Федеральное государственное автономное
образовательное учреждение
Высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
«_____» _____ 2022 г.

БАКАЛАВАРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование специальности

Сервис совместного просмотра видеоконтента
тема

Руководитель	_____	Старший преподаватель	О.В. Шмелев
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		П.А. Кононов
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____	Старший преподаватель	О.В. Шмелев
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2022

Министерство науки и высшего образования РФ
Федеральное государственное автономное образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы,фамилия
«_____» _____ 2022 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Красноярск 2022

Студенту Кононову Павлу Александровичу
фамилия, имя, отчество

Группа КИ18-06Б направление (специальность) 09.03.01
номер код

Информатика и вычислительная техника
наименование

Тема выпускной квалификационной работы Сервис совместного просмотра видеоконтента

Утверждена приказом по университету № _____ от _____

Руководитель ВКР О.В. Шмелев, старший преподаватель
инициалы, фамилия, должность, ученое звание и место работы

Исходные данные для ВКР: сформулировать цели и задачи, провести анализ существующих аналогов в предметной области, реализовать сервис совместного просмотра видеоконтента.

Перечень разделов ВКР: анализ задания на выполнения и обзор аналогов, проектирование сервиса совместного просмотра видеоконтента, программная реализация.

Перечень графического материала: презентация в формате PowerPoint, видеоролик с демонстрацией работы сервиса.

Руководитель _____ О. В. Шмелев
Подпись, дата Инициалы и фамилия

Задание принял к исполнению П. А. Кононов
Подпись, инициалы и фамилия студента

«____» _____ 2021 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Сервис совместного просмотра видеоконтента» содержит 41 страниц, исследовано 5 источников, использовано 25 рисунков.

СЕРВИС, КЛИЕНТ, СЕРВЕР, ФРЕЙМВОРК, VUE.JS, LARAVEL, REDIS, POSTGRESQL, WEBSOCKET, HTTP.

Цель работы: разработать сервис совместного просмотра видеоконтента.

При выполнении данной работы был произведен обзор предметной области, задания на выпускную квалификационную работу, изучены существующие аналоги и сформированы требования, предъявляемые к сервису.

Задачи:

- осуществить выбор программных средств моделирования и разработки сервиса;

- выполнить моделирование разрабатываемого сервиса;

- выполнить программную реализацию сервиса для совместного просмотра видеоконтента;

- проанализировать полученные результаты работы.

В результате работы над выпускной квалификационной работой был разработан и реализован сервис совместного просмотра видеоконтента.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время потребление медиаконтента тесно связано с повседневной жизнью. Существует большое количество сервисов, для прослушивания музыки и подкастов, чтения книг, просмотра кино и сериалов. Но в основном все эти сервисы выполняют свою функцию для одного пользователя. И если для совместного прослушивания сервисы могут предлагать настроить свою радиоволну, то сервисы, предоставляющие видеоконтент, зачастую не предлагают подобного функционала. Сервис, который я хочу создать, будет предлагать пользователю весь функционал для совместного просмотра видеоконтента, а также поддерживать функции онлайн переписки и онлайн звонков.

Целью данной работы является разработать сервис для совместного просмотра видеоконтента с функциями загрузки своих видео, создания приватных и общих комнат для совместного просмотра, онлайн чата, онлайн звонками, а также одиночного просмотра.

1. Анализ задания для выполнения

Необходимо разработать сервис для совместного просмотра видеоконтента, реализующее следующий функционал:

- авторизация;
- личный кабинет;
- загрузка видео;
- одиночный просмотр видео;
- совместный просмотр видео;
- онлайн переписка;
- онлайн звонки;
- модерирование загруженных видео;
- возможность оставлять отзывы на видео.

Данное приложение будет основано на модели клиент-серверного взаимодействия (Рис. 1)

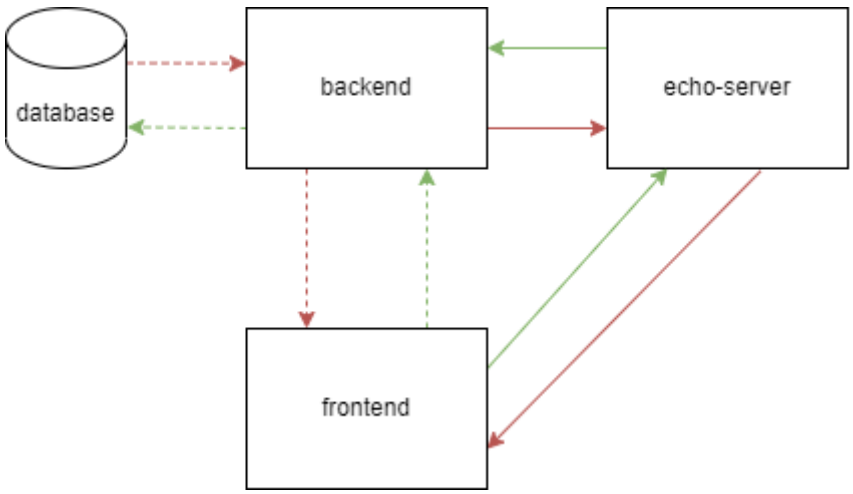

Рисунок 1 – Клиент-серверное взаимодействие

Где помимо стандартных частей клиент-серверного приложения, есть прослойка в виде “echo-server”, которая позволяет back-end и front-end общаться посредством протокола websocket [1].

1.1 Анализ технологий

1.1.1 SSR веб-приложения

Серверный рендеринг – это рендеринг на сервере клиентской части или универсального приложения HTML [1].

При серверном рендеринге на запрос клиента сервер генерирует HTML-код страницы. Такой подход позволяет добиться быстрой первой отрисовки, так же добиться большей производительности на стороне клиента, так как выполнение всех скриптов производит сервер.

При использовании SSR пользователь не будет ждать, пока отработают все скрипты, прежде чем начнут использовать сайт. Однако о подобного подхода к рендерингу есть недостаток: высокая нагрузка на сервер.

1.1.2 CSR веб-приложения

Рендеринг на стороне клиента подразумевает рендеринг страниц прямо в браузере с помощью javascript. Вся логика, получение данных, шаблонизация и маршрутизация обрабатывается на клиенте, а не на сервере [1]

У такого подхода главным недостатком является то, что на слабых устройствах веб-приложение будет работать медленно.

При клиентском рендеринге сервер будет выступать в роли API.

1.1.3 Реактивность веб-приложения

Реактивное программирование – парадигма программирования, ориентированная на потоки данных и распространении изменений [2].

Проще говоря, при изменении значения переменной она автоматически обновляется в визуальном отображении и наоборот.

Такой подход позволит избежать лишних перезагрузок веб-страницы, а так же визуально украсить все действия пользователя.

Данный подход используется во многих современных веб-приложениях.

1.2 Обзор существующих решений

На данный момент уже существуют сервисы, обеспечивающие совместный просмотр видео, поддерживающие аудио, видео и текстовые чаты.

Однако, у предоставленных решений есть ряд минусов. Это касается функционала - если сервис бесплатен, то он не реализует весь функционал, который я хочу предоставить пользователю, если сервис платный, то он предоставляет подобный функционал, за исключением некоторых пунктов. В таблице 1 предоставлен сравнительный разбор решений, которые сейчас существуют

Таблица 1 – сравнительный разбор предоставленных вариантов

Название	Дополнительные зависимости	Стоимость использования	Возможность загрузки видео	Поддержка русского языка	Онлайн звонки	Онлайн чат
Okko Party	Требует расширение для браузера	199 руб.	-	+	+	+
Яндекс видео	-	Частично бесплатен	-	+	-	+
Watch2Gether	Требует расширение для браузера	3,49 евро	-	+	+	+

1.2.1 Okko Party

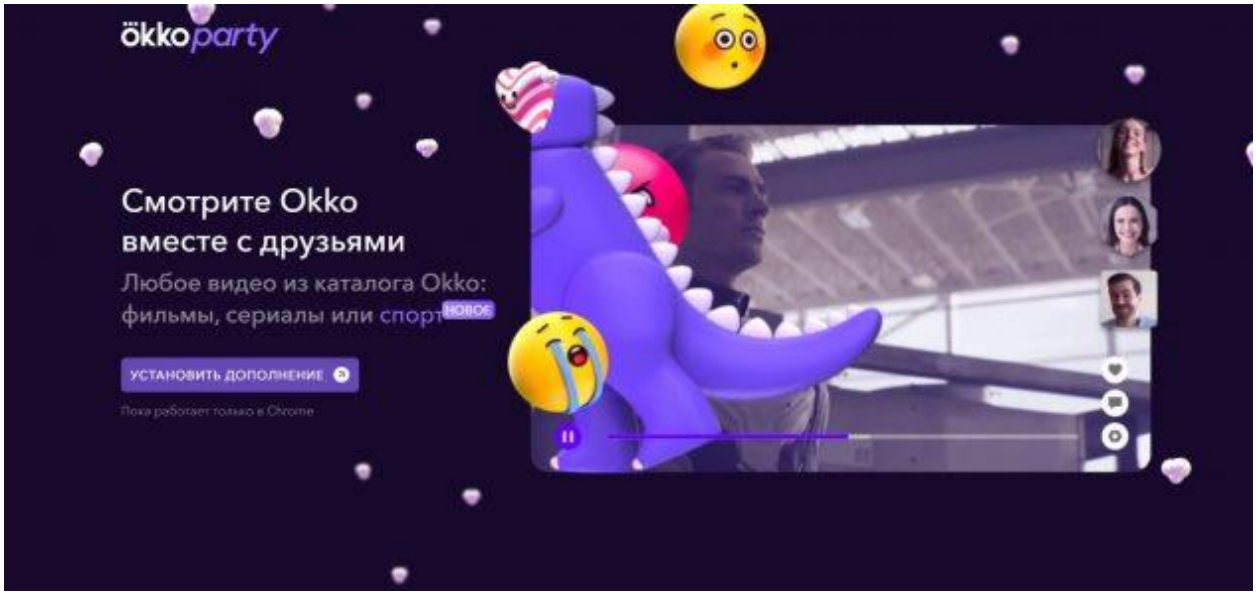

Рисунок 2 – Okko party

Okko является одним из крупнейших онлайн кинотеатров и недавно они запустили сервис совместного просмотра видео. На данный момент сервис включает в себя 40 тысяч фильмов и сериалов.

