ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ИРКУТСКОЙ ОБЛАСТИ  
«АНГАРСКИЙ ТЕХНИКУМ СТРОИТЕЛЬНЫХ ТЕХНОЛОГИЙ»

Курсовая работа

Тема: «Сайт стримингового сервиса»

Разработал \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Осин Я.Е.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Денисюк А.В.

Ангарск, 2025

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 4](#_Toc198547998)

[**I.** **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ** 6](#_Toc198547999)

[**1.1 Среды для разработки веб-сайтов** 6](#_Toc198548000)

[**1.1.1 Visual Studio Code** 7](#_Toc198548001)

[**1.1.2 MySQL Workbench** 13](#_Toc198548002)

[**1.2 Используемые технологии** 17](#_Toc198548003)

[**1.2.1 HTML и CSS: базовые элементы веб-разработки** 17](#_Toc198548004)

[**1.2.2 JavaScript: интерактивность и динамика** 18](#_Toc198548005)

[**1.2.3 MySQL: работа с базами данных** 19](#_Toc198548006)

[**II.** **ПРАКТЧЕСКАЯ ЧАСТЬ** 21](#_Toc198548007)

[**2.1 Проектирование структуры веб-сайта стримингового сервиса** 21](#_Toc198548008)

[**2.1.1 Анализ требований к ресурсу** 21](#_Toc198548009)

[**2.1.2 Разработка структуры базы данных** 22](#_Toc198548010)

[**2.1.3 Проектирование пользовательского интерфейса** 23](#_Toc198548011)

[**2.2 Реализация функционала ресурса** 24](#_Toc198548012)

[**2.2.1 Разработка фронтенд-части** 24](#_Toc198548013)

[**2.2.2 Разработка бэкенд-части** 25](#_Toc198548014)

[**2.2.3 Интеграция фронтенда и бэкенда** 29](#_Toc198548015)

[**2.3 Тестирование и оптимизация ресурса** 29](#_Toc198548016)

[**2.3.1 Функциональное тестирование** 29](#_Toc198548017)

[**2.3.2 Оптимизация производительности** 30](#_Toc198548018)

[**2.3.3 Тестирование на различных устройствах** 30](#_Toc198548019)

[**Заключение** 34](#_Toc198548020)

[**Список использованных источников** 36](#_Toc198548021)

# **ВВЕДЕНИЕ**

Современный этап развития цифровых технологий характеризуется стремительным ростом популярности стриминговых сервисов, которые стали неотъемлемой частью повседневной жизни миллионов пользователей по всему миру. Стриминговые платформы, такие как Netflix, Spotify, YouTube и другие, предоставляют доступ к огромным массивам мультимедийного контента — фильмам, сериалам, музыке, трансляциям и видеозаписям — в режиме реального времени. Это стало возможным благодаря развитию интернет-технологий, увеличению скорости передачи данных и совершенствованию методов сжатия и доставки контента.

Актуальность темы курсовой работы обусловлена несколькими факторами. Во-первых, рынок стриминговых услуг продолжает активно развиваться, предлагая новые возможности для бизнеса и удовлетворения потребностей пользователей. Во-вторых, разработка стримингового сервиса требует комплексного подхода, включающего решение задач, связанных с архитектурой системы, обработкой и хранением больших объемов данных, обеспечением высокой производительности и безопасности, а также созданием удобного пользовательского интерфейса. В-третьих, изучение принципов работы стриминговых платформ позволяет глубже понять современные тенденции в области распределенных систем, мультимедиа и веб-технологий.

Целью данной курсовой работы является разработка концепции и прототипа стримингового сервиса, который сможет обеспечивать стабильную передачу мультимедийного контента с минимальными задержками и высоким качеством. Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ существующих стриминговых платформ и технологий, используемых для их реализации.

2. Определить основные требования к функциональности и архитектуре разрабатываемого сервиса.

3. Разработать прототип системы, включающий серверную часть, клиентское приложение и механизмы передачи данных.

4. Протестировать работоспособность системы и оценить её производительность.

Объектом исследования в данной работе являются стриминговые сервисы и технологии, используемые для их реализации. Предметом исследования выступают методы и подходы к разработке эффективных и масштабируемых стриминговых платформ.

Практическая значимость работы заключается в возможности применения полученных результатов для создания реальных стриминговых сервисов, а также в углублении знаний в области разработки распределенных систем и мультимедийных приложений.

Курсовая работа состоит из введения, трех глав, заключения и списка использованных источников. В первой главе рассматриваются теоретические аспекты стриминговых технологий, во второй главе описывается процесс проектирования и разработки сервиса, а в третьей главе представлены результаты тестирования и анализа работы системы.

# **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

# **1.1 Среды для разработки веб-сайтов**

Современная веб-разработка требует применения специализированных инструментов, обеспечивающих полноценный цикл создания программного продукта - от написания кода до тестирования и управления проектом. Интегрированные среды разработки нового поколения предоставляют возможности для комплексной работы как с клиентской, так и с серверной частью приложений, включая инструменты для проектирования и управления базами данных.

