на фильм: ~10,614 байт (10.4 КБ)

2. Коллекция director
   Поля и размеры:
- _id: 4 байта name: 50 байт (50 символов)
- birth_date: 8 байт (timestamp)
- birth_place: 100 байт (100 символов)
- wiki_link: 50 байт (URL)
- films_list: 40 байт (10 элементов по 4 байта)

Итого на режиссера: 252 байта

3. Коллекция actor
   Поля и размеры:
- _id: 4 байта
- name: 50 байт (50 символов)
- birth_date: 8 байт (timestamp)
- birth_place: 100 байт (100 символов)
- wiki_link: 50 байт (URL)
- films_list: 40 байт (10 элементов по 4 байта)

Итого на актера: 252 байта

4. Коллекция admin
   Поля и размеры:
- login: 20 байт (20 символов)
- password: 32 байта (хеш SHA-256)

Итого на администратора: 52 байта

Оценка объема информации

Средний размер данных

- Коллекция film

4 (id) + 50 (title) + 1000 (description) + 4 (year) + 50 (country) + 8 (budget) + 8 (directors) + 4 (duration) + 50 (poster) + 16 (created/updated) + 50 (video_path) + 150 (genres) + 1200 (ratings) + 8000 (views) + 20 (actors) = 10,614 байт

- Коллекция director

4 (id) + 50 (name) + 8 (birth_date) + 50 (birth_place) + 50 (wiki) + 40 (films_list) = 202 байт

- Коллекция actor

4 (id) + 50 (name) + 8 (birth_date) + 50 (birth_place) + 50 (wiki) + 40 (films_list) = 202 байт

- Коллекция admin

20 (login) + 16 (password hash) = 36 байт

Расчет для N фильмов

Компонент	Формула	На 1 фильм	На 10,000 фильмов
Основные данные фильма	10,614 * N	10.6 КБ	106 МБ
Режиссеры (2 на фильм)	202 * 2 * N	0.4 КБ	4 МБ
Актеры (5 на фильм)	202 * 5 * N	1.0 КБ	10 МБ
Итого	(10,614 + 404 + 1010) * N	12.0 КБ	120 МБ

Избыточность данных

- Избыточные данные:
  - Жанры: 150 байт (3 строки)
  - ID актеров/режиссеров: 28 байт
- Чистый объем: (10,614–150 - 28) * N = 10,436 * N

- Коэффициент избыточности: 12,028 / 10,436 ≈ 1.15

Направление роста

Операция	Прирост данных
Добавление фильма	+12.0 КБ
Добавление оценки	+12 байт (score + timestamp)
1000 просмотров	+8 КБ
Новый актер/режиссер	+202 байта
Новый администратор	+36 байт

Пример роста для 1 млн фильмов

Компонент	Объем
Фильмы	10.6 ТБ
Актеры	1.0 ТБ
Режиссеры	0.4 ТБ
Итого	12 ТБ

Примеры данных

- Коллекция film


```
{
  "id": 1,
  "title": "Начало",
  "description": "Вор, крадущий корпоративные секреты через технологию общего сна, получает задание внедрить идею.",
  "year": 2010,
  "country": "США",
  "budget": 160000000,
  "directors": [1],
  "duration": 148,
  "poster": "https://storage.example.com/posters/inception.jpg",
  "created_at": "2010-07-16T00:00:00Z",
  "updated_at": "2020-01-01T00:00:00Z",
  "video_path": "https://storage.example.com/videos/inception.mp4",
  "genres": ["Фантастика", "Триллер"],
  "ratings": [
    {"score": 9, "timestamp": "2023-01-16T12:00:00Z"},
    {"score": 8, "timestamp": "2023-01-17T14:30:00Z"}
  ],
```

```
  "views": [
    {"timestamp": "2023-05-01T18:45:00Z"},
    {"timestamp": "2023-05-02T20:15:00Z"}
  ],
  "actors": [1, 2]
}
```

- Коллекция director

```
{
      "_id": 1,
      "name": "Кристофер Нолан",
      "birth_date": "1970-07-30T00:00:00Z",
      "birth_place": "Лондон, Великобритания",
      "wiki_link": "https://ru.wikipedia.org/wiki/Кристофер_Нолан",
      "films_list": [1, 2, 3]
}
```

- Коллекция actor

```
{ "_id": 1,
      "name": "Леонардо ДиКаприо",
      "birth_date": "1974-11-11T00:00:00Z",
      "birth_place": "Лос-Анджелес, США",
      "wiki_link": "https://ru.wikipedia.org/wiki/Леонардо_ДиКаприо",
      "films_list": [1, 4, 5]
}
```

- Коллекция admin

```
{
  "login": "admin",
  "password": "5f4dcc3b5aa765d61d8327deb882cf99"
}
```

