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

ЛЕСОСИБИРСКИЙ ПЕДАГОГИЧЕСКИЙ ИНСТИТУТ –
филиал Сибирского федерального университета

Высшей математики, информатики, экономики и естествознания
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
Храмова Л.Н. Храмова
подпись инициалы, фамилия
«13» 06 2023 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии
код-наименование направления

СОЗДАНИЕ И СОПРОВОЖДЕНИЕ WEB-ПРИЛОЖЕНИЯ ДЛЯ ПРОСМОТРА ФИЛЬМОВ

Руководитель	13.06.23 подпись, дата	доцент, канд. пед. наук должность, ученая степень	Т. В. Захарова инициалы, фамилия
Выпускник	13.06.23 подпись, дата		И. П. Струков инициалы, фамилия
Нормоконтролер	13.06.2023 подпись, дата		Е. В. Киргизова инициалы, фамилия

Лесосибирск 2023

РЕФЕРАТ

Выпускная квалификационная работа по теме «Создание и сопровождение web-приложение для просмотра фильмов» содержит 74 страницы текстового документа, 12 иллюстраций, 4 таблицы, 41 использованный источник.

Цель исследования – теоретически обосновать и разработать web-приложение для просмотра фильмов.

Объект исследования – web-приложение для просмотра фильмов.

Предмет исследования – процесс создания и сопровождения web-приложения для просмотра фильмов.

Для достижения поставленной цели необходимо решить следующие задачи:

– на основе анализа учебной и технической литературы по теме исследования определить понятийно-категориальный аппарат, инструментальные средства разработки web-приложения и характеристики, языки программирования и области их применения;

– рассмотреть структуру и создание пользовательского интерфейса web-приложения для просмотра фильмов;

– разработать web-приложение для просмотра фильмов.

Во время разработки web-приложения для просмотра фильмов изучены основы таких языков программирования, как JavaScript, PHP, а также язык разметки HTML и язык стилей CSS.

В ходе написания выпускной квалификационной работы автор принял участие в конференции и опубликовал статью на тему «Создание и продвижение сайта для просмотра фильмов».

В результате выполнения выпускной квалификационной работы разработано web-приложение для просмотра фильмов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Стриминговые сервисы и онлайн-кинотеатры набирают всё большую популярность по всему миру. Имея хорошую скорость соединения с интернетом с компьютера или телефона, есть возможность мгновенно получить доступ к бесчисленному объему видео файлов, ассортимент которых постоянно обновляется. Также следует отметить, что весь контент доступен в любое удобное для пользователя время в любом объеме и месте. Создание и сопровождение web-приложений для просмотра фильмов стало значимой темой в сфере web-разработки. Такие приложения предоставляют возможность пользователям искать, просматривать, оценивать фильмы и делиться своими рекомендациями с другими пользователями. Они объединяют людей с общими интересами в кино и обеспечивают им удобство и разнообразие вариантов для просмотра фильмов.

Цель исследования – теоретически обосновать и разработать web-приложение для просмотра фильмов.

Объект исследования – web-приложение для просмотра фильмов.

Предмет исследования – процесс создания и сопровождения web-приложения для просмотра фильмов.

Для достижения поставленной цели необходимо решить следующие задачи:

– на основе анализа учебной и технической литературы по теме исследования определить понятийно-категориальный аппарат, инструментальные средства разработки web-приложения и характеристики, языки программирования и области их применения;

– рассмотреть структуру и создание пользовательского интерфейса web-приложения для просмотра фильмов;

– разработать web-приложение для просмотра фильмов.

Методы исследования:

– теоретические: анализ учебной и научно-технической литературы по теме исследование, обобщение и сравнительный анализ;

– эмпирические: проектирование, сопровождение и тестирование программного продукта.

Результаты исследования представлены на научных мероприятиях:

1. Всероссийском молодежном научном форуме «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ–филиал СФУ, 7-12 ноября 2022 г., участие)

2. II Всероссийском конкурсе научных работ «Молодежный научный потенциал» (г. Лесосибирск, ЛПИ – филиал СФУ, 12 апреля 2023 г., участие).

По результатам исследования опубликована статья – Струков И.П. Создание и продвижение сайта для просмотра фильмов / И.П. Струков // Информатика, информационные технологии и экономика: II Всероссийского молодежного научного форума, Лесосибирск 10-15 апреля 2023 года / Сибирский федеральный университет – Красноярск, 2023. С. 78-82.

Структура работы – работа состоит из введения, двух глав, заключения, приложений и списка использованных источников, включающего 41 наименование. Результаты работы представлены в 12 иллюстрациях, 4 таблицах. Общий объем работы – 74 страницы.

1 Теоретические основы создания и сопровождения web-приложений для просмотра фильмов

Создание и сопровождение web-приложений для просмотра фильмов требует знания нескольких теоретических основ:

1. Front-end (Клиентская часть приложения) – это создание пользовательского интерфейса на клиентской стороне web-сайта или приложения. Это всё, что видит пользователь, когда открывает web-страницу, и с чем он взаимодействует: кнопки, баннеры и анимация. Во front-end могут использоваться HTML, CSS и JavaScript. HTML определяет структуру страницы, CSS отвечает за стилизацию элементов, а JavaScript обеспечивает динамическое поведение и взаимодействие на стороне клиента [1].

2. Back-end (Серверная часть приложения) – это разработка бизнес-логики продукта (сайта или веб-приложения). Back-end отвечает за взаимодействие пользователя с внутренними данными, которые потом отображает front-end. Попросту говоря, это то, что скрыто от глаз пользователя и происходит вне его браузера и компьютера. В back-end для web-приложений можно использовать языки программирования, такие как Python, PHP, Ruby или Node.js [4].

3. Базы данных – хранение информации о фильмах, пользовательских данных и других связанных с приложением данных, обеспечивает база данных. Для создания базы данных можно использовать системы управления базами данных, такие как MySQL, PostgreSQL и SQLite [29].

4. API YouTube – это интерфейс, который позволяет встраивать видео, создавать списки воспроизведения и предлагать другие функции YouTube на веб-сайте. Это помогает компаниям предлагать расширенные функции обмена видео на веб-сайтах или web-приложениях без необходимости писать код с нуля [21].

1.1 Анализ существующих web-приложений для просмотра фильмов

Анализ существующих web-приложений при разработке проекта имеет несколько важных причин:

– Изучение лучших практик: Анализ успешных web-приложений в этой отрасли может помочь выявить и изучить лучшие практики и тенденции в дизайне, взаимодействии пользователя, функциональности и других аспектах разработки. Используя эту информацию, как основу для разработки web-приложений можно улучшить пользовательский опыт [10].

– Идентификация сильных и слабых сторон: Изучение существующих web-приложений поможет определить, какие элементы и функциональность работают хорошо, а какие могут быть улучшены. Можно извлечь уроки из опыта других и применить их в собственном проекте для создания более эффективного и привлекательного web-приложения [10].

– Определение конкурентного преимущества: Анализ конкурентов позволяет определить, какие уникальные функции и возможности предлагают другие web-приложения в этой отрасли. Можно использовать эту информацию, чтобы определить собственное преимущество и предложить уникальные возможности или улучшения в своем проекте [10].

– Инспирация для дизайна и визуального оформления: Рассмотрение дизайна и визуального оформления других web-приложений может помочь найти вдохновение и идеи для создания привлекательного и современного дизайна для проекта. Необходимо обратить внимание на стиль, цветовую палитру, компоновку элементов и другие аспекты дизайна, которые можно адаптировать и применить в своем проекте [8].

– Понимание ожиданий пользователей: Анализ других web-приложений позволяет понять, какие функции и возможности пользователи ожидают видеть в web-приложениях этой отрасли. Необходимо учесть эти ожидания и предложить решения, которые лучше соответствуют потребностям аудитории.

В целом, анализ других сайтов предоставляет ценные уроки и идеи, которые помогут создать более эффективный, привлекательный и конкурентоспособный web-проект.

Однако важно помнить, что анализ существующих web-приложений не означает прямого копирования их функций или дизайна. Эту информацию необходимо использовать в качестве источника вдохновения и руководства для создания уникального и привлекательного web-приложения для просмотра фильмов.

В наше время существует множество популярных и широко использующихся российских сайтов для просмотра фильмов. Рассмотрим лучшие отечественные web-приложения для просмотра фильмов и выясним их достоинства и недостатки:

– Кинопоиск.ru – один из самых популярных российских сайтов для просмотра фильмов. Он предлагает бесплатный доступ к фильмам, сериалам и мультфильмам, с возможностью озвучки на русском языке. Кроме того, сайт имеет удобный интерфейс, возможность поиска фильмов по жанрам, списки актеров к каждому фильму, доступ к рейтингам, году выпуска и другим параметрам. Однако некоторые пользователи жалуются на низкое качество видео;

– IVI – российский сайт для просмотра фильмов, который предоставляет доступ к большой библиотеке фильмов и сериалов, в некоторых случаях даже бесплатно. Основным недостатком является реклама, которая может доставлять неудобства, но содержится она в фильмах только при бесплатном просмотре;

– Okko.tv – российский сайт для просмотра фильмов, который предлагает платный доступ к контенту. Он предлагает фильмы, сериалы, мультфильмы и спортивные трансляции в высоком качестве. Сайт имеет удобный интерфейс и доступен на различных устройствах. Кроме того, Okko.tv предлагает возможность скачивания контента для просмотра в офлайн-режиме. Однако, платный доступ может быть дорогим для некоторых пользователей;

– Wink – онлайн-кинотеатр с бесплатным и платным контентом, который помимо прочего предлагает возможность просмотра телевизионных каналов. Однако, пользователи могут столкнуться с низким качеством постеров, а также с нестабильностью работы сайта;

– START – онлайн-кинотеатр, где можно найти множество российских фильмов и сериалов. Однако, также могут быть ограничения доступа в некоторых регионах, а также отсутствие некоторых иностранных фильмов.