Преимущества:

- просмотр совместного видеоконтента;
- поддержка русского языка;
- наличие онлайн звонков и текстового чата.

Недостатки:

- требует установки расширения для браузера;
- за некоторые фильмы или сериалы необходимо доплачивать;
- отсутствие возможности загрузки своих видео;
- все участники совместного просмотра должны иметь активную подписку на сервис;
- сервис работает только на ПК.

1.2.2 Яндекс видео

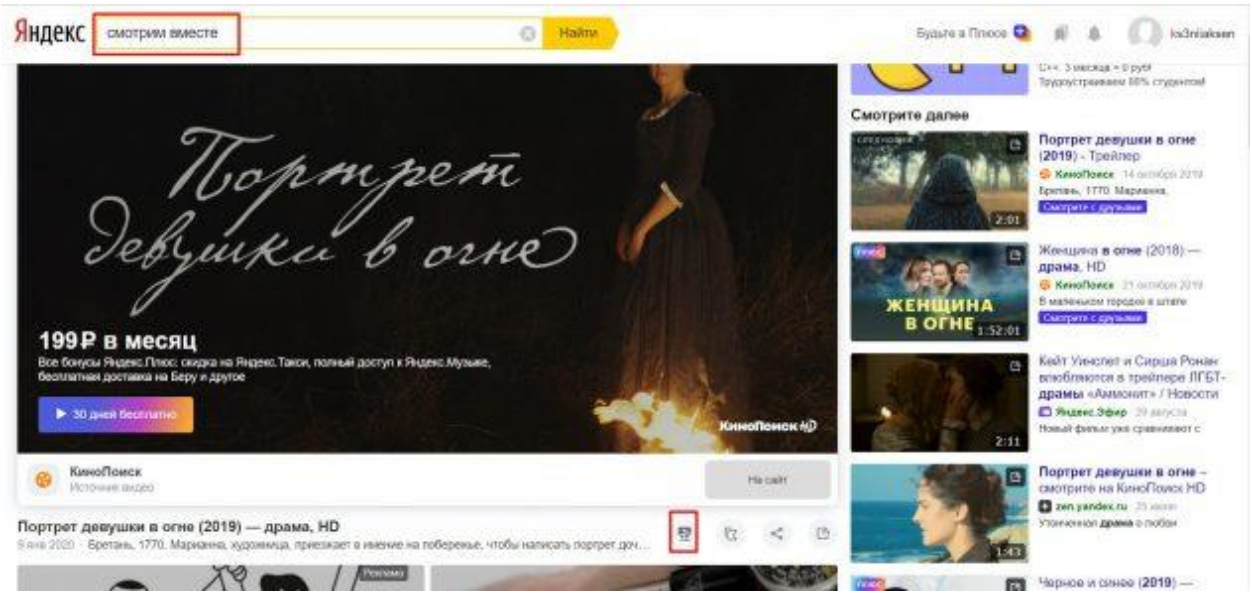

Рисунок 3 – Яндекс видео

Яндекс видео – сервис, который существует уже давно, однако функция совместного просмотра у него появилась совсем недавно. Сервис включает в себя весь контент «Яндекс.Эфира» и часть фильмов с дочерней компании «Кинопоиск HD».

Преимущества:

- не требует расширения для браузера;
- имеет поддержку русского языка;
- имеет поддержку онлайн чата;
- сервис кроссплатформенный.

Недостатки:

- за большую часть контента, предоставляемую сервисом, необходимо доплачивать;
- отсутствие голосового чата;
- отсутствие возможности загружать свои видео;
- не работает за пределами России.

1.2.3 Watch2gether

Рисунок 4 – Watch2gether

Данный сервис предлагает совместный просмотр контента, подгружая его с других источников, таких как: YouTube, Vimeo, SoundCloud, Netflix, Disney+.

Преимущества:

- возможность просмотра с разных источников;
- поддержка большого количества языков;
- поддержка аудио чата и онлайн переписки.

Недостатки:

- на каждый сервис, который вы хотите просматривать вместе, требуется подписка, причем каждому участнику;
- бесплатная версия сервиса имеет очень навязчивую рекламу;
- требует расширение для браузера.

1.2.4 Итоги обзора

Из обзора существующих решений, предоставленных в пунктах 1.2.1 – 1.2.3, можно сделать вывод, что все рассмотренные аналоги предназначены для совместного просмотра видео.

Каждое из рассмотренных решений позволяет общаться с помощью онлайн переписки.

Из 3 приложений, лишь 1 не требует дополнительных зависимостей.

Для комфортного использования сервисов, так или иначе требуется подписка с возможностью доплаты за отдельные услуги.

Watch2gether предоставляет бесплатный сценарий использования, однако из-за большого количества навязчивой рекламы, такой сценарий нельзя назвать комфортным.

Проанализировав существующие решения, можно сказать, что ни один из этих сервисов не обеспечивает полный функционал, который может предоставлять мой сервис.

1.3 Выбор инструментов

1.3.1 Выбор инструментов для клиентской части приложения

Для разработки клиентской части приложения был создан полноценный язык - javascript. Javascript является мультипарадигменным языком программирования, поддерживающий императивный и функциональный стили с момента выхода. Поддержка объектно-ориентированного подхода появилась гораздо позже.

Так как javascript является основным языком разработки клиентской части приложений, то большинство фреймворков для клиентской части приложений поддерживают его по умолчанию. Наиболее известными представителями front-end фреймворков являются:

- vue.js;
- react;
- angular.

Все они используются для одной задачи, а именно – Разработка реактивных, легко масштабируемых, скоростных приложений. Так как я знаком с vue.js, то выберу его в качестве front-end фреймворка.

1.3.2 Выбор инструментов для серверной части приложения

Для разработки серверной части приложения существует большое количество вариантов. Например, это язык C# с фреймворком ASP.net, кибо это будет язык php, который разрабатывался исключительно для того, чтобы создавать веб-приложения. Все они имеют одинаковые возможности, поэтому выбирать имеет смысл исходя из других параметров. Например, скорость, отказоустойчивость, удобство использования, нагрузка на хост-машину. Для разработки своей серверной части я выбрал язык php, он удобен в использовании, благодаря нестрогой типизации, а также отказоустойчив,

поскольку в случае ошибки выполнения скрипта, он будет просто прекращен, для остальных пользователей не будет никаких последствий.

PHP так же имеет богатый набор фреймворков, которые облегчают его использование, а так же расширяют функциональные возможности. Для реализации своей задачи я выбрал фреймворк Laravel [2].

1.3.3 Выбор базы данных

1.3.3.1 MySQL

MySQL – это быстрая и простая в использовании СУБД, используемая для многих малых и крупных приложений. MySQL обычно используется как сервер, обращение к которому происходит локально, либо с помощью удалённых клиентов. Однако в дистрибутив входит библиотека, которая позволяет встраивать MySQL в автономные программы.

Данная СУБД имеет 21 тип данных (INTEGER, FLOAT, DOUBLE, DATE, TIME, TEXT, ENUM ит.д.). Из достоинств данной СУБД хотелось бы отметить: безопасность, простоту в работе и богатый функционал, масштабируемость и скорость.

1.3.3.2 PostgreSQL

PostgreSQL – это одна из самых профессиональных СУБД, часто используется для веб-баз данных. Она соответствует стандартам SQL и свободно распространяется.

Отличие PostgreSQL от других СУБД в поддержке востребованного реляционного и объектно-ориентированного подхода к БД. PostgreSQL поддерживает надежные транзакции (атомарность, изоляционность, последовательность, прочность). Мощные технологии PostgreSQL делают

данную СУБД очень производительной. PostgreSQL имеет функции, которые упрощают использование повторяемых операций.

PostgreSQL имеет 28 типов данных, что на порядок больше, чем у других.

1.3.3.3 MongoDB

MongoDB реализует новый подход к построению баз данных, где не используются таблицы, схемы, SQL запросы, отсутствуют внешние ключи. MongoDB не является реляционной базой данных, а предлагает документо-ориентированную модель данных, благодаря чему работает быстрее, обладает лучшей масштабируемостью, а так же проста в использовании. Благодаря этому MongoDB позволяет хранить сложные по своей структуре данные, используя один из популярных стандартов обмена данными и их хранения – JSON.