**В рамках данного проекта был сформирован оптимальный набор профессиональных инструментов:**

1. **Visual Studio Code** - многофункциональная среда для разработки фронтенд- (HTML, CSS, JavaScript) и бэкенд-компонентов

2. **MySQL Workbench** - специализированное решение для работы с реляционными базами данных

**Ключевые преимущества выбранных инструментов:**

**Visual Studio Code отличается расширяемой архитектурой, поддерживающей:**

* Интеллектуальное автодополнение кода
* Подсветку синтаксиса для различных языков программирования
* Встроенную консоль для выполнения команд
* Богатую экосистему расширений

**MySQL Workbench предлагает:**

* Визуальные средства проектирования схем данных
* Полнофункциональный редактор SQL-запросов
* Инструменты анализа и оптимизации производительности
* Возможности администрирования баз данных

**Комбинация этих инструментов формирует эффективную рабочую среду, где:**

- Visual Studio Code отвечает за аспекты программирования

- MySQL Workbench обеспечивает работу с данными

- Оба решения интегрируются в единый производственный процесс

**Такой технологический стек гарантирует:**

1. Качественную реализацию проекта на всех этапах

2. Гибкость при внесении изменений

3. Возможности для масштабирования решения

4. Поддержку современных стандартов разработки

5. Оптимизацию рабочих процессов

Использование профессиональных инструментов значительно повышает продуктивность разработки и обеспечивает стабильность работы конечного продукта.

# **1.1.1 Visual Studio Code**

Visual Studio Code (VS Code) представляет собой современный редактор исходного кода с открытой лицензией, созданный корпорацией Microsoft. Этот многофункциональный инструмент получил широкое признание в среде разработчиков благодаря своей адаптивности, высокой скорости работы и обширным возможностям кастомизации. Редактор находит применение в создании разнообразных программных продуктов - от веб-приложений и мобильных решений до десктопных программ и серверных компонентов. Благодаря своей универсальности и поддержке широкого спектра языков программирования, VS Code является оптимальным выбором для разработки веб-сайтов.

**Основные характеристики Visual Studio Code**

**Поддержка множества языков программирования**

Visual Studio Code обладает широкой языковой поддержкой, охватывая наиболее популярные технологии разработки - от веб-языков

(HTML, CSS, JavaScript, TypeScript) до серверных (PHP, Python, Java, C++) и многих других. Для каждого поддерживаемого языка редактор автоматически активирует набор профессиональных функций: интеллектуальную подсветку синтаксиса, контекстное автодополнение кода, автоматическое форматирование и т.д. Такая комплексная поддержка превращает VS Code в универсальную рабочую среду.

**Интеграция с Git и системами контроля версий**

Visual Studio Code оснащен встроенными инструментами для работы с Git, что дает разработчикам полный контроль над версиями кода без необходимости выходить из редактора. Благодаря глубокой интеграции системы контроля версий, вы можете наглядно отслеживать правки, фиксировать изменения, переключаться между ветками и разрешать конфликты — все это доступно через удобный графический интерфейс.

Такая реализация Git-функционала превращает VS Code в самодостаточную среду для командной разработки. Вам больше не нужно открывать дополнительные приложения или вспоминать консольные команды — все операции с версиями выполняются интуитивно понятными способами. Это особенно ценно при совместной работе над проектом, когда важно оперативно синхронизировать изменения и поддерживать порядок в кодовой базе.

**Расширяемость за счет плагинов и расширений**

Главным преимуществом Visual Studio Code является его модульная архитектура, позволяющая значительно расширять базовый функционал. **Богатая экосистема дополнений включает:**

* Поддержку редких и специализированных языков программирования
* Инструменты автоматического тестирования и проверки кода
* Интеграцию с популярными облачными платформами
* Средства статического анализа (линтеры)

**Для веб-разработчиков особенно полезны расширения:**

* Для работы с CSS-препроцессорами (SASS/LESS)
* Поддерживающие современные JS-фреймворки (React/Vue/Angular)
* Обеспечивающие взаимодействие с СУБД (MySQL/PostgreSQL)

Эта система дополнений превращает VS Code из простого редактора в мощную IDE, которую можно адаптировать под конкретные проектные требования. Разработчик сам определяет необходимый набор инструментов, создавая оптимальную среду для работы.

**Интегрированный терминал**

Visual Studio Code имеет встроенный терминал, позволяющий работать с командной строкой прямо в редакторе. Это удобно для запуска серверов, выполнения сборки через Webpack или Gulp, управления npm/yarn-зависимостями и автоматизации задач. Терминал поддерживает несколько сессий, различные оболочки (bash, PowerShell) и полностью интегрирован в процесс разработки. Благодаря этому все необходимые операции можно выполнять, не переключаясь между окнами, что ускоряет работу над проектом.

**Мощные инструменты отладки**

Visual Studio Code оснащен удобными инструментами для отладки кода, которые помогают разработчикам эффективно находить и исправлять ошибки. Встроенный отладчик позволяет устанавливать точки останова в любом месте программы, выполнять код пошагово и отслеживать значения переменных в процессе работы. Вы можете детально анализировать выполнение программы, просматривая стек вызовов функций и текущее состояние приложения. Эти профессиональные возможности особенно важны при разработке сложных веб-приложений, где необходимо точно понимать поток выполнения программы и взаимодействие между компонентами.

**Интеллектуальное автодополнение и рефакторинг**

Visual Studio Code значительно упрощает разработку благодаря интеллектуальной системе IntelliSense. Она анализирует код в реальном времени, предлагая точные автодополнения и снижая количество ошибок. Редактор также включает мощные инструменты рефакторинга, позволяющие безопасно оптимизировать структуру кода — переименовывать переменные, выделять методы и реорганизовывать компоненты без риска что-то сломать. Эти функции делают процесс написания кода быстрее и надежнее, позволяя разработчику сосредоточиться на логике приложения.

**Кроссплатформенность**

Visual Studio Code обладает кроссплатформенной поддержкой, работая на трех основных операционных системах — Windows, macOS и Linux. Благодаря такой универсальности, разработчики могут использовать знакомую среду разработки независимо от того, какую платформу они предпочитают для работы. Это особенно удобно для команд, где участники используют разные операционные системы, но хотят сохранить единую рабочую среду.