Примеры запросов

Регистрация администратора

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
db.admins.insertOne({
  login: "admin",
  password: "5f4dcc3b5aa765d61d8327deb882cf99"
});
```

1. Сценарий: Главная/подборка (Популярное)

```
db.film.aggregate([
  { $sort: { views: -1 } },
  { $limit: 20 },
  { $lookup: {
      from: "director",
      localField: "directors",
      foreignField: "id",
      as: "director_info"
  }},
  { $project: {
      title: 1,
      poster: 1,
      duration: 1,
      "director_info.name": 1,
      views: 1
  }}
]);
```

2. Сценарий: Поиск фильма

```
db.film.createIndex({ title: "text" }); // Создать индекс один раз
db.film.find({
  $text: { $search: "матрица" }
}, { score: { $meta: "textScore" } })
.sort({ score: { $meta: "textScore" } });
```

3. Сценарий: Фильтры (по жанру и длительности)

```
db.film.find({ genres: "Фантастика", duration: { $gte: 90, $lte:
180 } });
```

4. Сценарий: Сортировка фильмов

- Сортировка по просмотрам (от большего к меньшему)

```
db.film.find().sort({ views: -1 });
```

- Сортировка по дате добавления (новые первыми)

```
db.film.find().sort({ created_at: -1 });
```

5. Сценарий: Получение рекомендаций

```
db.film.aggregate([

    { $match: { genres: { $in: ["Драма", "Триллер"] } },

    { $sample: { size: 10 } }

]);
```

6. Сценарий: Страница фильма

```
db.film.aggregate([
  { $match: { id: 123 } },
  { $lookup: {
      from: "director",
      localField: "directors",
      foreignField: "id",
      as: "directors_info"
  }},
  { $lookup: {
      from: "actor",
      localField: "actors",
      foreignField: "id",
      as: "actors_info"
  }}
]);
```

7. Сценарий: Добавление оценки

– Добавить оценку в массив

```
db.film.updateOne(
  { id: 123 },
  { $push: { ratings: 8 } }
);
```

– Обновить средний рейтинг

```
db.film.updateOne(
  { id: 123 },
  [{
    $set: {
```

```
      avg_rating: { $avg: "$ratings" }
    }
  }]
);
```

8. Сценарий: Вход в админ-панель

```
db.admin.findOne({
  login: "admin@example.com",
  password: "хеш_пароля"
});
```

9. Сценарий: Импорт базы данных

– Для каждой коллекции (пример для films)

```
db.film.insertMany([
  { id: 1, title: "Фильм 1", ... },
  { id: 2, title: "Фильм 2", ... }
]);
```

10. Сценарий: Экспорт базы данных

```
db.film.find().toArray();
db.director.find().toArray();
db.actor.find().toArray();
```

11. Сценарий: Редактирование каталога

– Поиск фильма для редактирования

```
db.film.find({ title: { $regex: "Начало", $options: "i" } });
```

– Обновление данных

```
db.film.updateOne(
  { id: 1 },
  { $set: { title: "Новое название", duration: 150 } }
);
```

12. Сценарий: Удаление фильма

```
db.film.deleteOne({ id: 1 });
```

13. Сценарий: Добавление фильма

```
db.film.insertOne({
  id: 999,
  title: "Новый фильм",
  directors: [1, 2],
  actors: [5, 6],
  genres: ["Боевик"],
  duration: 120,
  ratings: [],
  views: 0
});
```

14. Сценарий: Просмотр статистик

```
// Статистика по жанрам
db.film.aggregate([
  { $unwind: "$genres" },
  { $group: {
      _id: "$genres",
      total: { $sum: 1 },
      avg_views: { $avg: "$views" }
  }}
]);
```

15. Сценарий: Просмотр актеров фильма

```
db.actor.find({
  id: { $in: [1, 2, 3] } // IDs из поля actors фильма
});
```

16. Сценарий: Пагинация