1.3.3.4 Итог

Исходя из поставленной задачи, а также рассмотренных баз данных, можно сделать вывод, что нам требуется реляционная база данных, для хранения статичных данных, так как нам нужно указывать взаимосвязи между пользователями, фильмами, загруженными видео, в чем нам помогут внешние ключи. Выбирая между реляционными базами данных: PostgreSQL и MySQL выбор пал на первое, так как в перспективе дополнительные возможности PostgreSQL улучшит масштабируемость всего сервиса.

1.4 Динамическое хранилище данных

Для того, чтобы хранить динамические данных, а именно, созданные комнаты, информация о состоянии просмотра фильмов, пользователи,

находящиеся в комнатах, будет использоваться база данных Redis [3]. Redis — резидентная система управления базами данных класса NoSQL с открытым исходным кодом, работающая со структурами данных типа «ключ — значение». Такая база данных идеально подходит, для реализации поставленной цели.

1.5 Итоги анализа задания

На основе анализа аналогов было выявлено, что ни один из сервисов не реализует полный функционал бесплатно. При этом 2 из 3 рассмотренных решений требуют дополнительных зависимостей, в виде расширения для браузера.

Ни одно из рассмотренных решений не имеет возможности загрузки видео.

В качестве инструментов разработки были выбраны:

- vue.js;
- laravel;
- redis;
- postgreSQL;

2. Проектирование

На рисунке 5 приведена архитектура системы. Система состоит из пяти подсистем: Клиентское приложение, серверное приложение, echo-server, posgresql, redis. Клиентское приложение выполняет роль пользовательского интерфейса. Для получения данных оно общается с серверным приложением по протоколу HTTP, а так же через промежуточный echo-server по протоколу websocket. Серверное приложение, или же API, принимает запросы от клиента, обрабатывает их и возвращает ответ. Для хранения данных на сервере используется две базы данных. Одна хранит в себе статические данные, например: пользователи, фильмы, рейтинги, и тд. Другая хранит в себе динамические данные, такие как: Состояние комнаты, пользователи комнаты, текущее состояние плеера, временная метка позиции плеера, а также чат.

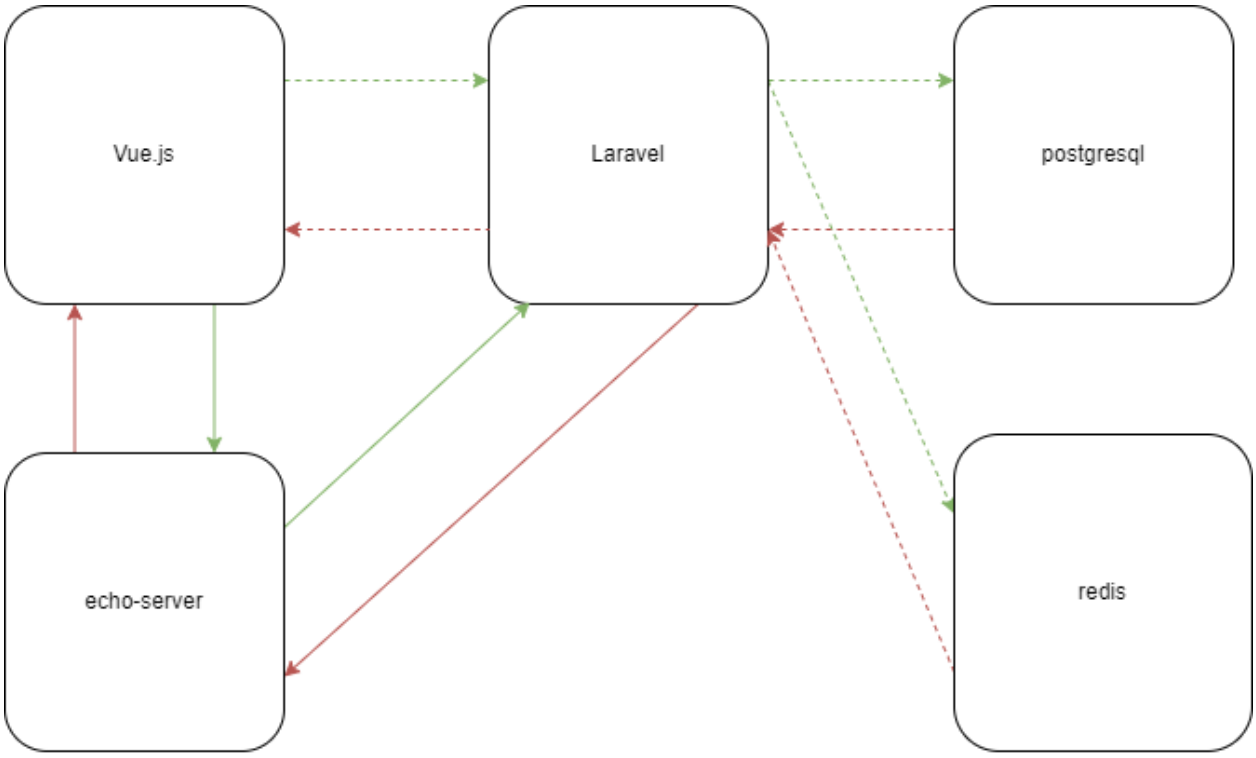

Рисунок 5 – Структура приложения

2.1 Динамическая модель системы

2.1.1 Диаграммы последовательностей

Значительная часть логики приложения связана с пользователями и фильмами, в разделе приведены диаграммы последовательностей, для наиболее значимых прецедентов, связанных со взаимодействием различных компонентов системы.

На рисунке 6 приведена диаграмма последовательностей для варианта использования «Просмотр профиля фильма», наглядно представляющая процесс взаимодействия клиента с сервером, для получения данных фильма и ее зависимости с пользователем, если он авторизован.

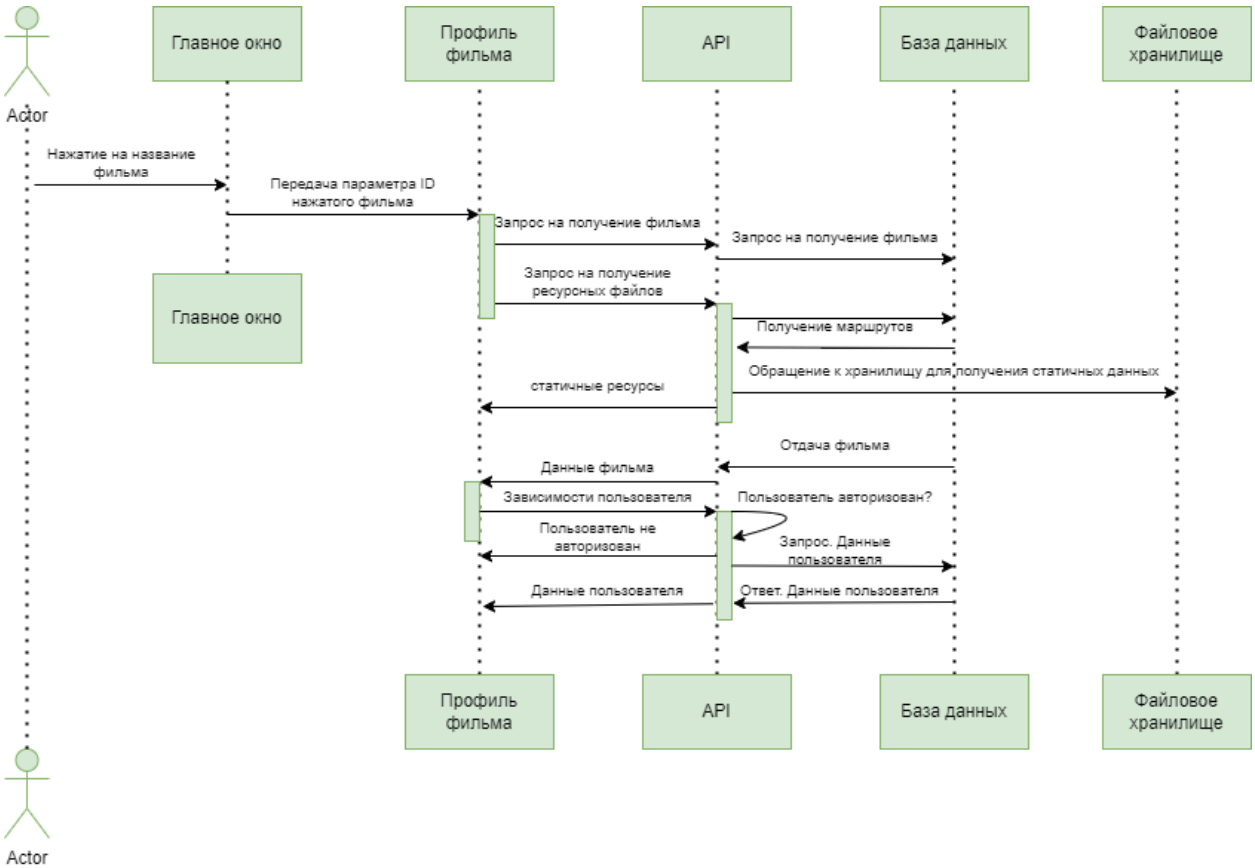

Рисунок 6 – Диаграмма последовательности для варианта «Просмотр профиля фильма»

Когда пользователь нажимает на название фильма, он переходит на страницу фильма, в этот же момент клиент отправляет два запроса, первый –

получение данных фильма. В них входят: рейтинг, жанр, тип, количество просмотров, описание, студия. Второй запрос выполняется для получения адресов статичных объектов. Запросы выполняются асинхронно. Когда основные данные загружены, мы можем сделать запрос для получения статуса просмотра фильма пользователем. Для этого отправляется запрос на сервер, если сервер понимает, что пользователь не авторизован, то ему возвращается статус 401. Если пользователь авторизован, то ему возвращается статус просмотра, а именно: «Смотрю», «Просмотрено», «Запланировано», «Брошено». Данные статусы можно редактировать в панели администратора.