**Настраиваемый интерфейс**

VS Code предлагает широкие возможности кастомизации интерфейса. Разработчики могут персонализировать цветовые темы, шрифты и расположение элементов, создавая комфортную рабочую среду под свои нужды. Гибкие настройки позволяют оптимизировать пространство редактора, снижая нагрузку на глаза и повышая продуктивность при длительной работе с кодом.

**Применение Visual Studio Code в разработке веб-сайта**

Для создания веб-сайта VS Code может быть использован на всех этапах разработки:

1. **Фронтенд-разработка:** Создание интерфейсов на HTML/CSS/JS с поддержкой препроцессоров (SASS/LESS) и фреймворков (React/Vue) позволяет создавать современные и интерактивные веб-страницы.
2. **Бэкенд-разработка:** Разработка серверной части на PHP, Node.js или Python с инструментами отладки и тестированием серверной логики.
3. **Работа с базами данных:** Работа с MySQL/PostgreSQL через плагины и расширения.
4. **Управление проектом:** Встроенный терминал и Git для контроля версий и командной работы.

**Преимущества Visual Studio Code**

1. Благодаря оптимизированной архитектуре редактор демонстрирует высокую производительность даже на компьютерах с ограниченными ресурсами, что особенно важно для начинающих специалистов или команд со скромным бюджетом на оборудование.
2. Доступность решения подчеркивается не только его бесплатной лицензией, но и открытостью исходного кода. Это позволяет опытным разработчикам изучать внутреннюю структуру, участвовать в улучшении продукта и создавать собственные модификации, что способствует постоянному развитию экосистемы.
3. Огромное сообщество пользователей обеспечивает всестороннюю поддержку - от официальной документации до многочисленных форумов, видеоуроков и тематических блогов. Такое информационное богатство значительно сокращает время решения возникающих проблем.
4. Гибкость настройки охватывает как визуальные аспекты (темы оформления, шрифты), так и функциональные (подбор расширений под

конкретный технологический стек). Эта адаптивность делает VS Code универсальным инструментом для различных сфер разработки.

1. Широкая совместимость с современными технологиями позволяет использовать редактор для работы с актуальными фреймворками и языками программирования, обеспечивая комфортную среду как для фронтенд-, так и для бэкенд-разработки. Поддержка облачных сервисов и систем контроля версий дополняет этот впечатляющий функционал.

**Недостатки Visual Studio Code**

1. Базовый функционал редактора имеет определенные рамки - для многих специализированных задач потребуется дополнительная установка расширений, что увеличивает время первоначальной настройки рабочей среды.
2. В отличие от узкоспециализированных IDE, VS Code не предоставляет встроенных инструментов для таких направлений как мобильная разработка или создание игр, где более подходящими решениями будут Android Studio или Unity.
3. Экосистема расширений, будучи одним из главных преимуществ редактора, одновременно создает и определенные риски. Качество и стабильность работы дополнительных модулей полностью зависит от их разработчиков, что иногда приводит к несовместимости или неожиданным ошибкам.
4. Несмотря на кажущуюся простоту интерфейса, начинающим программистам потребуется значительное время для полноценного освоения всего функционала. Кривая обучения может быть достаточно крутой, особенно при работе с комплексными проектами, требующими тонкой настройки окружения.

# **1.1.2 MySQL Workbench**

MySQL Workbench — это официальный графический инструмент от Oracle для работы с MySQL. Он предоставляет полный набор функций для проектирования, администрирования и оптимизации баз данных. Среда включает визуальное моделирование через ER-диаграммы, интеллектуальный SQL-редактор, инструменты миграции и мониторинга производительности. Решение сочетает удобный интерфейс с профессиональными возможностями управления пользователями, репликацией и резервным копированием, оставаясь полностью совместимым со всеми версиями MySQL.

**Основные характеристики MySQL Workbench**

**Проектирование баз данных (Data Modeling)**

MySQL Workbench включает мощные средства визуального проектирования структур данных. С помощью инструментария Data Modeling разработчики могут создавать детализированные ER-диаграммы, отображающие полную схему базы - сущности, их атрибуты, взаимосвязи и ключевые элементы. Такой подход особенно ценен при проектировании веб-приложений, позволяя на этапе архитектурного проектирования четко определить организацию данных, продумать отношения между таблицами и оптимизировать структуру будущей базы. Визуальное представление схемы помогает избежать логических ошибок и создать эффективную модель хранения информации.

**Разработка и выполнение SQL-запросов**

MySQL Workbench предлагает полнофункциональный SQL-редактор, значительно упрощающий работу с запросами. Интеллектуальная подсветка синтаксиса и контекстное автодополнение делают процесс написания кода более удобным и эффективным. Встроенные механизмы проверки ошибок помогают выявлять проблемы

до выполнения запросов. Особенно полезной является возможность сохранения часто используемых запросов, что позволяет создавать персональную коллекцию SQL-шаблонов для регулярных операций.

**Администрирование баз данных**

MySQL Workbench предоставляет полный набор инструментов для администрирования серверов MySQL. Включает настройку параметров, управление пользователями и правами, резервное копирование данных и мониторинг производительности в реальном времени. Все функции доступны через удобный графический интерфейс.

**Миграция данных**

MySQL Workbench включает мощный инструмент миграции, который позволяет легко переносить данные между различными СУБД. Поддерживается импорт из популярных систем, включая Microsoft SQL Server, PostgreSQL и SQLite. Эта функция особенно полезна в двух случаях: при переносе существующих проектов на MySQL или при необходимости интеграции с другими базами данных. Инструмент автоматизирует процесс конвертации, сохраняя целостность данных и минимизируя ручную работу.