Анализ отечественных онлайн-кинотеатров позволил выявить следующие распространенные недостатки со стороны пользователей, как:

– необходимость регистрации и подписки на платные пакеты для получения доступа к полному контенту;

– частые рекламные вставки, которые могут быть раздражающими для пользователей;

– невозможность просмотра контента в высоком качестве без дополнительной оплаты.

– в разрабатываемом web-приложении будут исключены недостатки, а также, по возможности, сохранены главные преимущества перечисленных выше онлайн проектов.

Таким образом, анализ существующих web-приложений позволил изучить лучшие практики, идентифицировать сильные и слабые стороны, определить конкурентное преимущество и найти вдохновение для дизайна. Кроме этого анализ позволил лучше понять ожидания пользователей. В целом, это позволит создать более эффективное, привлекательное и конкурентоспособное web-приложение для просмотра фильмов, отвечающее потребностям и предпочтениям пользователей.

1.2 Сущность понятия «web-приложения»

Web-приложение – это прикладное программное обеспечение, которое работает на веб-сервере, в отличие от компьютерных программ, которые запускаются локально в операционной системе [4].

Web-приложения работают на сервере и обычно взаимодействуют с клиентскими устройствами (компьютеры, смартфоны, планшеты) через интернет. Пользователи могут получать доступ к web-приложениям через URL-адрес, запуская их в web-браузере без необходимости установки специального программного обеспечения [4].

Выделим основные черты web-приложений:

1. Клиент-серверная архитектура: Web-приложение состоит из клиентской и серверной частей. Клиентская часть отвечает за отображение пользовательского интерфейса и взаимодействие с пользователем, в то время как серверная часть обрабатывает запросы клиентов, выполняет бизнес-логику, работает с базами данных и обеспечивает данные и функциональность, которые клиентская часть запрашивает [16].

2. Доступ через web-браузер: Web-приложения доступны через web-браузеры, такие как Yandex Browser, Mozilla Firefox, Safari или Internet Explorer. Это означает, что пользователи могут получить доступ к web-приложению с любого устройства, имеющего доступ к интернету, и в любое удобное время [5].

3. Взаимодействие с пользователем: Web-приложения предоставляют пользователю интерфейс для взаимодействия и выполнения задач, такие как регистрация и вход в систему, просмотр, добавление и редактирование данных, обмен сообщениями и другие операции, в зависимости от целей и требований приложения [28].

4. Гибкость и обновляемость: Web-приложения позволяют легко вносить изменения и обновления. Поскольку пользователи получают доступ к

приложению через интернет, разработчики могут вносить изменения на сервере и обновлять приложение без необходимости обновления клиентской части на каждом устройстве пользователя [33].

5. Масштабируемость: Web-приложения могут масштабироваться для обслуживания большого количества пользователей [16].

Рассмотрим основные отличия между web-приложениями и web-сайтами заключаются в их целях, функциональности и способе взаимодействия с пользователем:

1. Цель: Web-сайт предназначен для предоставления информации, представления контента или продвижения продукта или услуги. Основная цель web-сайта – информационная или маркетинговая. В то же время, web-приложение разработано для выполнения определенных функций и задач, таких как управление данными, выполнение бизнес-процессов или предоставление конкретного сервиса пользователю [11].

2. Функциональность: Web-сайт обычно предоставляет статический контент, такой как текст, изображения и видео, и позволяет пользователям его просматривать. Он может содержать интерактивные элементы, такие как формы для связи с владельцами сайта или комментарии пользователей. С другой стороны, web-приложение предоставляет динамическую функциональность, позволяющую пользователям выполнять различные задачи и операции, такие как создание, редактирование и удаление данных, взаимодействие с базой данных, обработка транзакций и другие действия [11].

3. Взаимодействие с пользователем: Web-сайт обычно ограничивается взаимодействием пользователя с контентом, предоставляемым сайтом. Например, пользователь может просмотреть информацию на страницах сайта или заполнить форму для отправки сообщения. Web-приложение, с другой стороны, обеспечивает более сложное и интерактивное взаимодействие с пользователем. Оно может иметь различные функциональные элементы, такие

как авторизация, учетные записи пользователей, интерфейсы для работы с данными, поиск, фильтрацию, чаты и другие возможности взаимодействия [11].

4. Комплексность: Web-приложения могут требовать разработки сложных бизнес-логик и функций, интеграции с внешними сервисами или API, обработки пользовательских данных, аналитики и многого другого. Это требует более глубокого понимания программирования и разработки web-приложений [11].

5. Динамичность и персонализация: Web-приложения часто имеют возможность динамического обновления содержимого и функциональности в реальном времени. Они могут предоставлять персонализированный контент, основанный на предпочтениях и поведении пользователя. Web-сайты предлагают статичное содержимое, которое не изменяется в зависимости от пользователя [11].

6. Уровень сложности разработки: Разработка web-приложений требует более продвинутых навыков программирования и знания специфических технологий и инструментов.

Таблица 1 – Различия между web-сайтом и web-приложением

Параметр	**Web-приложение**	**Web-сайт**
Основное назначение	Создается для того, чтобы взаимодействовать с пользователем	Предполагает только наличие статей, без какого-либо возможного взаимодействия с пользователем
Взаимодействие с пользователем	Пользователь может производить манипуляции с данными, однако с ограниченным доступом	Пользователь может читать информационный контент, однако не может его никак менять
Аутентификация	Большинство web-приложений требуют произвести аутентификацию, чтобы пользователь мог воспользоваться приложением	Для данных сайтов не требуется обязательная аутентификация. Сайт может лишь иметь пароль которые отсылает администратор ресурса
Задача и сложность	Web-приложение имеет много функций и закрывает множество проблем	Web-сайт отображает лишь статическую страницу, на которой изображена текстовая информация
Изменение проекта	Чтобы внести какие-либо изменения в проект, нужно делать ревью всего кода и после вписывать изменения	Довольно-таки просто вносить изменения, лишь сделав пару изменений в html коде страницы

Краткое описание различий между web-сайтом и web-приложением приведено в таблице 1. Также web-приложение можно классифицировать по различным критериям:

1. По способу работы:

– Статические web-приложения: это простые приложения, которые предоставляют статический контент без динамической функциональности. Они могут быть простыми сайтами с информацией или web-страницами без интерактивности [25].

– Динамические web-приложения: Эти приложения обеспечивают динамическую функциональность, позволяющую пользователям взаимодействовать с контентом и выполнять различные операции. Они часто используют базы данных и обрабатывают пользовательские запросы на сервере [25].

2. По архитектуре:

– Клиент-серверные web-приложения: В таких приложениях клиентская часть (браузер) взаимодействует с серверной частью, которая выполняет обработку запросов, хранение данных и бизнес-логику. Клиент и сервер взаимодействуют посредством протокола HTTP или других протоколов передачи данных [28].

– Одностраничные web-приложения (SPA): В SPA весь код приложения загружается полностью при первоначальной загрузке страницы. Дальнейшая навигация и взаимодействие с приложением происходят без перезагрузки страницы. SPA обычно строятся с использованием фреймворков, таких как Angular, React или Vue.js [25].

3. По предметной области:

– Социальные сети и сообщества: это web-приложения, которые позволяют пользователям обмениваться сообщениями, фотографиями, видео, связываться с друзьями и создавать сообщества с общими интересами [25].

– Электронная коммерция: Web-приложения для электронной коммерции позволяют пользователям покупать и продавать товары или услуги, оформлять заказы, осуществлять платежи и управлять своими аккаунтами [25].

– Управление задачами и проектами: Эти приложения предназначены для планирования, отслеживания и управления задачами, проектами, сроками и командами [25].

4. По технологиям и языкам программирования:

– PHP web-приложения: Разработанные с использованием PHP, такие приложения могут использовать фреймворки, такие как Laravel или Symfony, для обработки запросов и управления бизнес-логикой [28].

– JavaScript web-приложения: Разработанные с использованием JavaScript и его фреймворков, таких как Angular, React или Vue.js. Эти приложения работают на клиентской стороне и взаимодействуют с сервером через API [25].

– Python web-приложения: Разработанные с использованием Python и его фреймворков, таких как Django или Flask. Python широко используется для разработки web-приложений, особенно в научных и аналитических областях [25].

– Ruby web-приложения: Разработанные с использованием Ruby и его фреймворков, таких как Ruby on Rails. Rails позволяет быстро создавать и развертывать web-приложения, следуя принципам соглашения по конфигурации [25].

– Java web-приложения: Разработанные с использованием Java и его фреймворков, таких как Spring или JavaServer Faces (JSF). Java web-приложения могут быть масштабируемыми и надежными [25].