На рисунке 7 изображена диаграмма последовательности, для создания виртуальной комнаты совместного просмотра. Этот прецедент интересен тем, что использует для соединения дополнительный протокол, а именно – websocket.

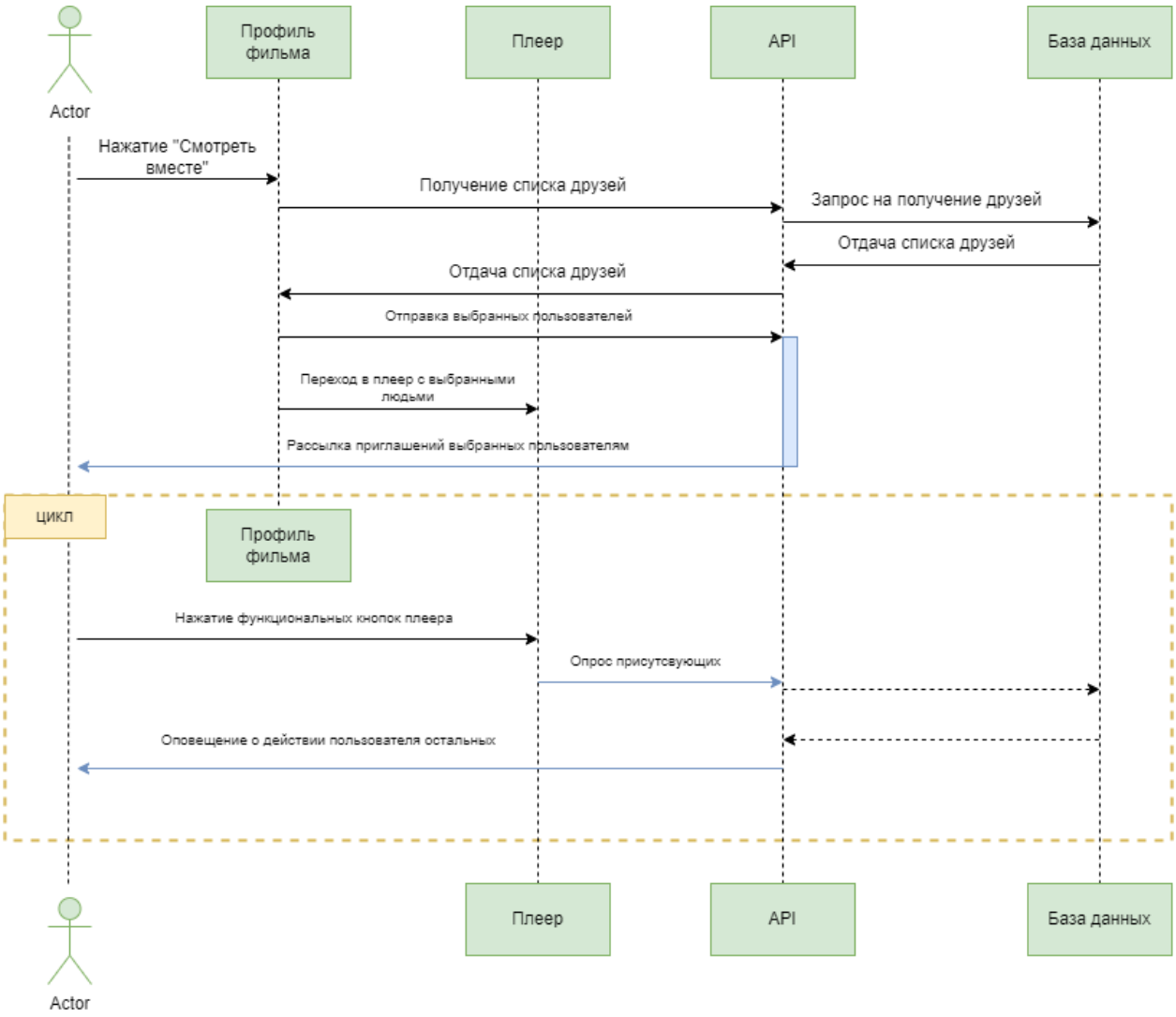

Рисунок 7 – Диаграмма последовательности «Комнаты совместного просмотра»

Когда пользователь нажимает кнопку «Посмотреть вместе», то ему показывается модальное окно, которое содержит его друзей, а также ранее созданные конференции. Когда пользователь выбрал людей, на сервер уходит запрос с этим списком, а ответ возвращается в виде триггера websocket,

каждому из выбранных пользователей. Созданная конференция отобразится пользователям в специальном окне на сайте, и они могут присоединится в любой момент, пока конференция существует. Если пользователь подключился к конференции, сервер его записывает в список присутствующих. Теперь взаимодействия каждого пользователя с плеером будут передаваться посредством протокола websocket остальным. Использование websocket выделено синими стрелками, на рисунке 7.

2.1.2 Диаграмма вариантов использования

На рисунке 8 изображена диаграмма вариантов использования.

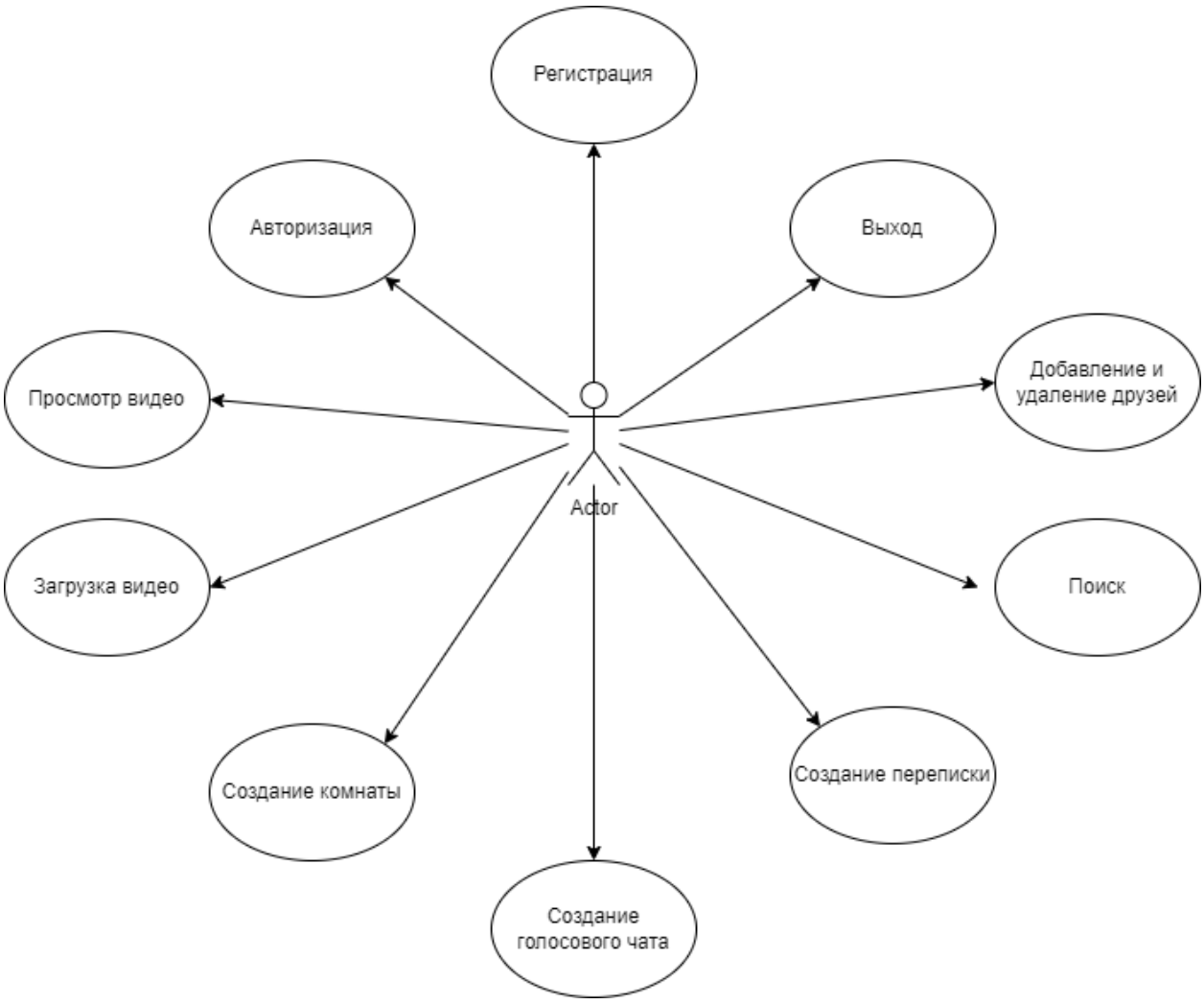

Рисунок 8 – диаграмма вариантов использования

Текстовое описание вариантов использования:

Название прецедента: Регистрация

Цель сценария: добавить пользователя в базу данных для дальнейшего взаимодействия с сервисом.