**Мониторинг производительности**

MySQL Workbench оснащен продвинутыми инструментами для анализа производительности сервера в реальном времени. Разработчики могут отслеживать ключевые метрики: нагрузку на процессор и память, активные соединения, статистику выполнения запросов. Встроенные средства визуализации помогают быстро выявлять узкие места и bottlenecks в работе базы данных, что позволяет своевременно оптимизировать запросы и конфигурацию сервера для повышения общей производительности системы.

**Интеграция с другими инструментами**

MySQL Workbench обеспечивает плавную интеграцию с популярными средами разработки, включая Visual Studio Code. Эта функциональность создает единое рабочее пространство, где можно одновременно работать с базой данных и прикладным кодом. Такая связка инструментов значительно ускоряет процесс разработки, позволяя выполнять все необходимые операции без переключения между разными приложениями.

**Применение MySQL Workbench в разработке веб-сайта**

Для веб-сайта MySQL Workbench может быть использован для следующих задач:

1. **Проектирование структуры БД**

Создание детализированных ER-диаграмм для визуализации связей между таблицами, что обеспечивает четкую организацию данных.

2. **Работа с SQL-запросами**

Формирование и выполнение запросов любой сложности - от простой выборки информации до комплексных операций обновления пользовательских данных.

**3. Администрирование и безопасность**

- Управление учетными записями и правами доступа

- Настройка автоматического резервного копирования

- Восстановление данных при необходимости

4. **Оптимизация производительности**

Анализ выполнения запросов с последующей тонкой настройкой для ускорения работы веб-ресурса, особенно при высокой нагрузке.

**Преимущества MySQL Workbench**

* **Интуитивный интерфейс**

Графическая среда Workbench с продуманным UX-дизайном снижает порог входа для новичков, сохраняя профессиональный функционал для опытных пользователей.

* **Полная совместимость с MySQL**

Поддержка всего спектра возможностей СУБД — от проектирования схем до тонкой настройки серверных параметров и анализа производительности.

* **Кроссплатформенная доступность**

Нативные версии для всех популярных ОС (Windows, Linux, macOS) обеспечивают идентичный опыт работы независимо от платформы.

* **Гибкая интеграция**

Совместимость с VS Code и другими IDE позволяет использовать MySQL Workbench в рамках единого рабочего процесса.

* **Открытая лицензия**

Бесплатная модель распространения без ограничений функциональности делает инструмент доступным для индивидуальных разработчиков и команд.

**Недостатки MySQL Workbench**

* **Сложность освоения**

Начинающим пользователям потребуется время для изучения всего функционала, несмотря на продуманный интерфейс, особенно при работе с продвинутыми функциями администрирования.

* **Узкая специализация**

Хотя инструмент поддерживает базовый импорт данных из других СУБД, его основные возможности оптимизированы исключительно для MySQL, что ограничивает универсальность.

* **Ресурсоемкость**

Работа с объемными базами данных требует значительных вычислительных мощностей — на слабых машинах могут наблюдаться задержки при выполнении сложных операций.

# **1.2 Используемые технологии**

Для разработки веб-сайта используются современные технологии, которые обеспечивают создание функционального, интерактивного и производительного веб-приложения. Эти технологии охватывают как фронтенд (клиентскую часть), так и бэкенд (серверную часть), а также работу с базами данных. Рассмотрим основные технологии, которые будут применяться в проекте.

# **1.2.1 HTML и CSS: базовые элементы веб-разработки**

**HTML (HyperText Markup Language)**

HTML — это базовый язык разметки, формирующий структуру веб-страниц. Он задает ключевые элементы контента: заголовки различных уровней, текстовые абзацы, медиа-вставки, навигационные ссылки и интерактивные формы. Будучи фундаментальной технологией веб-разработки, HTML создает структурный каркас, обязательный для любого сайта — от простейшей информационной страницы до сложного веб-приложения. Без HTML-разметки невозможно корректное отображение контента в браузере, так как именно она определяет расположение и тип всех элементов на странице.

**CSS (Cascading Style Sheets)**

CSS — представляет собой мощный инструмент стилизации веб-страниц, отвечающий за их визуальное представление. Этот специализированный язык позволяет разработчикам точно контролировать внешний вид элементов, задавая такие параметры как

цветовые схемы, типографику, пространственное расположение компонентов и динамические эффекты. Благодаря CSS создаются эстетически привлекательные интерфейсы с продуманной эргономикой,

что существенно повышает удобство взаимодействия пользователей с веб-ресурсом. От простых статических страниц до сложных интерактивных приложений - CSS обеспечивает единообразие визуального оформления и адаптивность под различные устройства и экраны.

**Преимущества использования HTML и CSS:**

• Простота изучения и использования.

• Широкая поддержка всеми современными браузерами.

• Возможность создания адаптивного и кроссбраузерного дизайна.

• Высокая скорость загрузки страниц благодаря лёгкости кода.

# **1.2.2 JavaScript: интерактивность и динамика**

**JavaScript**

JavaScript представляет собой скриптовый язык программирования, который придает веб-страницам динамичность и интерактивность. С его

помощью реализуются сложные поведенческие сценарии, включая обработку пользовательских действий (нажатия, ввод данных), создание плавных анимаций и асинхронную подгрузку контента. Эти возможности позволяют существенно улучшить пользовательский опыт, делая взаимодействие с веб-приложением более плавным и отзывчивым, без необходимости полной перезагрузки страницы при каждом действии пользователя.

**Фреймворки и библиотеки JavaScript**

Для оптимизации процесса разработки и улучшения производительности часто применяются современные фреймворки и библиотеки:

**Node.js** — среда выполнения JavaScript, которая позволяет создавать масштабируемые серверные приложения. Ее модульная архитектура помогает организовать код в независимые компоненты, упрощая разработку и дальнейшее сопровождение проекта.