```
db.film.find()
  .skip(20)
  .limit(10)
  .sort({ created_at: -1 });
```

3.2. Реляционная модель.

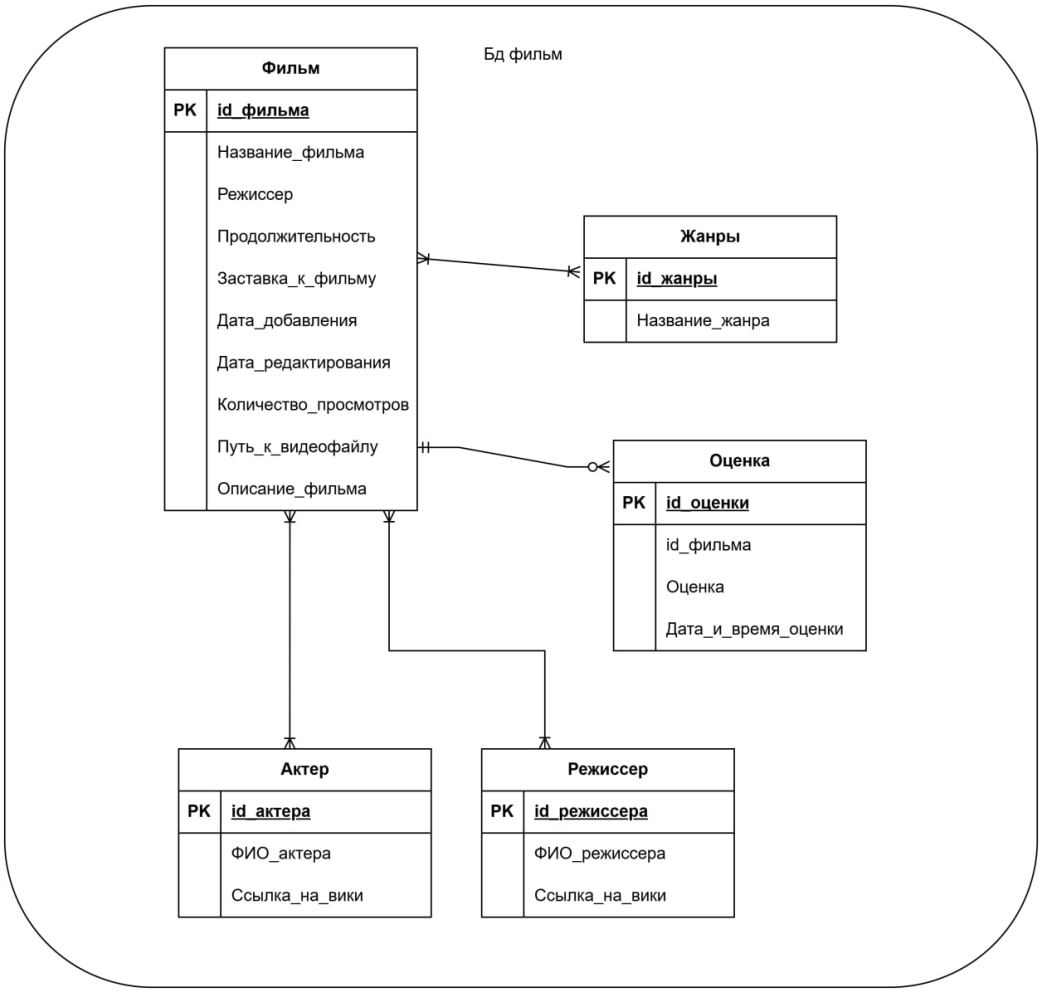

Рисунок 31. Реляционная модель данных.

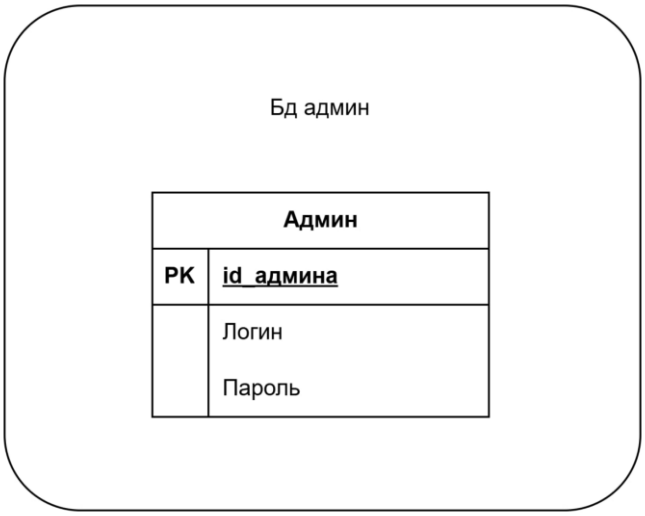

Рисунок 32. База данных для хранения администраторов.

Таблицы и их структура

1. Фильмы film

- id: INTEGER (~4 байта) (PRIMARY KEY) — Уникальный идентификатор фильма
- title: TEXT (~50 байт) — Название фильма
- director_id: INTEGER (~4 байта) (FOREIGN KEY) — Ссылка на режиссера
- duration: INTEGER (~4 байта) — Длительность в минутах
- poster_url: TEXT (~50 байт) — Путь к изображению постера
- created_at: TIMESTAMP (~8 байт) — Дата и время добавления
- updated_at: TIMESTAMP (~8 байт) — Дата и время последнего обновления
- views_count: INTEGER (~4 байта) — Количество просмотров
- video_path: TEXT (~50 байт) — Путь к видеофайлу
- description: TEXT (~1000 байт) — Описание фильма

2. Актеры actor

- id: INTEGER (~4 байта) (PRIMARY KEY) — Уникальный идентификатор
- full_name: TEXT (~50 байт) — Полное имя
- wiki_link: TEXT (~50 байт) — Ссылка на страницу в Википедии

3. Режиссеры director

- id: INTEGER (~4 байта) (PRIMARY KEY) — Уникальный идентификатор

- full_name: TEXT (~50 байт) — Полное имя
- wiki_link: TEXT (~50 байт) — Ссылка на страницу в Википедии

4. Жанры genres

- id: INTEGER (~4 байта) (PRIMARY KEY) — Уникальный идентификатор жанра
- name: TEXT (~50 байт) — Название жанра

5. Оценки ratings

- id: INTEGER (~4 байта) (PRIMARY KEY) — Уникальный идентификатор оценки
- film_id: INTEGER (~4 байта) (FOREIGN KEY) — Ссылка на фильм
- score: INTEGER (~4 байта) — Значение оценки (1-10)
- created_at: TIMESTAMP (~8 байта) — Дата и время оценки

6. Админы admins

- id: INTEGER (~4 байта) (PRIMARY KEY) — Уникальный идентификатор
- login: TEXT (~20 байт) — Логин пользователя
- password_hash: TEXT (~16 байт) — Хеш пароля

Связующие таблицы (отношения многие-ко-многим)

Фильмы-Жанры *film_genre*

- film_id: INTEGER (4 байта) (FOREIGN KEY)
- genre_id: INTEGER (4 байта) (FOREIGN KEY)
- PRIMARY KEY (film_id, genre_id)

Фильмы-Актеры *film_actor*

- film_id: INTEGER (4 байта) (FOREIGN KEY)
- actor_id: INTEGER (4 байта) (FOREIGN KEY)
- PRIMARY KEY (film_id, actor_id)

Фильмы-Режиссеры *film_director*

- film_id: INTEGER (4 байта) (FOREIGN KEY)
- director_id: INTEGER (4 байта) (FOREIGN KEY)
- PRIMARY KEY (film_id, director_id)

Оценка объема информации

Средний размер информации, хранимой в модели:

- Films: 4 + 50 + 4 + 4 + 50 + 8 + 8 + 4 + 50 + 1000 = 1182 байта
- Actor: 4 + 50 + 50 = 104 байта
- Director: 4 + 50 + 50 = 104 байта
- Genres: 4 + 50 = 54 байта
- Ratings: 4 + 4 + 4 + 8 = 20 байт
- Admins: 4 + 20 + 16 = 40 байт

Связующие таблицы:

- Film_Genres: 4 + 4 = 8 байт
- Film_Actors: 4 + 4 = 8 байт
- Film_Directors: 4 + 4 = 8 байт

В среднем фильм содержит:

- 3 жанра
- 5 актеров
- 50 оценок
- 2 режиссёра

При количестве фильмов, равном films:

V(films) = (1182 + 3 * 8 + 5 * 8 + 50 * 20 + 2 * 8) * films = 2262 * films байт

Избыточность данных

В таблице Films избыточными являются:

- poster_url (50 байт)
- video_path (50 байт)

Итоговое уменьшение:

Films: 1182 - 100 = 1082 байта

Новая формула для чистого объема данных:

V_clear(films) = (1082 + 3 * 8 + 5 * 8 + 50 * 20) * films = 2092 * films байт

Коэффициент избыточности:

$$\frac{V(films)}{V_{clear}(films)} = \frac{2192 * films}{2092 * films} = 1.048$$

Направление роста модели

При добавлении нового фильма увеличиваются следующие таблицы:

- Films (+1182 байта)
- Film_Genres (+24 байта
- Film_Actors (+40 байт)
- Ratings (+1000 байт)

При добавлении нового жанра, актера или режиссера увеличивается соответствующая таблица на 54–104 байта. При добавлении оценки увеличивается таблица Ratings на 20 байт. При добавлении администратора увеличивается таблица Admins на 40 байт.

Примеры данных

Таблица admin

id	login	password_hash
1	admin	5f4dcc3b5aa765d61d8
2	moderator	6c569aabbf7775ef8e1
3	supervisor	098f6bcd4621d373cade

Таблица film

id	1	2
Название	Начало	Побег из Шоушенка
Длительность	148	142
Путь к постеру	https://storage.example.com/posters/inception.jpg	https://storage.example.com/posters/shawshank.jpg
Дата создания	2023-01-15 10:00:00	2023-01-16 11:30:00
Дата обновления	2023-01-15 10:00:00	2023-02-20 09:15:00
Просмотры	1500000	2500000
Путь к видео	https://stream.example.com/videos/inception.mp4	https://stream.example.com/videos/shawshank.mp4
Описание	Вор, который крадет корпоративные...	Два заключенных находят общий язык...

Таблица actor

id	Полное имя	Ссылка на Википедию
1	Леонардо ДиКаприо	https://ru.wikipedia.org/...
2	Том Харди	https://ru.wikipedia.org/...
3	Тим Роббинс	https://ru.wikipedia.org/...

Таблица director

id	Полное имя	Ссылка на Википедию
1	Кристофер Нолан	https://ru.wikipedia.org/...
2	Фрэнк Дарабонт	https://ru.wikipedia.org/...

Таблица genre

id	Название
1	Фантастика
2	Триллер
3	Драма
4	Криминал

Таблица rating

id	id_фильма	Оценка	Дата оценки
1	1	9	2023-01-16 12:00:00
2	1	8	2023-01-17 14:30:00
3	2	10	2023-01-18 09:15:00

Таблица film_genre

id_фильма	id_жанра
1	1
1	2
2	3
2	4

Таблица film_actor

id_фильма	id_актера
1	1
1	2
2	3

Таблица film_director

id_фильма	id_режиссера
1	1
2	2

Примеры запросов

Получение популярных фильмов (сортировка по количеству просмотров)

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
SELECT id, title, poster_url, views_count

FROM film

ORDER BY views_count

DESC LIMIT 20;
```