Предусловия: пользователь находится в окне авторизации и нажимает кнопку регистрация, после вводит данные и нажимает кнопку «Зарегистрироваться»

Основной сценарий:

А. Данные проходят проверку на корректность, после чего отправляются на сервер и проходят там проверку на наличие в базе данных.

Б. Данные добавляются в базу данных.

Постусловия: пользователь может войти в систему, используя регистрационные данные.

Название прецедента: Авторизация.

Цель сценария: проверить данные на корректность и наличие в базе, и открыть главное окно.

Предусловия: пользователь находится на главном экране, вводит свои данные и нажимает кнопку «Войти».

Основной сценарий:

А. Данные проходят проверку на корректность, после чего отправляются на сервер и проходят там проверку на наличие данных в базе.

Постусловия: пользователь попадает в главное окно.

Название прецедента: Выход

Цель сценария: Прекратить действующую сессию

Предусловия: Пользователь находится на любой странице сервиса и нажимает кнопку «Выход»

Основной сценарий:

А. Отправляется запрос на сервер с данными текущего пользователя, сервер обнуляет сессию данного пользователя.

Постусловия: Пользователь попадает на главное окно.

Название прецедента: Добавление и удаление друзей.

Цель сценария: Удаление либо добавление связей между пользователями

Постусловия: Пользователь находится в профиле другого пользователя и в зависимости от их текущих взаимоотношений нажимает кнопку «Удалить из друзей», либо «Добавить в друзья»

Основной сценарий:

А. Данные о двух пользователях поступают на сервер, их взаимоотношения меняются на противоположные.

Постусловия: Взаимоотношения пользователей заменены на противоположные.

Название прецедента: Поиск

Цель сценария: поиск по базе данных по названию, дате или жанру.

Предусловия: пользователь находится на главном экране, нажимает на текстовое поле «Поиск» и вводит текст.

Основной сценарий:

А. Происходит поиск по базе данных.

Постусловия: пользователь видит полученные результаты на экране.

Название прецедента: Просмотр видео

Цель сценария: запуск видеоконтента для просмотра.

Предусловия: пользователь находится на экране с видео, нажимает на выбранное видео.

2.1.3 Алгоритм использования «Совместного просмотра»

Для того, чтобы реализовать данную функцию, были использованы:

Протокол websocket, а так же NoSql база данных – redis. Websocket используются для жесткой связки пользователей, находящихся в одной комнате. Комната с пользователями хранится в базе данных redis.

При создании конференции, выбранных пользователям отправляются приглашения на участие. Если пользователь решил присоединится к конференции, то он подписывается на выделенный конференции канал websocket, по которому будут передаваться действия его и других пользователей. Так же пользователь вносится в конференцию в базе данных redis, что позволяет контролировать время жизни конференции, а так же хранить историю всех действий и ускорить все варианты действий, происходящих внутри конференции.

2.1.4 База статичных данных

Для хранения основной статичной информации, а именно: пользователи, фильмы, жанры, студии, статусы, роли, права и т.д. используется база данных. Ее структуру можно посмотреть на рисунке 9.

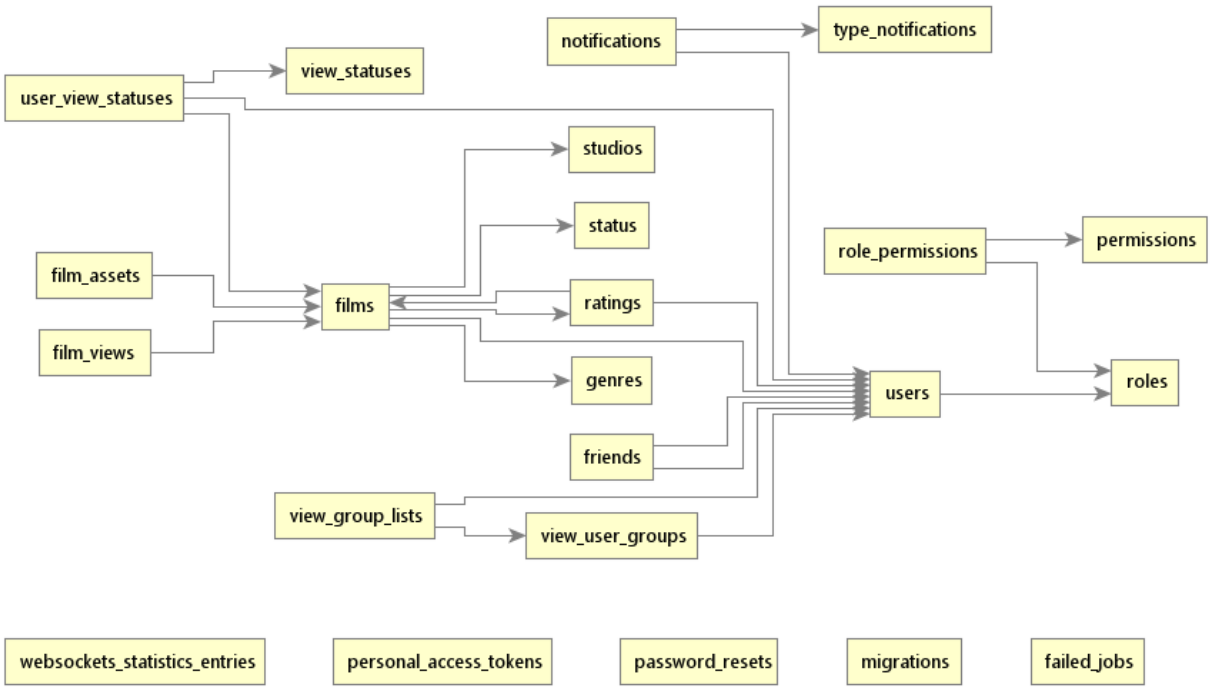

Рисунок 9 – ER-диаграмма(сокращенная) базы данных

На данном этапе база данных состоит из 19 связанных между собою таблиц и 5 несвязных таблиц, созданных для корректной работы фрейморка. Для наглядности были скрыты все поля. Основным таблицы являются: studios, films, users, view_statuses, genres, permissions, ratings, roles, notifications, film_assets, film_views, и так далее. Связывающими таблицами являются: user_view_statuses, role_permissions. Более детально ознакомиться с полями базы данных можно на рисунках 11,12,13.

Рисунок 11 – ER-Диаграмма части базы данных

Рисунок 12 – ER-Диаграмма части базы данных

Рисунок 13 – ER-Диаграмма части базы данных

2.1.5 База динамических данных

Для хранения динамических данных была выбрана модель, по которой объект базы сохраняется под ключом, который генерируется в зависимости от id пользователей, времени, а так же выбранного фильма.

Были определенны основные объекты:

- хранение групп, в которые входит пользователь, в качестве ключа используется его id с постфиксом “.groups”. Данный объект продемонстрирован на рисунке 14;

- хранение пользователей, входящие в каждую группу. В качестве ключа используется высчитанный hash группы. Данный объект продемонстрирован на рисунке 15;

- хранение информации о состоянии просмотра фильма, в качестве ключа используется высчитанный hash группы с постфиксом “filmStat”. Данный объект продемонстрирован на рисунке 16.

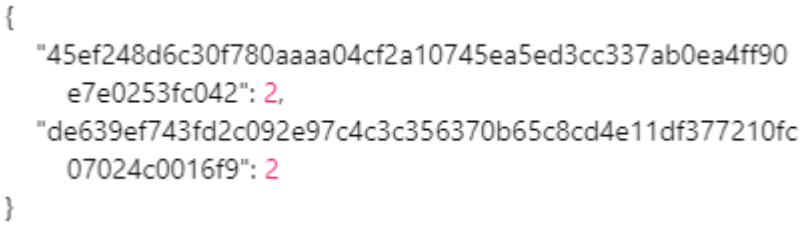

Рисунок 14 – Хранение групп, в которые входит пользователь

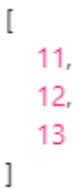

Рисунок 15 – Хранение пользователей, входящих в каждую группу

Рисунок 16 – Хранение информации о состоянии просмотра

2.2 Выводы по главе

В соответствии с технических заданием была предложена архитектура системы и структур баз данных, с помощью нотации диаграмм

последовательностей задокументированы наиболее сложные отношения в системе, более детально проработано взаимодействие объектов. Разработан алгоритм «Совместного просмотра».

3. Программная реализация

В процессе программной были решены следующие задачи:

- Выполнена верстка страниц.
- Реализованы модули основные модули для просмотра фильмов
- Реализована адаптивность
- Реализованы основные инструменты администрирования
- Реализованы системы хранилищ, как файловые, так и информационные
- Реализована навигация и авторизация, с проверкой прав доступа к Разным модулям.

3.1 Диаграмма классов

Для удобства использования и обращения к базе статичных данных, фреймворк Laravel предлагает привязывать классы (Далее именуемые, как модели) к сущностям из базы данных. Между моделями можно настроить отношения, что позволяет в последствии использовать «магические методы» фреймворка, для более удобного получения, сохранения, фильтрации информации. На рисунке 27 можно ознакомиться с диаграммой классов, разработанной в процессе реализации серверной части приложения.