Таким образом, web-приложение – это программное обеспечение, работающее на web-сервере и взаимодействующее с клиентскими устройствами через интернет. Они обладают клиент-серверной архитектурой, доступны через web-браузеры и предоставляют интерфейс для взаимодействия с

пользователями. Web-приложения гибкие и обновляемые, позволяющие вносить изменения на сервере без обновления клиентской части.

1.3 Инструментальные средства разработки web-приложения и их характеристика

Для реализации web-приложения для просмотра фильмов выбраны следующие инструментальные средства разработки как Visual Studio Code, Figma, MySQL и MAMP.

1. Visual Studio Code (VS Code) – это интегрированная среда разработки (IDE), разработанная Microsoft, которая предоставляет широкие возможности для разработки различных типов приложений, включая web-приложения. К основным характеристикам Visual Studio Code относятся:

– Мультиплатформенность: VS Code доступен для операционных систем Windows, macOS и Linux, что позволяет привлечь большую часть разработчиков, предпочитающих разные платформы.

– Расширяемость: Одна из главных особенностей VS Code — это его расширяемость. Он предоставляет широкий набор расширений и плагинов, которые позволяют разработчикам настраивать среду разработки под свои потребности, добавлять поддержку различных языков программирования и инструментов.

– Интеграция с Git: VS Code имеет встроенную интеграцию с системой контроля версий Git, что облегчает работу с репозиториями, отслеживание изменений и управление ветками.

– Отладка и инструменты разработчика: VS Code предоставляет мощные инструменты отладки, возможности автодополнения кода, форматирования и проверки синтаксиса, а также широкий выбор расширений для улучшения процесса разработки.

2. Figma – это онлайн-инструмент для дизайна интерфейсов, который позволяет дизайнерам создавать и совместно работать над дизайнами в реальном времени. Выделим основные характеристики Figma:

– Коллаборативность: Figma разработан с учетом коллаборативной работы. Он позволяет нескольким дизайнерам работать над одним проектом одновременно, обмениваться мнениями, комментариями и видеть изменения в реальном времени.

– Web-базированность: Figma работает в браузере, что делает его доступным на разных операционных системах без необходимости установки дополнительного программного обеспечения.

– Многофункциональность: Figma предлагает широкий спектр инструментов для дизайна интерфейсов, включая возможность создания макетов, прототипов, иконок, компонентов и многого другого. Это позволяет дизайнерам создавать полноценные пользовательские интерфейсы и визуализировать свои идеи.

– Гибкость и адаптивность: Figma позволяет создавать адаптивные дизайны, которые могут быть оптимизированы для различных устройств и разрешений экрана. Это особенно полезно при разработке мобильных приложений и отзывчивых web-дизайнов.

– Разделение на слои и компоненты: Figma предоставляет возможность организации дизайна на слои и компоненты, что облегчает повторное использование элементов интерфейса, создание стилей и обеспечивает единообразие дизайна на протяжении всего проекта.

– Figma обеспечивает легкую интеграцию с различными инструментами и платформами, включая Slack, Jira, Zeplin и многие другие. Это значительно упрощает совместную работу с разработчиками и другими участниками проекта. Благодаря возможности интеграции, вы можете эффективно взаимодействовать и сотрудничать, избегая необходимости переключаться между различными приложениями и платформами.

3. MySQL – популярная система управления базами данных (СУБД), предлагающая надежное и эффективное хранение данных. Вот основные характеристики MySQL:

– Доступность: MySQL является открытым программным обеспечением и доступен бесплатно. Разработчики могут свободно устанавливать, настраивать и использовать MySQL для своих проектов без необходимости приобретения лицензии.

– Производительность: MySQL славится своей высокой производительностью и масштабируемостью. Он способен эффективно обрабатывать большие объемы данных и поддерживать множество одновременных соединений, что делает его идеальным для широкого спектра приложений.

– Поддержка стандартов: MySQL полностью совместим с SQL (Structured Query Language) и поддерживает множество стандартных функций и возможностей. Это позволяет разработчикам эффективно работать с данными, создавать и управлять таблицами, выполнять запросы и многое другое.

– Надежность и безопасность: MySQL обеспечивает надежное хранение данных и имеет механизмы обеспечения целостности данных. Он также предоставляет функциональность безопасности, включая аутентификацию, авторизацию и шифрование данных.

– Многофункциональность: MySQL поддерживает широкий спектр функций, включая триггеры, хранимые процедуры, репликацию, полнотекстовый поиск, географические и пространственные данные и многое другое. Это делает его гибким и мощным инструментом для различных типов приложений и использования [4].

4. MAMP – это интегрированное приложение, которое предоставляет удобную среду для разработки web-сайтов и приложений на базе Apache, MySQL и PHP. Вот основные особенности MAMP:

– Простая установка: MAMP обеспечивает быструю и простую установку, что позволяет разработчикам быстро начать работу над своими проектами в локальной среде.

– Локальная разработка: MAMP предоставляет среду для локальной разработки, позволяя разработчикам создавать и тестировать web-сайты и приложения на своем компьютере, без необходимости подключения к удаленному серверу.

– Apache-сервер: MAMP включает в себя Apache-сервер, который обрабатывает HTTP-запросы и позволяет хостить web-сайты и приложения на локальной машине.

– MySQL-сервер: MAMP поставляется с MySQL-сервером, обеспечивающим надежное и эффективное хранение данных. Это позволяет разработчикам использовать базы данных для своих проектов.

– PHP-интерпретатор: MAMP включает PHP-интерпретатор, позволяющий разработчикам создавать динамические web-сайты и приложения с использованием языка программирования PHP.

– Управление серверами: MAMP предоставляет простой и интуитивно понятный интерфейс для управления серверами Apache и MySQL. Разработчики могут легко запускать, останавливать и настраивать серверы через графический интерфейс приложения.

– Встроенные инструменты: MAMP предлагает ряд встроенных инструментов, таких как phpMyAdmin для удобного управления базами данных MySQL и Xdebug для отладки PHP-кода.

– Гибкость и расширяемость: MAMP позволяет разработчикам добавлять дополнительные компоненты и настраивать конфигурацию серверов в соответствии с их потребностями.

Visual Studio Code предоставляет разработчикам удобную среду для написания кода, отладки и управления проектами. Он поддерживает различные языки программирования, предлагает интеграцию с системой контроля версий

Git и обладает широким набором расширений, которые позволяют настроить среду разработки под конкретные потребности. Благодаря этому разработчики могут эффективно работать над web-приложениями, включая написание серверной и клиентской части кода.

Figma является мощным инструментом для дизайна интерфейсов. Он предоставляет дизайнерам возможность создавать макеты, прототипы, иконки и компоненты, что помогает визуализировать идеи и создавать полноценные пользовательские интерфейсы. Figma также обладает функциональностью для организации дизайна на слои и компоненты, что способствует повторному использованию элементов интерфейса и обеспечивает единообразие дизайна на протяжении всего проекта.

MySQL, в свою очередь, является популярной системой управления базами данных. Он предоставляет надежное хранение данных и мощные возможности для работы с ними. MySQL используется для создания и управления базами данных, выполнения запросов, обеспечения безопасности данных и многое другое. Благодаря интеграции MySQL с Visual Studio Code и поддержке соответствующих расширений, разработчики могут легко взаимодействовать с базой данных MySQL из своей среды разработки, выполнять запросы, отлаживать и тестировать код, связанный с базой данных.

MAMP позволяет разработчикам быстро и легко настраивать и запускать серверы, а также управлять базами данных через простой и интуитивно понятный интерфейс. Он также предлагает удобные инструменты для управления базами данных, такие как phpMyAdmin.

Таким образом, сочетание Visual Studio Code, Figma, MySQL и MAMP позволяет разработчикам эффективно работать над созданием web-приложений. Они предоставляют инструменты, необходимые для разработки, дизайна, управления базами данных и создания качественных пользовательских интерфейсов.

1.4 Языки программирования для разработки web-приложения и область их применения

Языки HTML, CSS, JavaScript и PHP имеют различные области применения при разработке web-приложений.

1. HTML:

– Определение структуры web-страниц: HTML используется для создания основной структуры web-страницы, включая заголовки, параграфы, списки, таблицы и другие элементы [26].

– Разметка контента: HTML позволяет определить различные типы контента, такие как текст, изображения, видео, аудио и другие медиа элементы [27].

– Создание форм: HTML предоставляет возможность создания форм для ввода данных пользователем, таких как текстовые поля, кнопки, флажки, радиокнопки и другие элементы формы [27].

2. CSS:

– Стилизация web-страниц: CSS используется для задания внешнего вида и стилей web-страниц. Он позволяет определить цвета, шрифты, размеры, отступы, границы, фоны и другие аспекты визуального оформления [26].

– Медиазапросы: CSS позволяет создавать адаптивные и отзывчивые web-страницы с помощью медиазапросов. Это позволяет оптимизировать отображение web-страницы на различных устройствах и экранах [27].

3. JavaScript:

– Динамический контент и интерактивность: JavaScript используется для добавления динамического контента и создания интерактивности на web-страницах. Он позволяет создавать анимации, обрабатывать события, выполнять валидацию форм, обмениваться данными с сервером без перезагрузки страницы и многое другое [3].