Получение следующей страницы

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
SELECT id, title, poster_url

FROM film

ORDER BY created_at DESC

LIMIT 20 OFFSET 20;
```

Поиск по названию

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
SELECT id, title, poster_url, duration

FROM film

WHERE title LIKE '%Начало%'

ORDER BY title;
```

Фильтрация по жанру и году

- Количество запросов: 1
- Количество задействованных коллекций: 2

```
SELECT f.id, f.title, f.poster_url

FROM film f

JOIN film_genres fg ON f.id = fg.film_id
```

```
WHERE fg.genre_id = 1

AND strftime('%Y', f.created_at) = '2023'

ORDER BY f.views_count DESC;
```

Сортировка по рейтингу

- Количество запросов: 1
- Количество задействованных коллекций: 2

```
SELECT f.id, f.title, AVG(r.score) as avg_rating

FROM film f

LEFT JOIN ratings r ON f.id = r.film_id
```

Получение полной информации о фильме

- Количество запросов: 1
- Количество задействованных коллекций: 2

```
GROUP BY f.id

ORDER BY avg_rating DESC;

SELECT f.*, d.full_name as director

FROM film f

JOIN director d ON f.director_id = d.id

WHERE f.id = 1;
```

Получение актеров фильма

- Количество запросов: 1
- Количество задействованных коллекций: 2

```
SELECT a.id, a.full_name

FROM actor a

JOIN film_actors fa ON a.id = fa.actor_id

WHERE fa.film_id = 1;
```

Добавление оценки

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
INSERT INTO ratings (film_id, score, created_at)

VALUES (1, 9, datetime('now'));
```

Получение рекомендаций по жанрам

- Количество запросов: 1
- Количество задействованных коллекций: 2

```
SELECT f.id, f.title

FROM film f

JOIN film_genres fg ON f.id = fg.film_id

WHERE fg.genre_id IN (1, 2)

ORDER BY f.views_count DESC

LIMIT 10;
```

Авторизация администратора

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
SELECT id FROM admins

WHERE login = 'admin' AND password_hash = '5f4dcc3b5aa765d61d8';
```

Добавление нового фильма

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
INSERT INTO film (title, director_id, duration, poster_url,
created_at, updated_at, views_count, video_path, description) VALUES
('Новый фильм', 1, 120,
'https://storage.example.com/posters/new_film.jpg', datetime('now'),
datetime('now'), 0,
'https://storage.example.com/videos/new_film.mp4', 'Описание нового
фильма');
```

Редактирование фильма

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
UPDATE film
```

```
SET title = 'Обновленное название', updated_at = datetime('now')
WHERE id = 1;
```

Удаление фильма

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
DELETE FROM film WHERE id = 1;
```

Получение статистики по просмотрам

- Количество запросов: 1
- Количество задействованных коллекций: 1