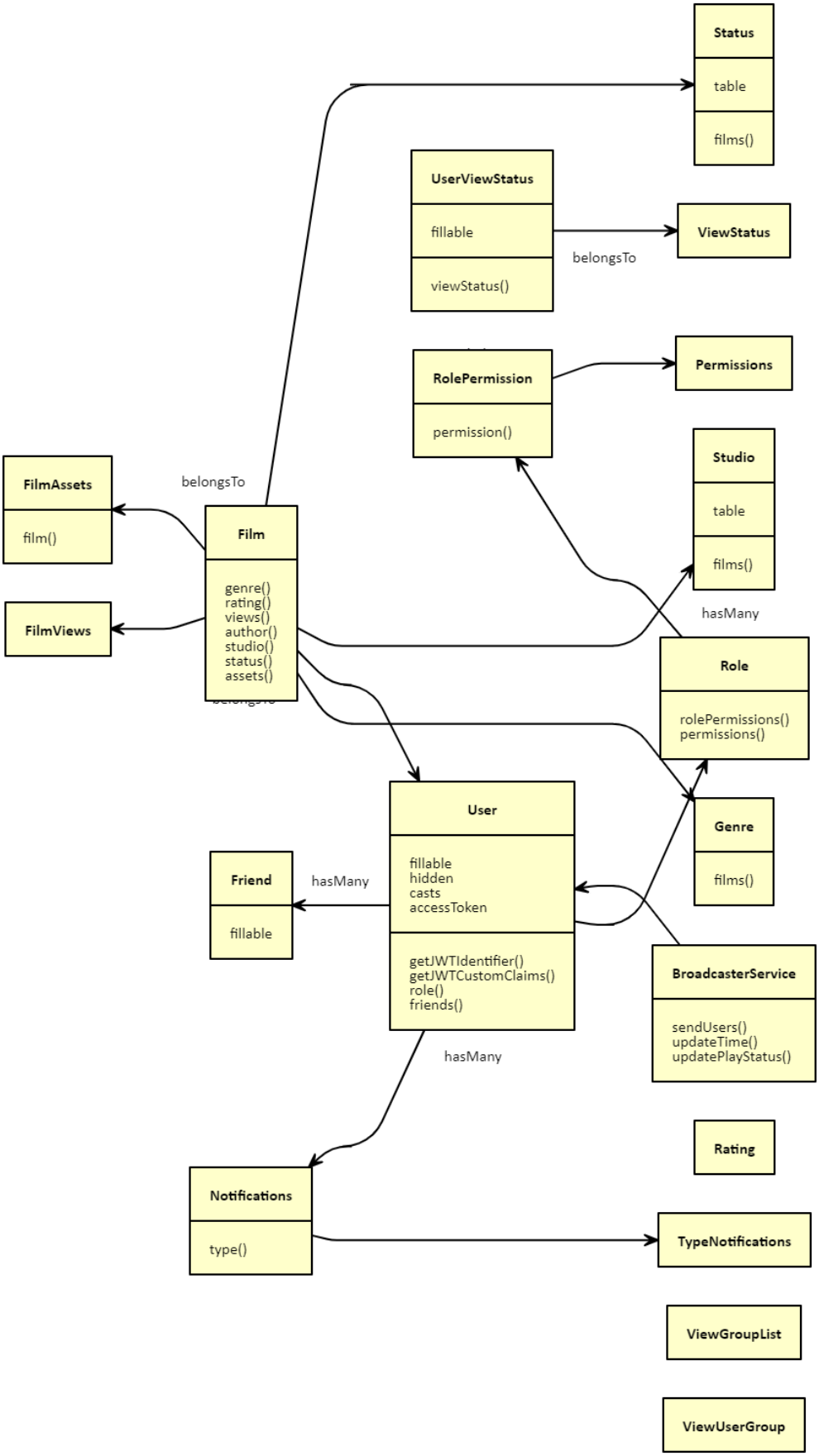

Рисунок 17 – Диаграмма Классов

3.2 Модуль авторизации

Для реализации авторизации была выбран стандарт для создания токенов доступа, именуемый JWT [4]. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности.

На рисунках 18 и 19. Можно ознакомиться с окнами регистрации и авторизации.

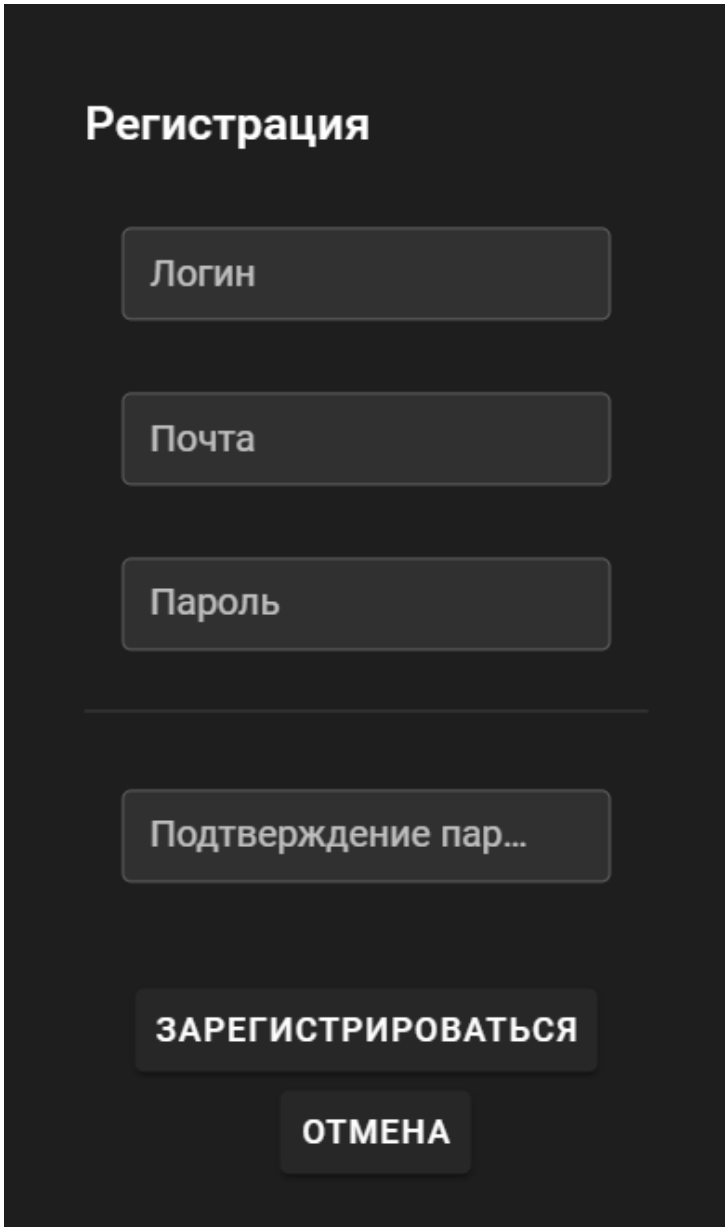

Рисунок 18 – Окно регистрации

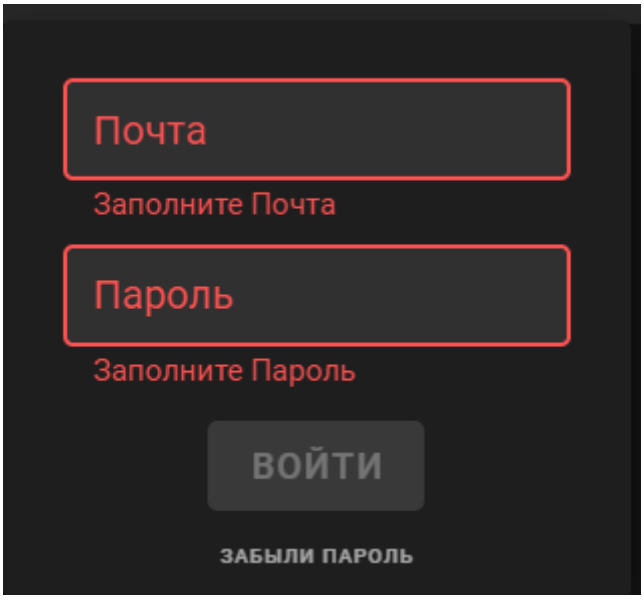

Рисунок 19 – Окно авторизации

3.3 Модуль главного окна

Главное окно было разработано по аналогии с существующими онлайн-кинотеатрами. Оно должно предоставлять максимально быстрый доступ к просмотру фильма, показывать афишу, а так же иметь поиск по заданным критериям, либо по названию. Стоит отменить, что афиша показывает не статичное изображение, а трейлер к фильму.

С главным окном можно ознакомиться на рисунке 20.