– Манипуляция DOM: JavaScript предоставляет возможность манипулировать элементами web-страницы с помощью модели объектов документа (DOM). Это позволяет изменять содержимое, стили, атрибуты и другие свойства элементов [30].

4. PHP:

– Обработка форм и данных: PHP широко применяется для обработки данных, отправляемых с web-страниц через формы. Он позволяет получать данные из форм, проверять их, сохранять в базе данных и выполнять другие операции с данными [36].

– Взаимодействие с базами данных: PHP обладает мощными возможностями для работы с базами данных, такими как MySQL. Он позволяет выполнять запросы к базе данных, получать, обновлять и удалять данные, создавать и управлять таблицами и многое другое [6].

– Генерация динамического контента: PHP позволяет генерировать динамический контент на web-страницах. Он может взаимодействовать с базой данных, получать данные и вставлять их на страницу, выполнять вычисления, форматировать данные и многое другое. Благодаря этому можно создавать динамические и персонализированные web-приложения, которые адаптируются к разным пользователям и условиям [4].

Web-разработка с использованием HTML, CSS, JavaScript и PHP предоставляет все необходимые инструменты для создания разнообразных и полнофункциональных web-приложений. Эти языки программирования взаимодействуют между собой и обладают уникальными возможностями, позволяющими создавать привлекательные, интерактивные и полезные web-приложения.

HTML отвечает за структуру и разметку web-страниц, определяет содержимое и расположение элементов на странице. CSS обеспечивает стилизацию и внешний вид web-приложений, позволяя создавать привлекательный дизайн и управлять внешним оформлением элементов.

JavaScript добавляет динамичность и интерактивность на страницы. С его помощью можно создавать анимации, обрабатывать события, взаимодействовать с пользователями и обновлять содержимое страницы без перезагрузки.

PHP предоставляет мощные возможности для обработки данных и взаимодействия с базами данных. Он позволяет обрабатывать данные, полученные из форм, выполнять запросы к базе данных, создавать и обновлять записи и многое другое.

Таким образом, выбранные языки программирования дают разработчикам свободу в создании разнообразных типов web-приложений, от информационных сайтов и блогов до социальных сетей, платформ электронной коммерции и управления данными. Они предоставляют богатые возможности для разработки интерактивных функций и обеспечивают широкий спектр взаимодействия с данными и базами данных.

2 Разработка web-приложения для просмотра фильмов

2.1 Проектирование дизайна интерфейса для web-приложения

В процессе разработки макета главной страницы web-приложения было проведено тщательное исследование существующих web-приложений, предоставляющих аналогичные услуги и функциональность. Анализ существующих web-приложений, представленный в параграфе 1.1, позволил создать удобный и эффективный пользовательский опыт на главной странице.

При изучении других сайтов особое внимание было уделено элементам и функциям, успешно взаимодействующим с посетителями и обеспечивающим высокую эффективность. Целью было включить эти успешные аспекты в макет, чтобы достичь максимальной функциональности и удовлетворить потребности пользователей.

При разработке макета было уделено внимание размещению основных элементов, таких как навигационное меню, важные блоки информации и элементы вызывающих действий. Необходимо было создать логичную и интуитивно понятную структуру, которая позволит пользователям быстро и легко находить необходимую информацию и взаимодействовать с web-приложением [2].

Опираясь на исследование конкурентов и анализ лучших практик web-дизайна, был создан макет главной страницы, объединяющий привлекательный дизайн и функциональность. Также была задача учесть различные аспекты, включая визуальное оформление, типографику и использование цветовых схем, для создания приятного визуального впечатления и усиления восприятия бренда.

Таким образом, разработанный макет главной страницы в соответствии с рисунком 1 представляет собой результат тщательного анализа существующих web-приложений и применения лучших практик web-дизайна. Цель: создать эффективную и привлекательную главную страницу, успешно

взаимодействующую с пользователями и достигающую поставленных целей web-приложения.

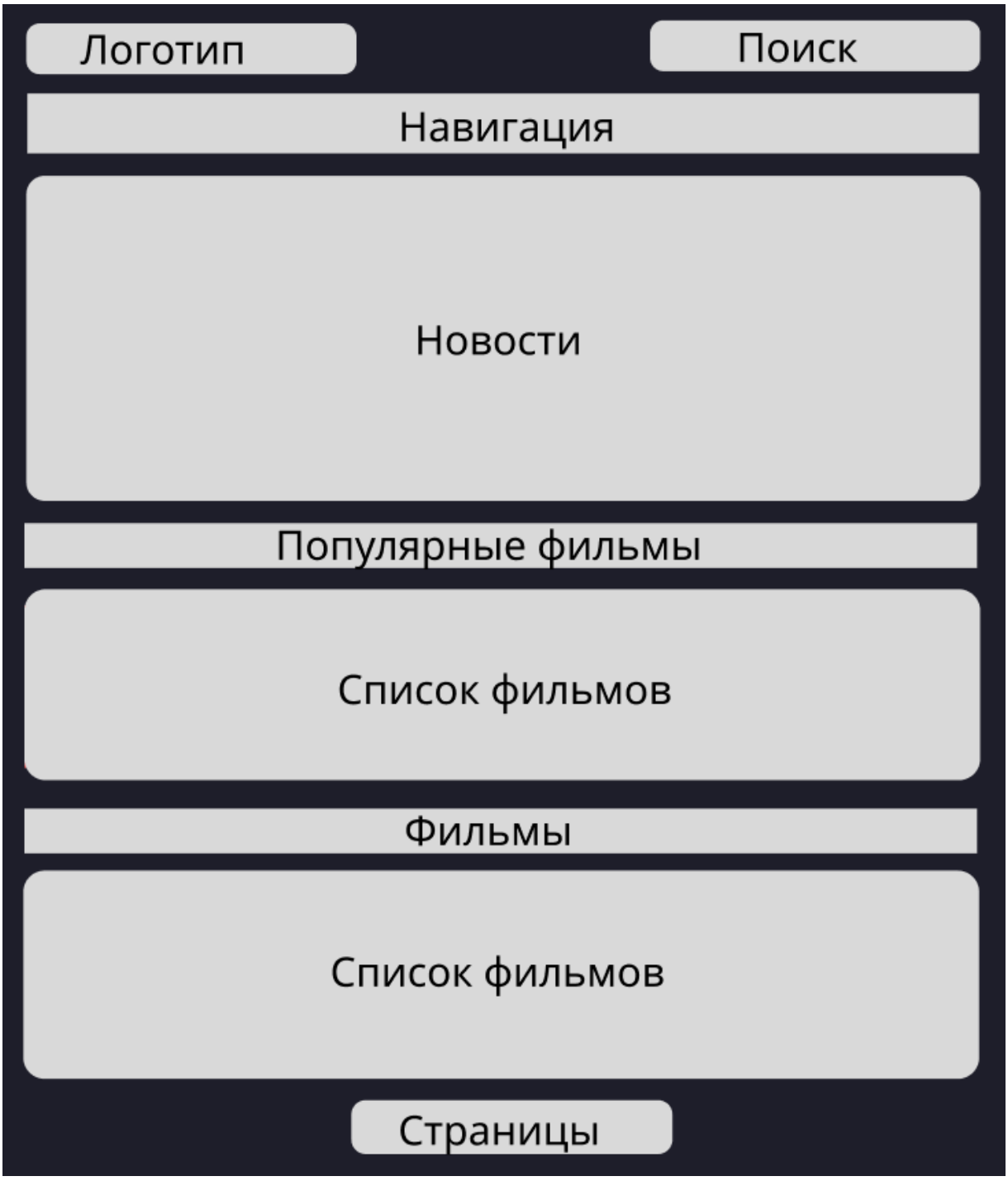

Рисунок 1 – Макет главной страницы

Когда все элементы на главной странице сайта определены, следующий шаг – назначение стилей, которые будут использоваться на странице. Это

включает в себя выбор шрифтов, цветов, иконок и размеров блоков. Для создания гармоничного дизайна главной страницы выбраны основные цвета: фиолетовый, белый и серый. Также выбран основной шрифт - «poppins», который создает чистый и современный вид страницы. Размеры блоков и иконок тщательно подобраны, чтобы достичь наилучшего эффекта и эстетического вида. В целом, все эти элементы и параметры в совокупности создали привлекательный и функциональный дизайн главной страницы, который обеспечивает максимальную эффективность и удобство для пользователей, представленный на рисунке 2.