```
SELECT strftime('%Y-%m', created_at) as month, COUNT(*) as views
FROM film
GROUP BY month
ORDER BY month;
```

Примечание

Количество запросов и задействованных коллекций может изменяться в зависимости от сложности запроса, которая может изменяться (например, запрос на фильтрацию может происходить по актерам, режиссерам, жанрам и т. д.)

3.3. Сравнение моделей

Удельный объем информации

Удельный объем информации у NoSQL модели меньше: на 1 фильм она занимает ~42.7 КБ, тогда как в реляционной — ~53.2 КБ. Это происходит благодаря отсутствию связей и включению всех связанных данных (актеры, режиссёры и пр.) в один документ. NoSQL модель выигрывает по компактности хранения при небольших потерях нормализации.

Запросы по отдельным юзкейсам

В SQL-модели для выполнения типовых юзкейсов (например, получения информации о фильме с актёрами, режиссёрами и жанрами)

требуется больше отдельных запросов, так как данные распределены по разным таблицам и связаны между собой внешними ключами. В среднем на один юзкейс уходит 2–4 запроса с объединениями (JOIN).

В NoSQL-модели всё содержимое фильма достаточно одного запроса и одной коллекции.

Вывод

NoSQL-модель более эффективна с точки зрения количества запросов и задействованных коллекций: все данные о фильме (включая актёров, режиссёров, жанры и рейтинги) хранятся в одном документе. Это позволяет обойтись одним запросом и одной коллекцией для большинства юзкейсов. В SQL-модели данные распределены по таблицам, что требует нескольких запросов с объединениями и обращения к 4–6 таблицам. При активном чтении и небольшом объеме обновлений NoSQL оказывается практичнее.

4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

4.1. Краткое описание приложения.

Frontend реализован с использованием HTML, CSS и JavaScript с применением библиотеки Tagify для работы с тегами. Он осуществляет взаимодействие с backend через REST API, поддерживающее GET, POST, PUT, DELETE и OPTIONS запросы, обмениваясь данными в формате JSON. На frontend полученные данные отображаются в удобном для пользователя виде — в форме карточек фильмов, детализированных страниц контента и списков с информацией об актерах и режиссерах.

Backend реализован на языке Python с использованием фреймворка Flask. Взаимодействие с frontend происходит через API, которое обрабатывает GET запросы для получения информации о фильмах, сериалах и персонах, POST запросы для добавления нового контента, PUT запросы для обновления существующих данных, DELETE запросы для удаления записей и OPTIONS запросы для обработки CORS.

Для упрощения развертывания и обеспечения изоляции окружения все компоненты приложения упакованы в контейнеры с использованием Docker. Это решение позволяет легко развертывать приложение в различных средах и гарантировать совместимость между всеми компонентами системы, включая базу данных MongoDB для хранения структурированных данных о фильмах, сериалах и персонах.

4.2. Использованные технологии.

Frontend: HTML/CSS/JavaScript.

Backend: Flask (Python).

DB: MongoDB.

4.3. Снимки экрана приложения.

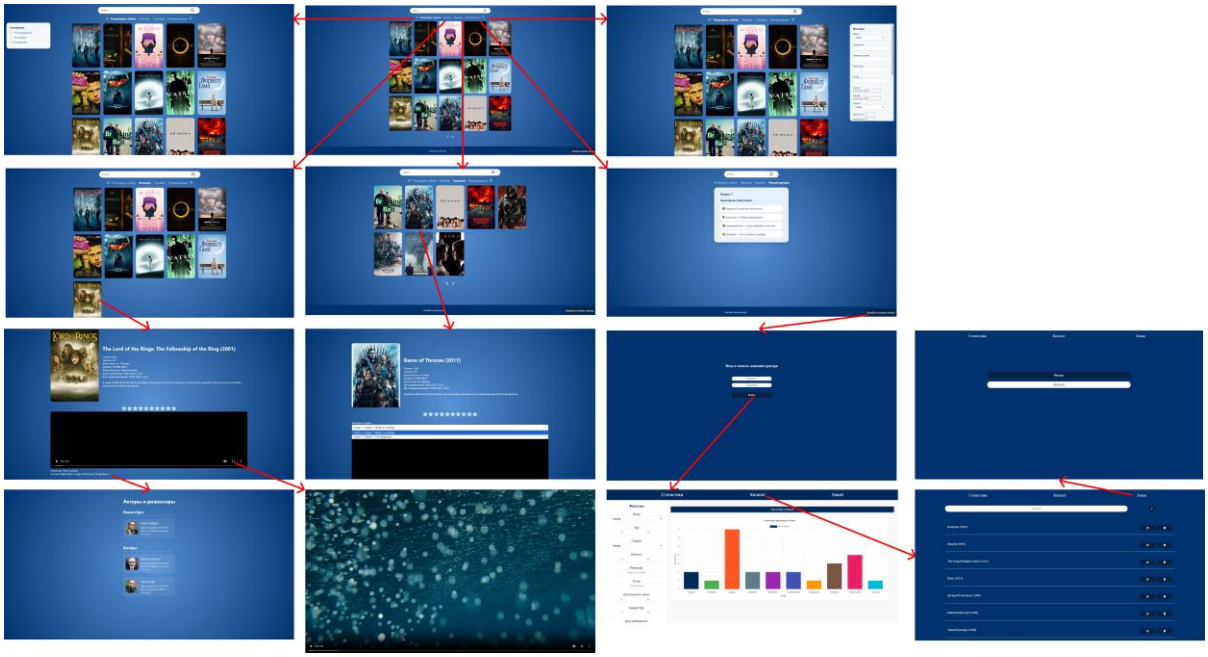

Рисунок 33. Схема экранов приложения

5. ВЫВОДЫ

5.1. Достигнутые результаты.

По итогам выполнения работы реализовано веб-приложение онлайн-кинотеатра. Пользователи могут просматривать каталог фильмов и сериалов, переключаясь между разделами "Популярно сейчас", "Фильмы", "Сериалы" и персонализированными "Рекомендациями". Для удобства навигации реализована система фильтрации и сортировки контента по различным параметрам (жанр, год выпуска, рейтинг и др.).

При выборе конкретного фильма или сериала пользователь получает доступ к детальной странице с полной информацией (описание, длительность, список актеров и режиссеров с данными из Wikidata и др.), возможностью просмотра и выставления оценки по 10-звездочной шкале. Система рекомендаций формирует персональные подборки на основе предпочтений, выявленных через интерактивный опрос.

Административная панель предоставляет расширенные возможности управления контентом:

- Просмотр и анализ статистики просмотров и рейтингов
- Полноценное редактирование каталога (добавление/изменение/удаление фильмов и сериалов)
- Система резервного копирования с импортом/экспортом данных в JSON-формате