**Express.js** — минималистичный веб-фреймворк для Node.js, обеспечивающий высокую скорость работы и простоту в освоении. Благодаря своей гибкости, он идеально подходит для быстрого создания RESTful API и веб-приложений.

**Преимущества использования JavaScript:**

• Возможность создания интерактивных и динамических веб-страниц.

• Широкая поддержка браузерами и сообществом разработчиков.

• Интеграция с современными фреймворками и библиотеками.

• Высокая производительность и скорость выполнения.

# **1.2.3 MySQL: работа с базами данных**

MySQL представляет собой реляционную систему управления базами данных, широко применяемую для структурированного хранения и эффективного управления информацией. Эта СУБД завоевала популярность в веб-разработке благодаря оптимальному сочетанию надежности, высокой производительности и относительно простого освоения. MySQL обеспечивает стабильную работу с большими объемами данных, поддерживая при этом стандартные SQL-запросы и предлагая различные механизмы оптимизации. Ее открытая архитектура и кроссплатформенность делают MySQL универсальным решением для проектов любого масштаба.

**Основные особенности MySQL:**

* Полноценная поддержка стандартного SQL-синтаксиса для выполнения любых операций с данными.
* Оптимизированная работа запросов благодаря продуманной системе индексации.
* Надежные механизмы защиты информации, включая транзакции с поддержкой ACID-свойств и автоматизированное резервное копирование.
* Гибкая масштабируемость для обработки растущих объемов информации.

**Интеграция с веб-приложением:**

* **Взаимодействие с базой данных**

Node.js позволяет выполнять полный спектр SQL-операций: выборку, вставку, модификацию и удаление данных через запросы к MySQL.

* **Обработка и передача данных**

Полученная из БД информация структурируется и преобразуется серверными языками перед отправкой на клиентскую сторону.

* **Безопасность операций**

Использование подготовленных запросов (prepared statements) и механизмов шифрования защищает систему от SQL-инъекций и утечек данных.

**Преимущества использования MySQL:**

• Надёжность и стабильность работы.

• Широкая поддержка сообществом и документация.

• Интеграция с различными языками программирования и платформами.

• Возможность работы с большими объёмами данных.

# **ПРАКТЧЕСКАЯ ЧАСТЬ**

# **2.1 Проектирование структуры веб-сайта стримингового сервиса**

# **2.1.1 Анализ требований к ресурсу**

Веб-сайт стримингового сервиса разрабатывался с учетом современных потребностей пользователей. Основные требования к проекту включали:

1. **Функциональные возможности**

Платформа предоставляет доступ к медиатеке контента (название, описание, жанр, год выпуска), реализует систему поиска, а также позволяет пользователям оставлять рецензии и оценки просмотренному контенту.

2. **Пользовательский интерфейс**

Сервис обладает адаптивным веб-интерфейсом с интуитивной навигацией, автоматически подстраивающимся под различные типы устройств (ПК, планшеты, смартфоны) и разрешения экранов.

3. **Система безопасности**

Защита данных пользователей при регистрации и авторизации с использованием хеширования паролей (библиотека bcrypt).

4. **Оптимизация производительности**

Быстрая загрузка страниц и эффективная работа с базой данных.

# **2.1.2 Разработка структуры базы данных**

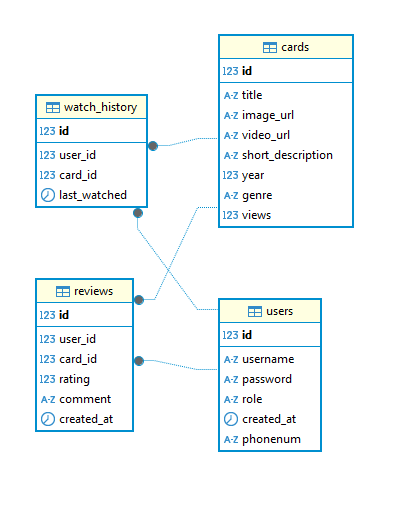
База данных была спроектирована с использованием MySQL и включает следующие таблицы:

Рисунок 1 Таблицы в БД со связями

**cards**: хранит информацию о видеоконтенте

**id**: идентификатор видеоконтента

**title**: название видеоконтента

**image\_url**: постер видеоконтента

**video\_url**: сам видеоконтент

**short\_description**: описание видеоконтента

**year**: год выхода

**genre**: жанр

**views**: количество просмотров

**watch\_history**: история просмотра пользователя

**id**: идентификатор в коде

**user\_id**: идентификатор пользователя

**card\_id**: идентификатор карточки видеоконтента

**last\_watched**: дата и время просмотра

**reviews**: информация о отзывах оставленных пользователем

**id:** идентификатор в коде

**user\_id:** идентификатор пользователя

**card\_id:** идентификатор видеоконтента

**rating:** оценка которую поставил пользователь

**comment:** отзыв пользователя

**created\_at:** дата и время когда пользователь оставил отзыв

**users:** информация о пользователе

**id:** идентификатор пользователя

**username**: логин пользователя

**password**: пароль пользователя

**role**: роль пользователя (пользователь\администратор)

**created\_at**: дата и время регистрации

**phonenum**: номер телефона

Связи между таблицами организованы через внешние ключи, что обеспечивает целостность данных. Например, отзыв (reviews) связан с игрой (cards) и пользователем (users).

# **2.1.3 Проектирование пользовательского интерфейса**

Интерфейс веб-сайта был разработан с использованием следующих технологий:

* **HTML и CSS**: Для создания структуры страниц и их визуального оформления.
* **EJS (Embedded JavaScript)**: шаблонизатор для динамического

отображения данных на сервере.

* **JavaScript**: для интерактивности, например, загрузки отзывов без перезагрузки страницы.