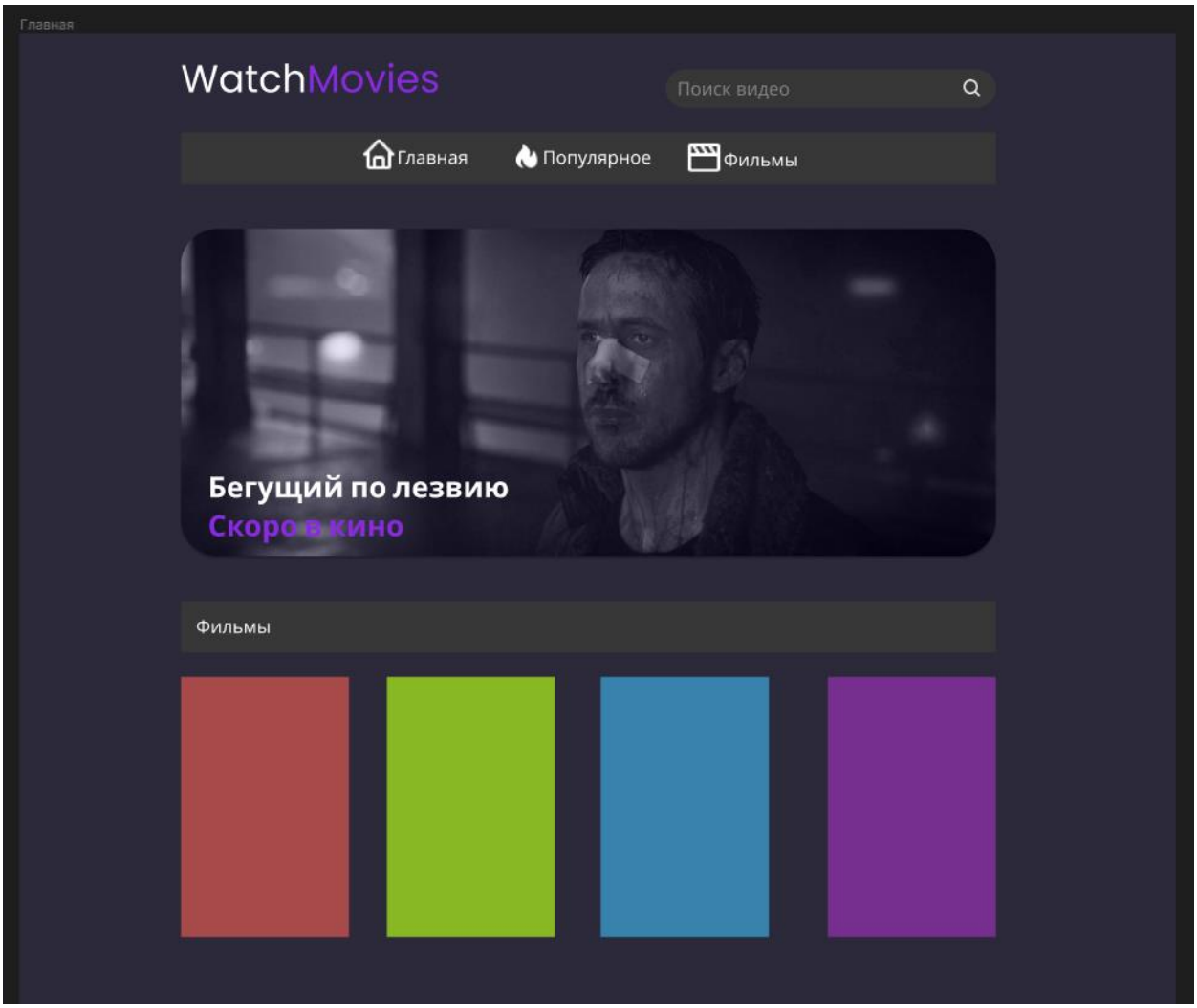

Рисунок 2 – Первоначальный дизайн главной страницы

В процессе разработки макета главной страницы было принято решение создать несколько вариантов дизайна. Цель заключалась в поиске наилучшего варианта, который предоставит пользователям самый удобный и интуитивно понятный интерфейс. При разработке каждого дизайна учитывались особенности сайта и потребности пользователей, с целью создания главной страницы, которая будет максимально удобной и привлекательной. В процессе проектирования были тщательно изучены все преимущества и недостатки каждого дизайна, чтобы выбрать наиболее подходящий вариант, представленный на рисунке 3.