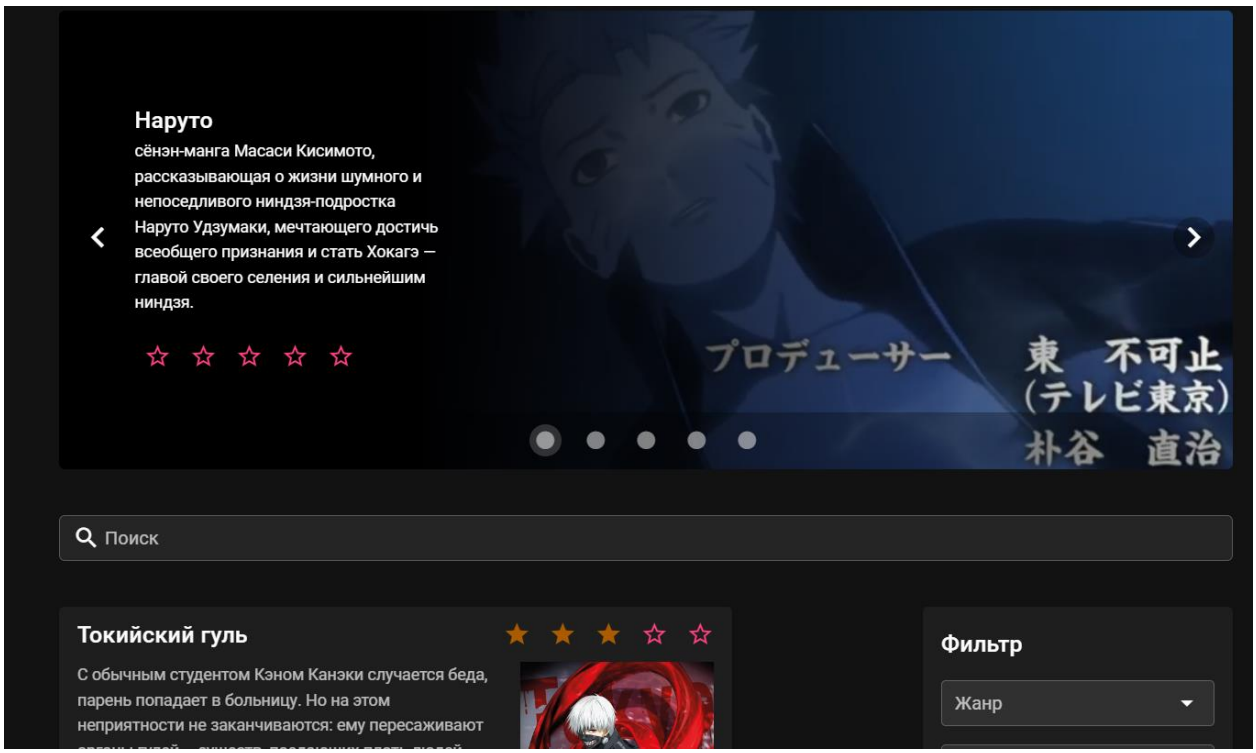

Рисунок 20 – Главное окно

3.4 Модуль профиля фильма

Профиль фильма, должен содержать в себе всю информацию о фильме, которая может понадобиться пользователю, а именно:

- рейтинг;
- жанр;
- год выхода;
- количество просмотров;
- тип;
- кем лицензировано;
- кем переведено.

Так же профиль фильма содержит в себе другие материалы, например: фотографии и картинки по мотивам этого фильма.

В самом низу расположен плеер, для одиночного просмотра.

Помимо этого, каждый авторизованный пользователь имеет право указать состояние просмотра данного фильма или сериала. Для этого

необходимо навести на аватар профиля. Ознакомиться с данным функционалом можно на рисунке 21. Ознакомиться со всем профилем можно на рисунке 22.

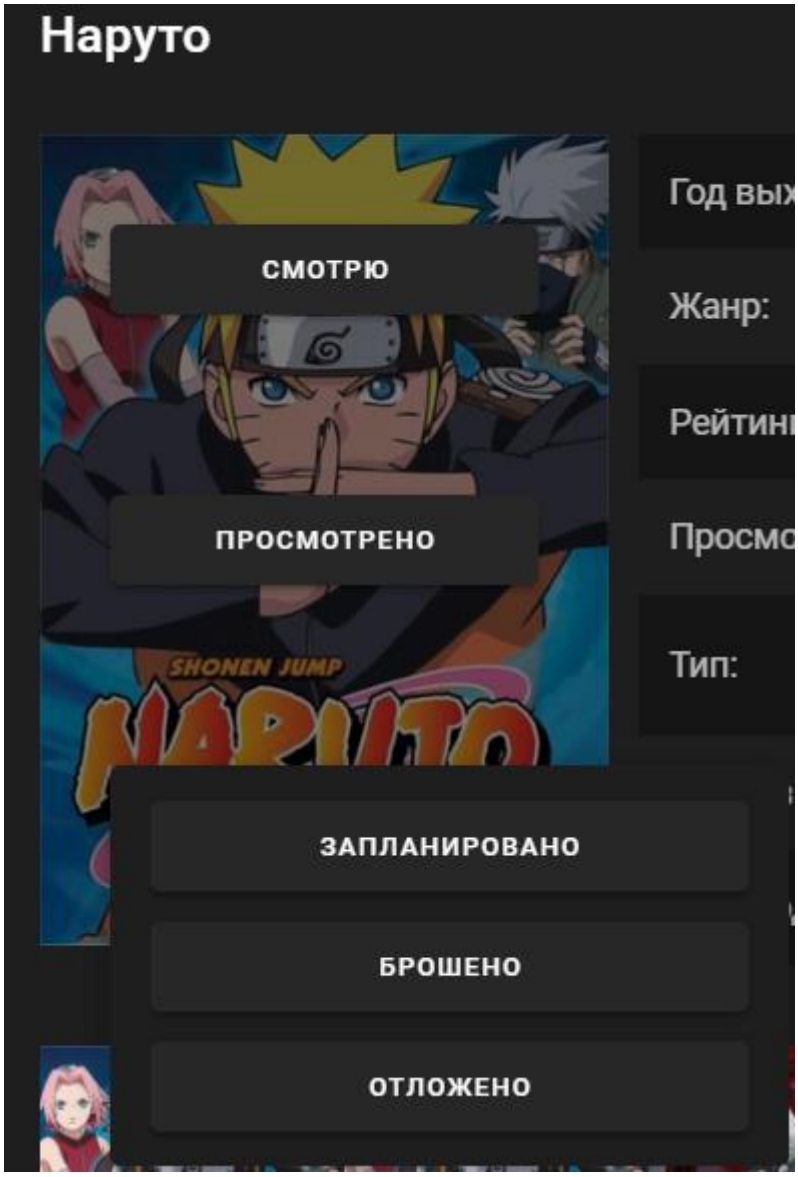

Рисунок 21 – Выбор состояния просмотра

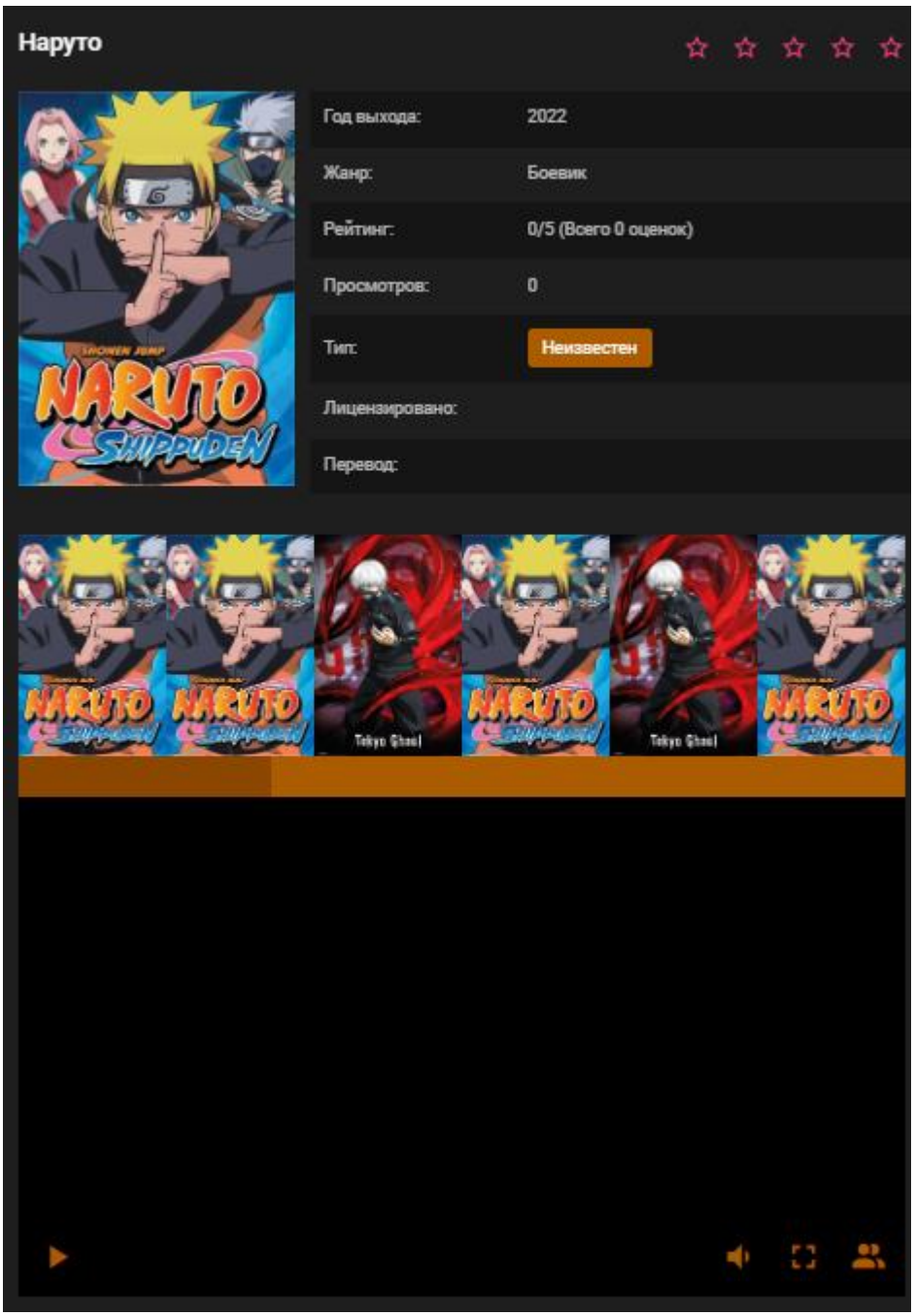

Рисунок 22 – Профиль фильма

3.5 Модуль плеера

Для реализации плеера за основу был взят стандартный плеер, предоставляемый языком разметки HTML. Однако поверх стандартных пунктов управления была разработана самостоятельная оболочка для того, чтобы расширить стандартный функционал, а также изменить стилистику

пунктов управления. Так, например, плеер для одиночного просмотра имеет кнопку для приглашения друзей, в плеере для совместного просмотра эта кнопка скрыта.