**Страницы ресурса:**

* **Главная страница**: Список фильмов с возможностью поиска.
* **Страница фильма**: Показывает подробную информацию о фильме: название, описание, год выхода, жанр, количество просмотров, рейтинг. А также, можно оставить отзыв и перейти на страницу похожего фильма.
* **Страницы регистрации и авторизации**: Содержат формы для ввода данных пользователя (email, пароль, номер телефона для регистрации).
* **Личный кабинет пользователя**: Данные пользователя и его история просмотра.
* **Административные страницы**: добавление и удаление фильмов (доступно только администраторам).

# **2.2 Реализация функционала ресурса**

# **2.2.1 Разработка фронтенд-части**

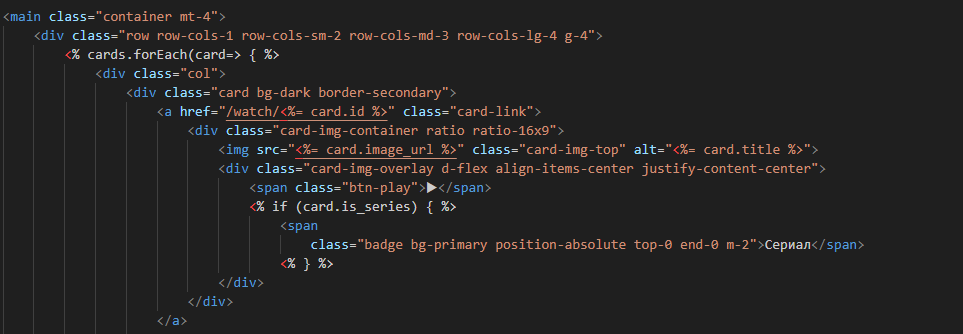
 Фронтенд-часть использует EJS для рендеринга данных на сервере. Например, главная страница выводит карточки фильмов, которые получает из базы данных:

Рисунок 2 Создание карточек из данных БД

# **2.2.2 Разработка бэкенд-части**

Серверная часть приложения разработана на платформе Node.js с использованием Express — минималистичного веб-фреймворка для обработки HTTP-запросов. Архитектура бэкенда включает следующие ключевые компоненты:

* **Обработка HTTP-запросов**

**GET-запросы:**

Используются для получения данных (например, списка фильмов или информации о пользователе). Сервер обрабатывает запрос, извлекает данные из БД и возвращает их в формате HTML.

**POST-запросы:**

Применяются для отправки данных на сервер (логин/регистрация, добавление контента). Данные проверяются перед обработкой.

* **Работа с базой данных**

**SQL-запросы:**

Через модуль db выполняются запросы к реляционной БД

**Безопасность:**

Для предотвращения SQL-инъекций используются параметризованные запросы.

* **Аутентификация и авторизация**

**Middleware isAuthenticated:**

Проверяет, авторизован ли пользователь. Если нет — перенаправляет на страницу входа.

**Middleware isAdmin:**

Дополнительно проверяет роль пользователя. Доступ к админ-панели разрешен только с ролью admin.

**Примеры маршрутов:**

**1. GET-запросы:**

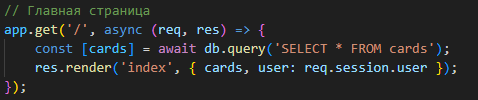
****Получение списка фильмов:

Рисунок 3 Загрузка карточек с фильмами из БД

Получение информации о фильме:

Рисунок 4 Получение данных о фильме

**2. POST-запросы:**

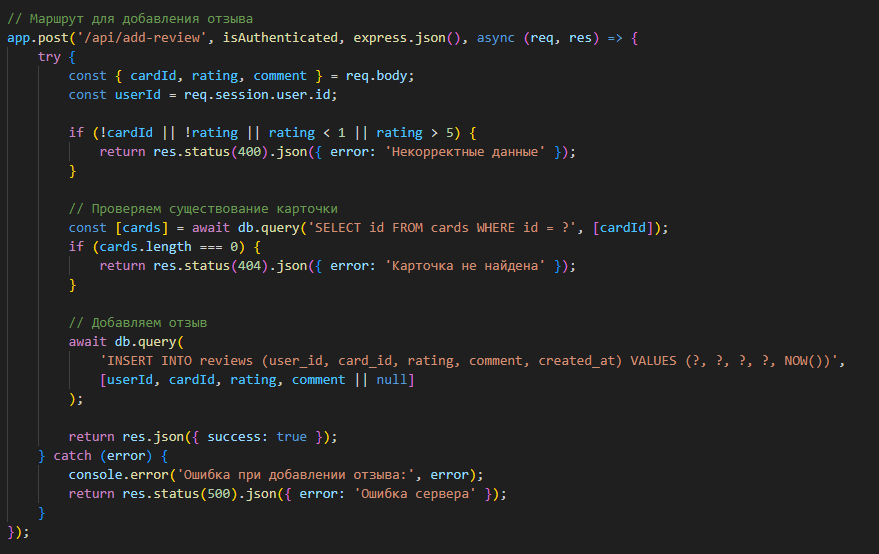
* Добавление отзыва:

Рисунок 5 маршрут добавления отзыва

* Регистрация пользователя:

Рисунок 6 вставка данных регистрации

# **2.2.3 Интеграция фронтенда и бэкенда**

Взаимодействие между фронтендом и бэкендом стримингового сревиса организовано через RESTful-эндпоинты. Фронтенд выполняет запросы к серверу, где они обрабатываются с использованием Express, после чего данные возвращаются в требуемом формате. Для передачи информации применяются два метода: серверный рендеринг и динамическая загрузка данных.

* **GET /**: получение списка фильмов.
* **GET /watch/:id**: получение информации о фильме.
* **POST /api/add-review**: добавление отзыва.
* **GET /search**: поиск фильмов по названию.
* **POST /register**: регистрация пользователя.
* **POST /login**: авторизация пользователя.