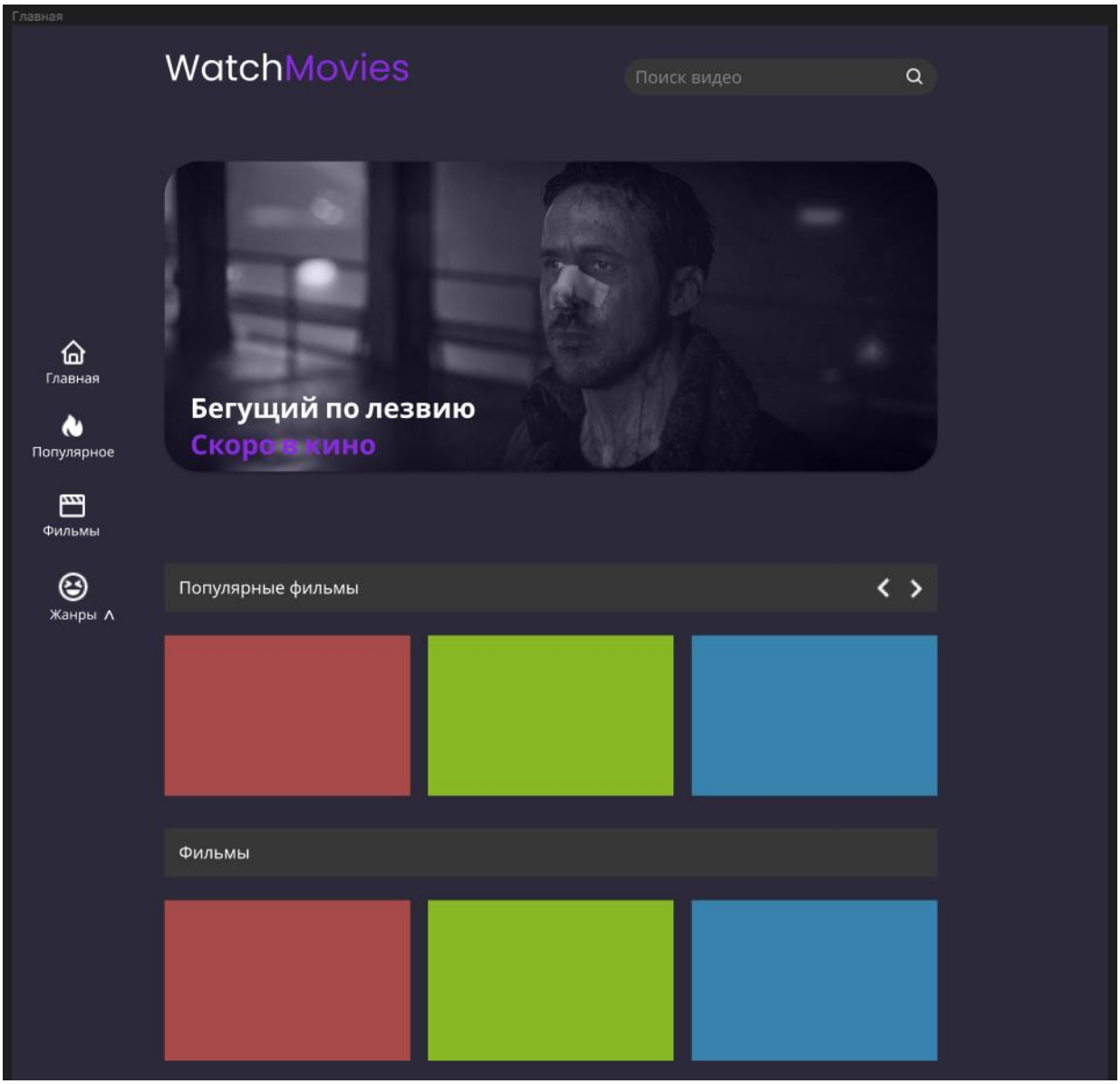

Рисунок 3 – Обновленный дизайн главной страницы

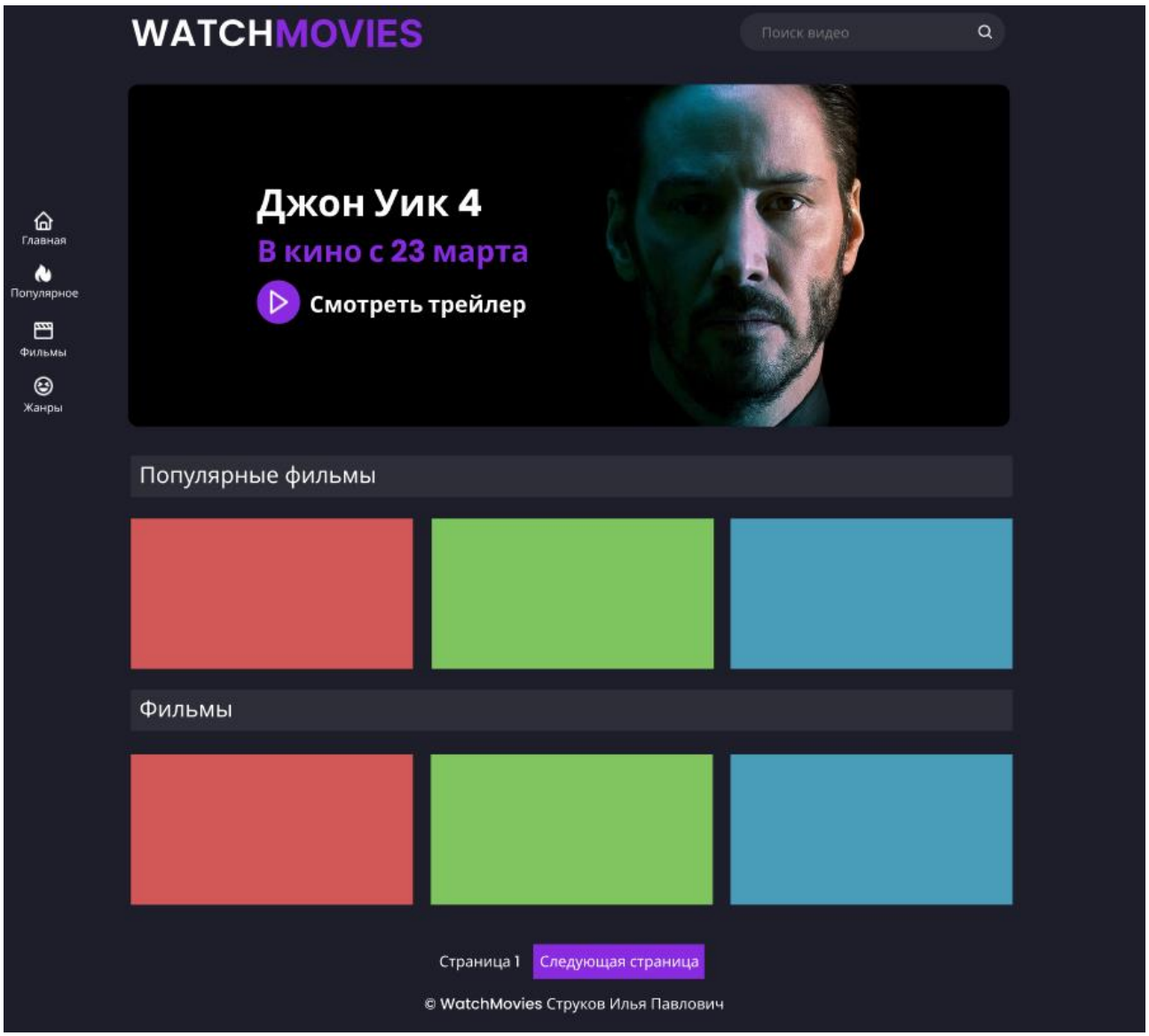

Рисунок 4 – Окончательный вариант страницы

В окончательном варианте макета, представленного на рисунке 4, принято решение дополнить главную страницу еще одной кнопкой для навигации под названием «Жанры», которая размещена в виде раскрывающегося списка. Также панель навигации теперь будет находиться слева, чтобы пользователи мог иметь к ней доступ с любого положения на странице и быстро переходить к нужным разделам сайта. Блоки с фильмами выставлены в альбомном виде.

В целом, эти изменения помогли создать главную страницу сайта, которая максимально удовлетворяет потребности пользователей и обеспечивает простую и интуитивно понятную навигацию.

2.2 Разработка базы данных

Для реализации необходимого функционала была разработана база данных с таблицами фильмов, комментариев.

Таблица 2 – Атрибуты сущности movies

Имя	Тип данных
id	Varchar(255)
moviename	Varchar(255)
thumbnail	Varchar(255)
added_date	datetime
rating	float

В таблице 2 представлены атрибуты сущности «movies» предназначенные для хранения информации о фильмах. Она содержит следующие атрибуты полей:

id (Varchar(255)): это уникальный идентификатор фильма. Он имеет тип данных VARCHAR и длиной до 255 символов, что позволяет хранить идентификаторы в различных форматах (например, числовые или текстовые).

moviename (Varchar(255)): это поле содержит название фильма. Оно имеет тип данных VARCHAR и длиной до 255 символов, что обеспечивает достаточное пространство для хранения полных названий фильмов.

thumbnail (Varchar(255)): это поле хранит путь к миниатюре фильма. Оно имеет тип данных VARCHAR и длиной до 255 символов, что позволяет сохранить ссылку или относительный путь к изображению, используемому в качестве миниатюры фильма.

added_date (datetime): это поле отслеживает дату и время добавления фильма в базу данных. Оно имеет тип данных DATETIME и используется для отслеживания времени добавления фильма.

rating (float): это поле содержит рейтинг фильма. Оно имеет тип данных FLOAT и используется для хранения числового значения рейтинга фильма, который может включать десятичные значения.

Эти поля позволяют хранить информацию о фильмах, включая их уникальный идентификатор, название, путь к превью, дату добавления и рейтинг.

Таблица 3 – Атрибуты сущности popular_movies

Имя	**Тип данных**
id	Varchar(255)
moviename	Varchar(255)
thumbnail	Varchar(255)
added_date	datetime

В таблице 3 представлены атрибуты сущности «popular_movies» предназначенные для хранения информации о популярных фильмах. Она содержит следующие атрибуты полей:

id (Varchar(255)): это уникальный идентификатор популярного фильма. Он имеет тип данных VARCHAR и длиной до 255 символов, что позволяет хранить идентификаторы в различных форматах (например, числовые или текстовые).

moviename (Varchar(255)): это поле содержит название популярного фильма. Оно имеет тип данных VARCHAR и длиной до 255 символов, что обеспечивает достаточное пространство для хранения полных названий фильмов.

thumbnail (Varchar(255)): это поле хранит путь к миниатюре популярного фильма. Оно имеет тип данных VARCHAR и длиной до 255 символов, что позволяет сохранить ссылку или относительный путь к изображению, используемому в качестве миниатюры популярного фильма.

added_date (datetime): это поле отслеживает дату и время добавления популярного фильма в базу данных. Оно имеет тип данных DATETIME и используется для отслеживания времени добавления популярного фильма.

Эти поля позволяют хранить информацию о популярных фильмах, включая их уникальный идентификатор, название, путь к миниатюре и дату добавления. Эта таблица может быть использована для отображения и предоставления пользователю списка популярных фильмов.

Таблица 4 – Атрибуты сущности comments

Имя	Тип данных
id	Int(11)
movie_id	Varchar(255)
username	Varchar(255)
comment	text
rating	float
comment_date	datetime

В таблице 4 представлены атрибуты сущности «comments» предназначенные для хранения информации о комментариях к фильмам. Она содержит следующие атрибуты полей:

id (Int(11)): это уникальный идентификатор комментария. Он имеет тип данных INT и длиной 11 символов.

movie_id (Varchar(255)): это поле содержит идентификатор фильма, к которому относится комментарий. Оно имеет тип данных VARCHAR и длиной до 255 символов, что позволяет хранить идентификаторы фильмов в различных форматах.

username (Varchar(255)): это поле хранит имя пользователя, оставившего комментарий. Оно имеет тип данных VARCHAR и длиной до 255 символов, что обеспечивает достаточное пространство для хранения имен пользователей.

comment (text): это поле содержит текст комментария. Оно имеет тип данных TEXT и может хранить большие объемы текстовой информации.

rating (float): это поле содержит рейтинг, оставленный пользователем для фильма. Оно имеет тип данных FLOAT и используется для хранения числового значения рейтинга, который может включать десятичные значения.

comment_date (datetime): это поле отслеживает дату и время написания комментария. Оно имеет тип данных DATETIME и используется для отслеживания времени создания комментария.

Эти поля позволяют хранить информацию о комментариях к фильмам, включая их уникальный идентификатор, идентификатор фильма, имя пользователя, текст комментария, рейтинг и дату комментирования. Эта таблица может быть использована для отображения комментариев пользователей и учета рейтинга фильмов.

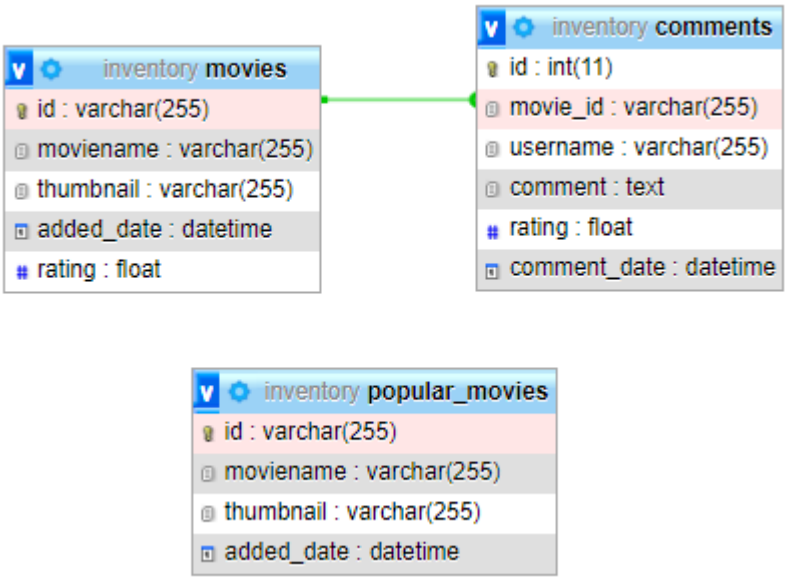

Рисунок 5 – База данных inventory

На рисунке 5 представлено графическое отображение связи таблиц базы данных inventory, с которой будет работать web-приложение.

База данных с фильмами была пополнена лицензионным контентом с YouTube-канала WatchMovies, автор которого позволил воспользоваться им в образовательных целях. Наполнение базы данных фильмами осуществлялось в связке PHP запросов, MySQL и YouTubeAPI.

Код позволяет получить информацию о видео с YouTube канала с помощью YouTube API и сохранить эту информацию в базе данных. Это полезно для создания web-приложения для просмотра фильмов, где видео можно получать автоматически из определенного YouTube канала и отображать их на web-странице.