При одиночном просмотре плеер работает как обычный, однако, при переходе в режим совместного просмотра, плеер начинает работать по принципу авторитарного сервера. Пользователь, управляющий плеером, теперь отправляет серверу информацию о его намерениях, например: постановка паузы, а сервер дает команду плееру, на совершение этого действия. Так как любое действие пользователя оказывает влияние на всех пользователей в группе, то такой подход позволяет минимизировать задержку, так как пользователь, производивший какое-либо действие, так же как и все, ждет команду от сервера. Ознакомиться с одиночным плеером можно на рисунке 23.

Рисунок 23 – Модуль плеера

3.6 Модуль совместного просмотра

Для режима совместного просмотра был реализован отдельный модуль. В данном модуле пользователь имеет возможность посмотреть людей, находящихся с ним в одной комнате, а так же пообщаться в чате и посредством голоса и видео. Для включения чата над блоком пользователей есть переключатель. Для включения микрофона и камеры на иконе пользователя есть специальные кнопки. Ознакомиться с данным модулем можно на рисунке 24.

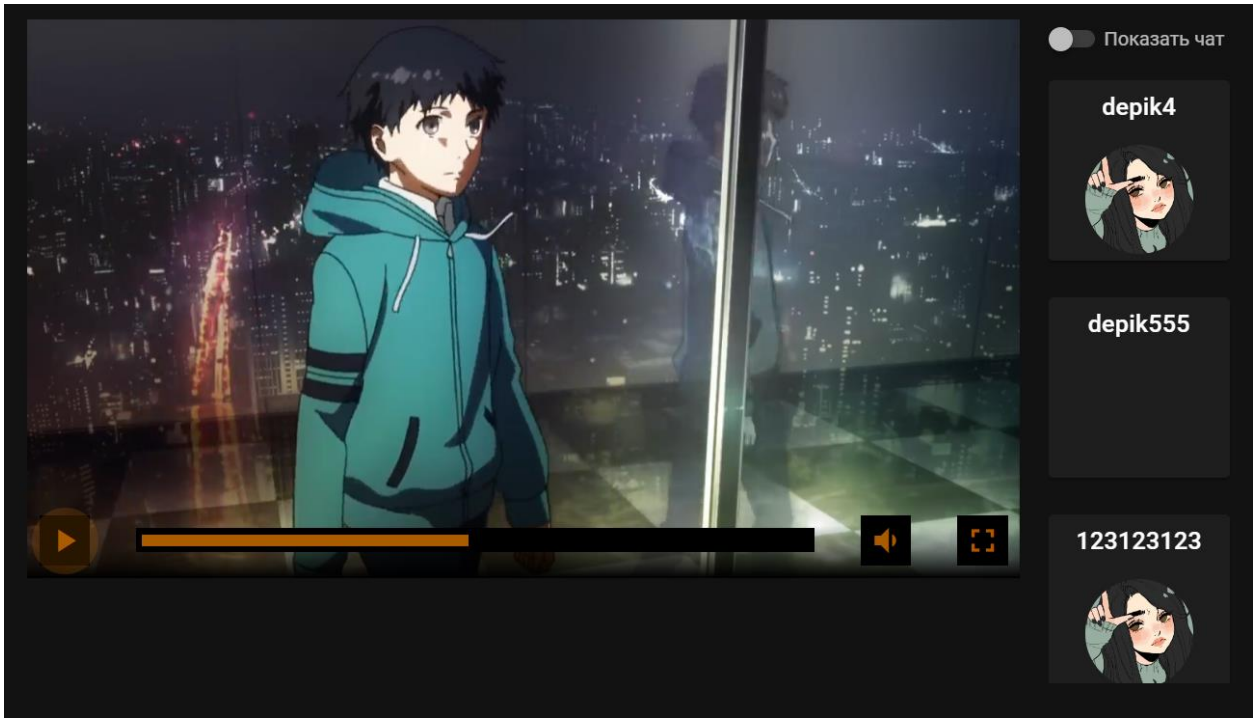

Рисунок 24 – Модуль совместного просмотра

3.7 Модуль администрирования

Для удобного администрирования данного приложения была разработана специальная административная панель. Пользователем можем попасть в нее посредством авторизации под Администратором, либо модератором. Данные права доступа выдаются владельцем приложения в этом же модуле. В данной административной панели можно редактировать

фильмы, пользователей, жанры, типы, студии. На рисунке 25 можно ознакомиться с данным модулем.

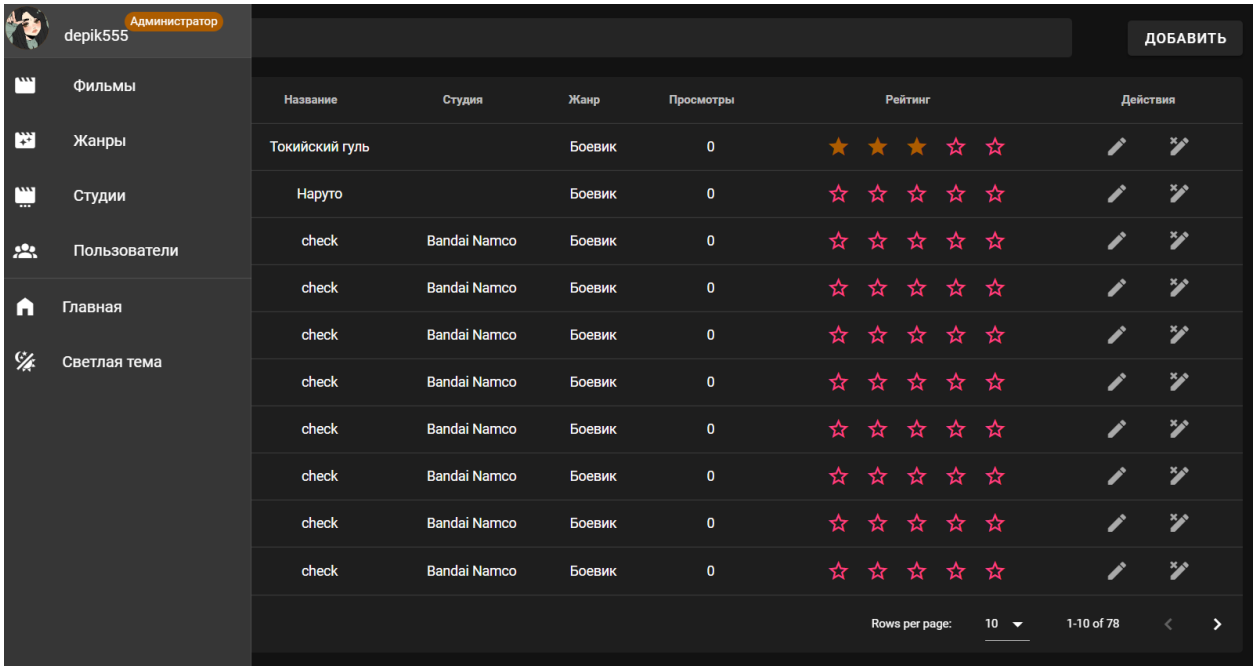

Рисунок 25 – Модуль администрирования

ЗАКЛЮЧЕНИЕ

В результате выполнения курсового проекта была изучена предметная область и существующие на данный момент аналоги.

После изучения аналогов был сформулирован ряд требований, предъявленных к сервису. На основе сформулированных требований были определенны технологии разработки. Клиентская часть была реализована с помощью фреймворка vue.js, серверная часть с помощью Laravel, базой данных была выбрана postgresSQL и redis.

В результате был разработан сервис для совместного просмотра видео-контента.

Была реализована верстка и логика для необходимых модулей, а также реализованы режимы доступа к сервису.

В дальнейшем требуется расширить функциональные возможности для взаимодействия между пользователями.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Tproger [Электронный ресурс]: – Режим доступа: https://tproger.ru;
2. Laravel [Электронный ресурс]: – Режим доступа: https://laravel.com/;
3. Redis [Электронный ресурс]: – Режим доступа: https://redis.io/;
4. JWT [Электронный ресурс]: - Режим доступа: https://jwt.io/;
5. Vue.js [Электронный ресурс]: - Режим доступа: https://ru.vuejs.org/.

Федеральное государственное автономное
образовательное учреждение
Высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

институт

Вычислительная техника

кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
«20» 06 2022 г.

БАКАЛАВАРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника

код и наименование специальности

Сервис совместного просмотра видеоконтента

тема

Руководитель	21.06.22	Старший преподаватель	О.В. Шмелев
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	21.06.22		П.А. Кононов
	подпись, дата		инициалы, фамилия
Нормоконтролер	21.06.22	Старший преподаватель	О.В. Шмелев
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2022