# **2.3 Тестирование и оптимизация ресурса**

# **2.3.1 Функциональное тестирование**

**Проверялись следующие сценарии:**

* Отображение списка фильмов и страницы просмотра.
* Поиск фильма по названию.
* Добавление и отображение отзывов.
* Регистрация и авторизация пользователей.
* Доступ к административным функциям (только для пользователей с ролью admin).

# **2.3.2 Оптимизация производительности**

Для повышения производительности были выполнены:

* **Оптимизация SQL-запросов**: использование индексов и избегание избыточных запросов.
* **Кэширование**: часто запрашиваемые данные кэшируются на стороне сервера.
* **Сжатие статических файлов**: минификация CSS и JavaScript.

# **2.3.3 Тестирование на различных устройствах**

Ресурс был протестирован на различных устройствах и браузерах:

* **Устройства**: ПК, планшеты, смартфоны.
* **Браузеры**: Chrome, Firefox, Yandex.

**Результаты**:

* + Интерфейс корректно адаптировался под разные размеры экранов благодаря использованию медиазапросов и Flexbox.
  + Все функции (поиск, добавление комментариев, регистрация) работали стабильно во всех браузерах.

**Заключение по второй главе**

Во второй главе была выполнена практическая реализация стримингового сервиса, включающая следующие ключевые компоненты:

**Проектирование и внедрение базы данных**

* Создана реляционная структура для хранения данных о видеоконтенте
* Реализованы таблицы для учётных записей пользователей
* Организовано хранение и обработка пользовательских отзывов

**Разработка клиентской части**

* Построен адаптивный интерфейс с использованием шаблонизатора EJS
* Реализована интерактивность средствами JavaScript
* Обеспечена кросс-браузерная совместимость

**Серверная реализация**

* Разработана бэкенд-часть на платформе Node.js с фреймворком Express
* Внедрена система аутентификации и разграничения прав доступа
* Настроено безопасное хранение пользовательских данных

**API-интерфейсы**

* Спроектированы RESTful-эндпоинты для клиент-серверного взаимодействия
* Реализована обработка CRUD-операций
* Настроена валидация входящих запросов

Проведённые тесты подтвердили стабильность работы всех компонентов системы. Реализованное решение представляет собой готовый к эксплуатации продукт с возможностью дальнейшего масштабирования функциональности.

**Подробнее о middleware**:

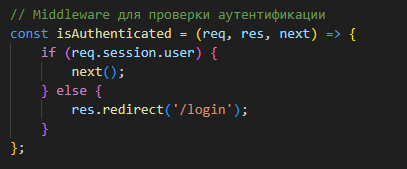
* + isAuthenticated проверяет, авторизован ли пользователь:

Рисунок 7 проверка авторизации

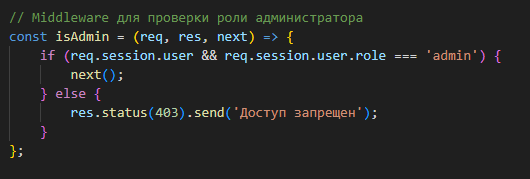
* +  isAdmin проверяет роль пользователя:

Рисунок 8 проверка прав администратора

**Пример SQL-запросов**:

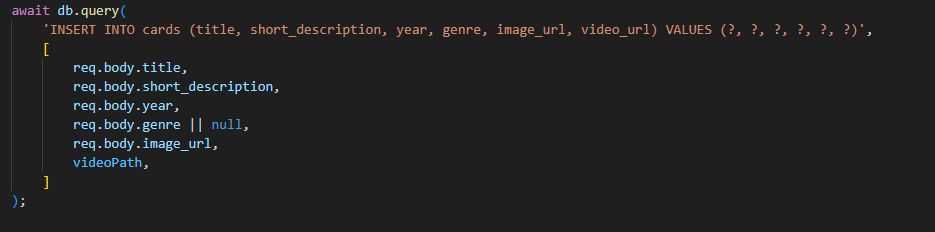
* + Добавление информации о фильме:

Рисунок 9 добавление информации о фильме

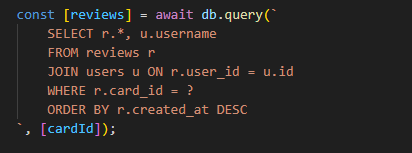
* + Получение отзывов с именем пользователя:

Рисунок 10 загрузка отзывов из БД

**Безопасность**:

* + Хеширование паролей с bcrypt.
  + Использование подготовленных SQL-запросов для предотвращения инъекций.

**Обработка ошибок**:

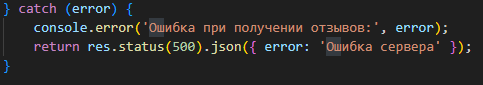
* + Пример обработки ошибок при добавлении отзыва:

Рисунок 11 обработка ошибок

**Поиск фильмов**:

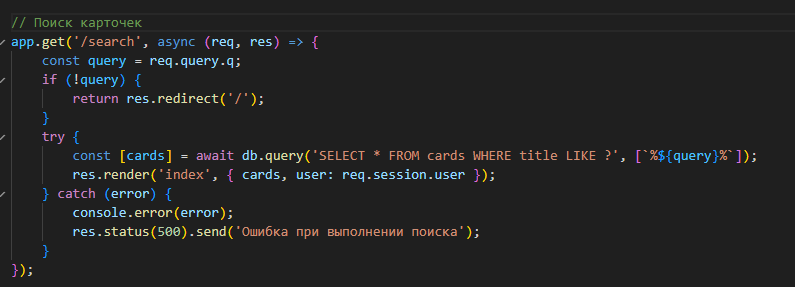
* + Реализация поиска по названию:

Рисунок 12 поиск фильма по названию

# **Заключение**

В ходе выполнения курсовой работы был успешно разработан стриминговый сервис, предоставляющий пользователям комплексное решение для просмотра фильмов, ознакомления с их описанием, рейтингами и отзывами. Проект реализован с применением современных веб-технологий, что обеспечило создание высокопроизводительного и удобного для пользователей ресурса.