Описание работы кода из приложения Ж:

1. Конфигурация подключения к базе данных:

– Задаются параметры подключения к базе данных MySQL, такие как имя пользователя, пароль, название базы данных, хост и порт.

2. Подключение к базе данных:

– Создается соединение с базой данных MySQL, используя функции mysqli_init() и mysqli_real_connect().

– Если подключение не удалось, выводится сообщение об ошибке.

3. YouTube канал и API ключ:

– Указывается идентификатор YouTube канала, с которого нужно получить видео (channelId).

– Указывается API ключ, необходимый для доступа к YouTube API (apiKey).

4. Получение списка видео с YouTube канала:

– Формируется URL-адрес запроса к YouTube API с использованием указанного channelId и apiKey.

– Запрашивается содержимое по указанному URL-адресу с помощью функции file_get_contents().

– Полученные данные в формате JSON декодируются в ассоциативный массив.

5. Получение всех страниц результатов:

– Цикл do-while используется для получения всех страниц результатов, если такие есть.

– Результаты каждой страницы добавляются в массив $videos.

6. Обработка полученных данных и добавление в базу данных:

– Цикл foreach перебирает каждое видео в массиве $videos.

– Из каждого видео извлекаются необходимые данные, такие как идентификатор видео (videoId), название видео (videoTitle), URL-адрес обложки видео (thumbnailUrl) и дата публикации (publishedAt).

– Дата публикации преобразуется в нужный формат с помощью функции date().

– Готовится SQL запрос для вставки данных в таблицу «movies» базы данных.

– SQL запрос выполняется с помощью функции mysqli_query().

id	moviename
-OQAG8cNiac	Капкан времени | Slipstream (Фильм 2005, фантастика, боевик, триллер)
0P2dl8ZeBrk	Интервью с Богом (Фильм 2018) Драма, детектив
0sj5PnCmyBw	Невидимый убийца (Toxic) ❗ Высокий рейтинг ❗ детектив, триллер (Фильм 2022)
1a5LJlIThTw	Страна призраков (Фильм 2017) Ужасы, триллер, детектив
1kjxIKMHiMc	Где-то во времени (Фильм 2018) Детектив, мелодрама, фантастика
23Bd0ZMG-kA	Гонка (Фильм 2013) драма, биография, спорт
28ViEOsPexA	Под прицелом / First Date (Кино комедия, криминал, детектив) HD Качество
2ajawo86uaI	Проклятие Плачущей. Возвращение |The Legend Of La Llorona (Фильм 2022, ужасы)
2LmIrvP_IPc	Кукловоды |Puppenspieler |Рейтинг 7.5 (Фильм 2017, драма, приключения, история)
35eFto4MN_0	Техасская резня бензопилой. Кожаное лицо (Фильм 1989) Ужасы, триллер
3kUwJ3pd9_M	Незваный гость| ПРЕМЬЕРА 2020| триллер, детектив
3L1vx80ZWJM	Громкое дело |Judgement (Фильм 2021 триллер, драма, криминал, биография)
5AYiGwYOtd4	Путешествие на край ночи | Journey to the End of the Night (Фильм триллер, драма, криминал)
5hup_kAEBfQ	Спеши любить (Рейтинг 7.8) Walk to Remember (Фильм драма, мелодрама)
5pNuJ5OtdHc	В стране чудес / Birthday Wonderland (Фильм 2019) Аниме, мультфильм, фэнтези, приключения
5tV9bsXcLCA	Перспектива (Фильм 2018) фантастика, триллер, драма
5w7nQTsqXPM	Побег из тюрьмы Мейз | Maze (Фильм 2016, триллер, криминал, детектив)
62j-iEXWX4s	Язычники (Фильм 2017) Драма
6BOtLJYPHns	Карп отмороженный (Фильм 2017) Комедия, драма

Рисунок 6 – База данных с фильмами

На рисунке 6 представлена часть заполненной базы данных таблицы «movies».

Таким образом, созданы все необходимые базы данных, которые сыграют важную роль в наполнении веб-страниц пользователей контентом. Одна из баз данных содержит информацию о фильмах, включая название, жанр, описание,

рейтинг и постеры. Другая база данных может хранить информацию о пользователях, их учетные данные и предпочтения, чтобы предоставлять персонализированный контент и рекомендации. Третья база данных может отслеживать информацию о просмотренных фильмах, добавленных в избранное или оставленных отзывах пользователей. Все эти базы данных позволяют web-приложению эффективно управлять и предоставлять пользователю релевантный и интересный контент.

2.3 Разработка серверной части приложения

На рисунке 7 представлен внешний вид списка фильмов в web-приложении.

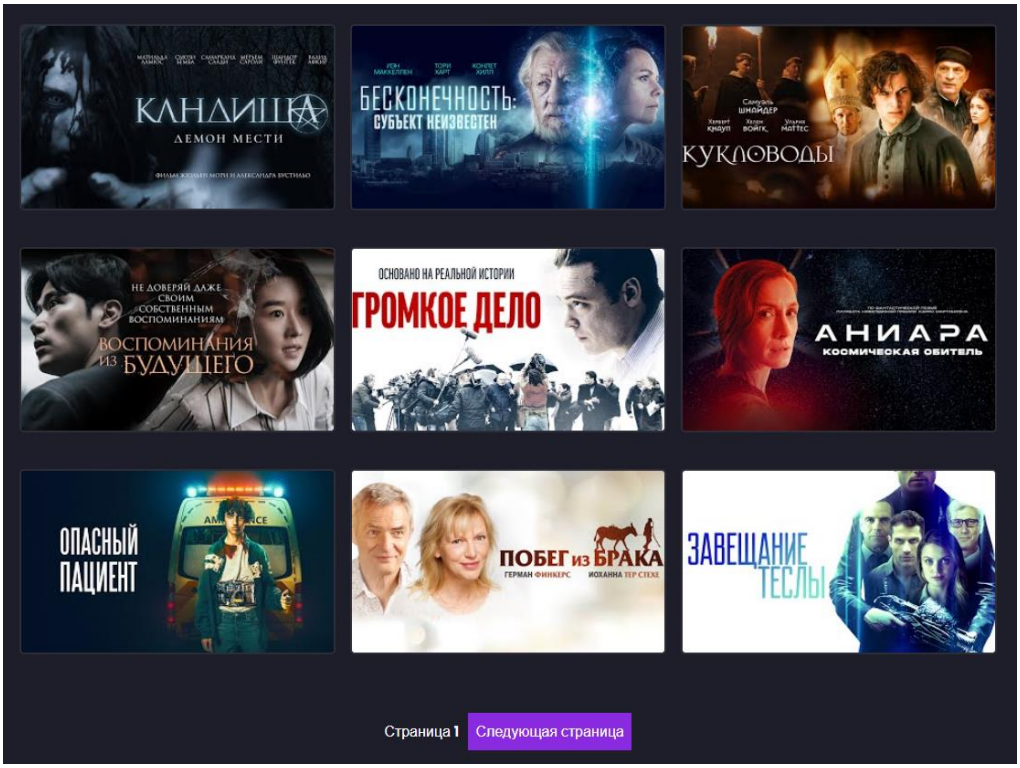

Рисунок 7 – Блоки с фильмами и кнопками создаваемые PHP-кодом

Для отображения списка фильмов, как на рисунке 7, необходимо обратиться к базе данных movies и получить необходимые данные, такие как название, превью и id. Для этого был написан PHP код, который реализует данный функционал.

Код позволяет отобразить видео на web-странице и предоставляет навигацию по страницам для просмотра большого количества видео. Он использует базу данных для хранения информации о видео и обеспечивает динамическую загрузку и отображение видео на web-странице.

Описание работы кода из приложения А:

1. Конфигурация подключения к базе данных:

– Указываются параметры подключения к базе данных MySQL, такие как имя пользователя, пароль, название базы данных, хост и порт.

2. Подключение к базе данных:

– Создается соединение с базой данных MySQL, используя функции mysqli_init() и mysqli_real_connect().

– Если подключение не удалось, выводится сообщение об ошибке и код прекращает свою работу.

3. Получение данных из базы данных:

– Задается количество видео на странице (limit).

– Получается текущая страница (page) из параметра GET запроса. Если параметр не указан, то используется значение 1.

– Рассчитывается смещение (offset) для выборки данных из базы данных.

– Формируется SQL запрос для выборки данных из таблицы «movies» в порядке убывания даты добавления (added_date) с ограничением по количеству и смещением.

– Выполняется SQL запрос с помощью функции mysqli_query().

4. Отображение данных:

– Начинается блок отображения видео с помощью HTML-тега .

– В цикле while перебираются результаты выборки данных из базы данных.

– Для каждого видео извлекаются необходимые данные, такие как идентификатор видео (videoId), название видео (videoTitle) и URL-адрес обложки видео (thumbnailUrl).

– Отображается блок с изображением видео, названием и ссылкой на страницу проигрывания видео. Идентификатор видео передается в параметре GET запроса.

– Закрывается блок отображения видео.

5. Кнопки предыдущей и следующей страницы:

– Рассчитываются значения предыдущей (prevPage) и следующей (nextPage) страницы.

– Отображаются кнопки предыдущей и следующей страницы только если соответствующие страницы существуют.