Приложение обеспечивает стабильную работу при высокой нагрузке за счет оптимизированной архитектуры и использования современных технологий (Flask, MongoDB, Docker). Реализована интеграция с внешними API (Wikidata) для автоматического дополнения информации о персонах.

Таким образом, онлайн-кинотеатр представляет собой полнофункциональную платформу для комфортного просмотра медиаконтента с развитыми возможностями администрирования и персонализации.

5.2. Недостатки и пути для улучшения полученного решения.

На текущий момент приложение имеет ряд ограничений, устранение которых позволит значительно улучшить пользовательский опыт и расширить функциональность:

1. Отсутствие персонализации без опроса

Система рекомендаций работает исключительно на основе явного опроса. Добавление алгоритмов анализа поведения (просмотренные фильмы, оценки) позволит предлагать контент автоматически.

2. Ограниченные возможности социального взаимодействия

В приложении отсутствуют функции комментирования и создания списков для обмена с другими пользователями. Реализация этих возможностей повысит вовлеченность аудитории.

3. Отсутствие мобильной версии

Текущая веб-версия не оптимизирована для мобильных устройств. Разработка адаптивного дизайна или отдельных приложений для iOS/Android улучшит пользовательский опыт.

4. Базовые функции администрирования

Админ-панель требует ручного управления всеми процессами. Автоматизация некоторых задач (например, обновление информации из Wikidata) сократит нагрузку на администраторов.

5. Отсутствие системы уведомлений

Пользователи не получают оповещений о новых сериях или рекомендованных фильмах. Реализация email-рассылок и push-уведомлений повысит активность пользователей.

6. Ограниченная поддержка пользователей

В приложении отсутствует система обратной связи и помощи. Добавление чата поддержки и FAQ-раздела решит эту проблему.

5.3. Будущее развитие решения.

В рамках дальнейшего развития онлайн-кинотеатра планируется внедрение следующего функционала:

1. Расширение персонализации контента

Система будет анализировать не только явные предпочтения через опросы, но и поведенческие данные (просмотренные фильмы, оценки, продолжительность просмотра) для автоматического формирования рекомендаций с использованием алгоритмов машинного обучения.

2. Мультиязычная поддержка

В приложение будет добавлена возможность выбора языка интерфейса (английский, испанский, китайский), что расширит целевую аудиторию на международные рынки.

3. Мобильные приложения

Планируется разработка нативных приложений для iOS и Android с функциями:

- Оффлайн-просмотра скачанного контента
- Push-уведомлений о новых сериях

4. Социальные функции

Будут реализованы:

- Система комментирования и рецензий
- Возможность создания персональных коллекций
- Совместный просмотр с синхронизацией воспроизведения

5. Автоматизация администрирования

Внедрение инструментов для:

- Автоматического обновления метаданных из Wikidata
- Мониторинга доступности видеоресурсов
- Пакетной обработки контента

Эти улучшения позволят трансформировать онлайн-кинотеатр в полноценную мультиплатформенную экосистему для просмотра и обсуждения кино- и сериального контента с персонализированным подходом к каждому пользователю.

6. ПРИЛОЖЕНИЯ

6.1. Документация по сборке и развертыванию приложения.

1. Склонировать репозиторий с проектом [1] по ссылке: https://github.com/moevm/nosql1h25-cinema.
2. Перейти в корневую директорию проекта.
3. Собрать приложение с помощью команды: docker-compose build --no-cache.
4. Запустить приложение с помощью команды: docker-compose up.
5. Открыть приложение в браузере по адресу http://127.0.0.1:5000 или нажать на порт контейнера frontend в приложении Docker Desktop.

6.2. Инструкция для пользователя.

Навигация по каталогу

На главной странице доступны вкладки:

- "Популярно сейчас" – рекомендованные фильмы
- "Фильмы" – полный каталог фильмов
- "Сериалы" – список всех сериалов
- "Рекомендации" – персонализированные подборки

Сортировка

В каталогах фильмов и сериалов доступна сортировка по:

- Популярности (по количеству просмотров)
- Новизне (по дате выхода)
- Рейтингу (по оценкам пользователей)

Поиск и фильтрация

Для поиска контента:

- Нажмите значок фильтра (иконка воронки) в верхней панели
- Откроется боковая панель "Фильтры" со всеми доступными параметрами

- Заполните нужные параметры:
    - Жанр: выберите из выпадающего списка
    - Описание: введите ключевые слова
    - Название серии: введите часть названия для поиска конкретных эпизодов
    - Режиссер: введите имя или фамилию
    - Актер: введите имя или фамилию
    - Год выпуска: укажите диапазон (от/до)
    - Страна: выберите из выпадающего списка
    - Рейтинг: укажите минимальную и максимальную оценку (1–10)
    - Длительность: задайте диапазон в минутах
    - Бюджет: укажите диапазон в долларах
    - Даты добавления/редактирования: выберите период
- Нажмите "Применить" для использования выбранных параметров
- Нажмите "Очистить" для сброса всех фильтров
- Для закрытия панели нажмите ещё раз на значок фильтра

Просмотр контента

1. На странице фильма/сериала отображается:
- Постер и основная информация (название, год)
- Детали: страна, оценка, длительность, бюджет, жанр, даты добавления/редактирования
- Видеоплеер с функциями:
    - Полноэкранный режим
    - Регулировка громкости
    - Изменение скорости воспроизведения
- Список актеров и режиссеров с кнопкой "Подробнее"
2. Для просмотра информации о персонах:
- Нажмите "Подробнее" в соответствующем разделе

- Откроется страница с:
    - Фотографией
    - Данными о персоне
    - Ссылкой на Wikipedia

3. Оценка контента:

- Выберите количество звезд (1–10)
- Рейтинг автоматически сохраняется

Персонализированные рекомендации

1. Перейдите во вкладку "Рекомендации"
2. Пройдите короткий опрос (5 вопросов)
3. Система сформирует подборку на основе ваших предпочтений

Административная панель

1. Авторизация:

- Нажмите "Перейти в админ-панель" на главной
- Введите email и пароль администратора

2. Управление контентом:

- Вкладка "Каталог":
    - Добавление: нажмите "+", заполните все поля формы
    - Редактирование: выберите фильм, нажмите иконку карандаша
    - Удаление: выберите фильм, нажмите иконку корзины
- Для сериалов дополнительно укажите:
    - Номер сезона и серии
    - Название серии
    - Ссылку на видео

3. Статистика:

- Во вкладке "Статистика" выберите:
    1. Настройки графика
        - Ось X: жанр / год / страна
        - Ось Y: количество / средний рейтинг

- Тип диаграммы: Столбчатая / Линейная / Круговая

После выбора параметров нажмите "Обновить график", чтобы построить визуализацию.

2. Фильтрация данных

- Можно применить фильтры для более детального анализа:
  - Жанр
  - Год выпуска (от и до)
  - Страна
  - Рейтинг (от и до)
  - Режиссер
  - Актер
  - Длительность (от и до, мин)
  - Бюджет (от и до, $)
  - Дата добавления (от и до)
  - Дата редактирования (от и до)
- Действия:
  - "Применить" – обновляет статистику с учетом выбранных фильтров.
  - "Сбросить" – отображает данные по всем фильмам/сериалам из базы.

4. Работа с бэкапами:

- "Экспорт": сохраняет текущую базу в JSON-файл
- "Импорт": загружает данные из JSON-файла

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ссылка на код проекта на GitHub. – [Электронный ресурс]. – URL: https://github.com/moevm/nosql1h25-cinema
2. Документация Python. – [Электронный ресурс]. – URL: https://docs.python.org/3/ (дата обращения 20.05.2025).
3. Документация JavaScript. – [Электронный ресурс]. – URL: https://learn.javascript.ru/ (дата обращения 20.05.2025).
4. Документация MongoDB. – [Электронный ресурс]. – URL: https://www.mongodb.com/docs/ (дата обращения 20.05.2025).
5. Документация pymongo. – [Электронный ресурс]. – URL: https://pymongo.readthedocs.io/en/stable/ (дата обращения 20.05.2025).
6. Документация MongoEngine. – [Электронный ресурс]. – URL: https://mongoengine.readthedocs.io/en/latest/ (дата обращения: 20.05.2025).
7. Документация Flask. – [Электронный ресурс]. – URL: https://flask.palletsprojects.com/en/latest/ (дата обращения 22.05.2025).
8. Документация Tagify (для работы с тегами). – [Электронный ресурс]. – URL: https://github.com/yairEO/tagify (дата обращения: 20.05.2025).
9. Wikidata API документация. – [Электронный ресурс]. – URL: https://www.wikidata.org/wiki/Wikidata:Data_access (дата обращения: 20.05.2025).
10. Документация Docker. – [Электронный ресурс]. – URL: https://docs.docker.com/ (дата обращения: 20.05.2025).