**Ключевые достижения проекта:**

**Аналитическая и проектная работа**

* Проведен сравнительный анализ существующих стриминговых платформ с выявлением их сильных и слабых сторон
* Разработана оптимальная структура реляционной базы данных, включающая:
* Таблицы для хранения видеоконтента
* Модули управления пользовательскими аккаунтами
* Систему хранения и обработки отзывов

**Интерфейсная часть**

* Создан адаптивный пользовательский интерфейс с поддержкой различных устройств
* Реализованы интерактивные элементы с использованием современных фронтенд-технологий
* Применен компонентный подход для обеспечения удобства поддержки кода

**Функциональные возможности**

* Разработана система поиска фильмов по названию
* Реализован механизм пользовательских оценок и отзывов
* Создана система регистрации и авторизации

**Работа с данными**

* Организовано надежное хранение информации в СУБД MySQL
* Реализовано эффективное взаимодействие с базой данных через ORM Sequelize

**Тестирование и оптимизация**

* Проведено комплексное тестирование функционала
* Выполнен ряд оптимизационных мероприятий:
* Внедрение системы кэширования
* Оптимизация запросов к базе данных
* Сжатие статических ресурсов

**Преимущества решения:**

* Интуитивно понятный интерфейс с адаптивным дизайном
* Полнофункциональная система работы с видеоконтентом
* Высокая производительность и отказоустойчивость
* Гибкая архитектура, позволяющая масштабировать функционал

**Итоги:**

Разработанный стриминговый сервис полностью соответствует поставленным задачам и демонстрирует эффективное применение современных веб-технологий. Проект обладает значительным потенциалом для дальнейшего развития и может служить основой для создания более сложных медиаплатформ. Работа над проектом позволила существенно расширить практические навыки, включая проектирование баз данных, создание RESTful API, разработку клиентской части и оптимизацию производительности. Полученный опыт создает прочную основу для профессионального роста и реализации более сложных ИТ-проектов в будущем.

# **Список использованных источников**

**Литература**

1. **Агапов, С. И.** Основы веб-разработки: учебное пособие / С. И. Агапов. — М.: Издательство "Инфра-М", 2020. — 320 с.

2. **Брагинский, М. И.** Современные технологии веб-дизайна / М. И. Брагинский, В. В. Витрянский. — М.: Статут, 2019. — 456 с.

3. **Васильев, А. Н.** JavaScript: полное руководство / А. Н. Васильев. — СПб.: Питер, 2021. — 800 с.

4. **Глушаков, С. В.** HTML и CSS: основы создания сайтов / С. В. Глушаков, А. В. Коваль. — М.: АСТ, 2018. — 352 с.

5. **Иванов, П. И.** Базы данных: проектирование и управление / П. И. Иванов. — М.: Юрайт, 2019. — 512 с.

6. **Кузнецов, М. В.** MySQL: справочник по языку запросов / М. В. Кузнецов, И. В. Симдянов. — СПб.: БХВ-Петербург, 2020. — 864 с.

7. **Лаптев, В. В.** Веб-программирование на Python и JavaScript / В. В. Лаптев. — СПб.: Питер, 2021. — 608 с.

8. **Макфарланд, Д.** JavaScript: исчерпывающее руководство / Д. Макфарланд. — М.: Эксмо, 2020. — 960 с.

9. **Петров, В. В.** Разработка веб-приложений с использованием Node.js / В. В. Петров. — М.: ДМК Пресс, 2019. — 400 с.

10. **Смит, Дж.** Веб-дизайн: принципы и практика / Дж. Смит. — М.: Вильямс, 2018. — 496 с.

11. **Флэнаган, Д.** JavaScript: подробное руководство / Д. Флэнаган. — СПб.: Символ-Плюс, 2021. — 1088 с.

12. **Харрис, Э.** PHP и MySQL: разработка веб-приложений / Э. Харрис. — М.: БХВ-Петербург, 2020. — 720 с.

13. **Черников, С. А.** Веб-разработка: от теории к практике / С. А. Черников. — М.: Лань, 2019. — 448 с.

14. **Шапошников, И. В.** Современные фреймворки для веб-разработки / И. В. Шапошников. — М.: ДМК Пресс, 2021. — 384 с.

**Электронные ресурсы**

15. **Документация по JavaScript** [Электронный ресурс]. — Режим доступа: https://developer.mozilla.org/ru/docs/Web/JavaScript. — Загл. с экрана.

16. **Официальный сайт MySQL** [Электронный ресурс]. — Режим доступа: https://www.mysql.com/. — Загл. с экрана.

17. **Node.js: официальная документация** [Электронный ресурс]. — Режим доступа: https://nodejs.org/docs/latest/api/. — Загл. с экрана.

18. **Express.js: официальная документация** [Электронный ресурс]. — Режим доступа: https://expressjs.com/. — Загл. с экрана.

19. **EJS: официальная документация** [Электронный ресурс]. — Режим доступа: https://ejs.co/. — Загл. с экрана.

20. **Руководство по HTML и CSS на MDN Web Docs** [Электронный ресурс]. ­− Режим доступа: https://developer.mozilla.org/ru/docs/Web/HTML и https://developer.mozilla.org/ru/docs/Web/CSS. — Загл. с экрана.