– Отображается текущая страница.

– Выполняется запрос для подсчета общего количества видео (totalVideos) в базе данных.

– Рассчитывается общее количество страниц (totalPages) на основе количества видео и количества видео на странице.

– Отображается блок пагинации с кнопками предыдущей и следующей страницы (если они доступны) и текущей страницей.

6. Закрытие соединения с базой данных:

– Соединение с базой данных закрывается с помощью функции mysqli_close().

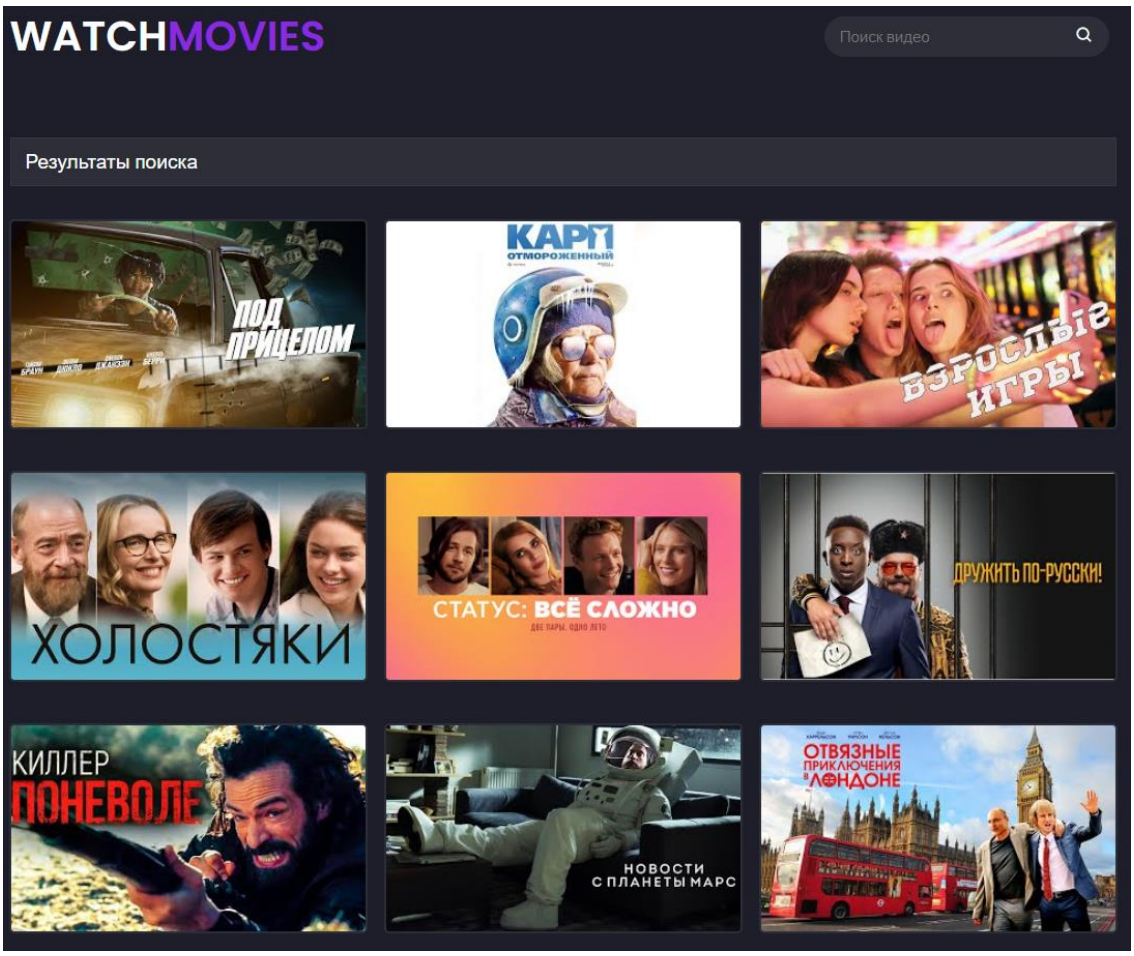

Рисунок 8 – Блоки с фильмами, созданные PHP-кодом по запросу «комедия»

Работа поля поиска основана на том, что при отправке формы, данные из поля ввода поискового запроса, передаются на страницу «search.php» методом GET. На странице «search.php» можно обработать искомый запрос и выполнить соответствующие операции поиска в базе данных и отобразить результаты поиска, как на рисунке 8.

PHP-код со страницы search.php позволяет выполнить поиск видео в базе данных по названию и отобразить результаты на web-странице.\

Вот описание кода из приложения Б.

1. Конфигурация подключения к базе данных:

– Указываются параметры подключения к базе данных MySQL, такие как имя пользователя, пароль, название базы данных, хост и порт.

2. Подключение к базе данных:

– Создается соединение с базой данных MySQL, используя функции mysqli_init() и mysqli_real_connect().

– Если подключение не удалось, выводится сообщение об ошибке и код прекращает свою работу.

3. Получение значения из поля поиска:

– Проверяется наличие значения в параметре GET запроса 'query' с помощью isset().

– Значение поля поиска сохраняется в переменную $searchQuery.

4. Защита от SQL-инъекций:

– Используется функция mysqli_real_escape_string() для экранирования значения $searchQuery и предотвращения возможных SQL-инъекций.

– Выполнение SQL-запроса для поиска видео по названию:

– Формируется SQL запрос для выборки данных из таблицы «movies», где название фильма (moviename) содержит искомую фразу.

– Выполняется SQL запрос с помощью функции mysqli_query().

5. Отображение результатов поиска:

– Начинается блок отображения найденных видео с помощью HTML-тега .

– В цикле while перебираются результаты выборки данных из базы данных.

– Для каждого видео извлекаются необходимые данные, такие как идентификатор видео (videoId), название видео (videoTitle) и URL-адрес обложки видео (thumbnailUrl).

– Отображается блок с изображением видео, названием и ссылкой на страницу проигрывания видео. Идентификатор видео передается в параметре GET запроса.

– Закрывается блок отображения найденных видео.

6. Закрытие соединения с базой данных:

– Соединение с базой данных закрывается с помощью функции mysqli_close().

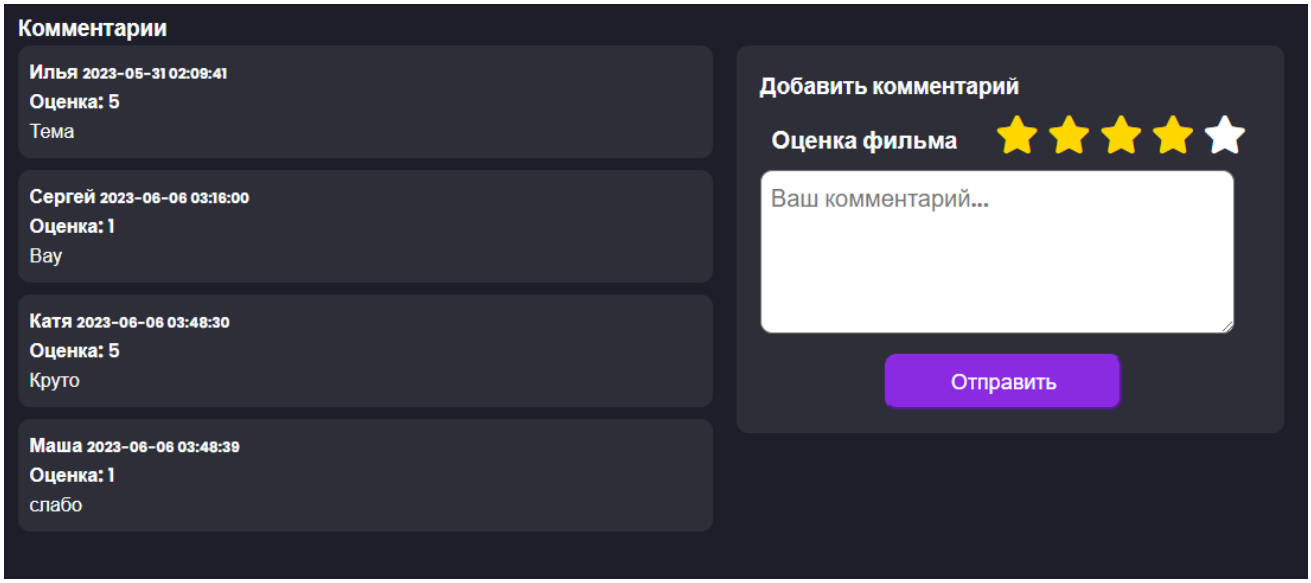

Рисунок 9 – форма с комментариями и добавлением комментария и оценки

Секция с комментариями представленная на рисунке 9, отображается на странице с самим фильмом. Комментарии вместе с оценками привязываются к id видео, для привязки системы оценок к определенному фильму.

Вот описание PHP-кода из приложения В.

1. Отображение существующих комментариев:

– Создается контейнер с классом «existing-comments-section».

– Выводится заголовок «Комментарии».

– Устанавливается соединение с базой данных с помощью указанных параметров.

– Выполняется SQL-запрос для получения комментариев из базы данных для конкретного фильма (определенного по $videoId).

– Если комментарии отсутствуют, выводится сообщение «Здесь пока нет комментариев».

– Если комментарии есть, они выводятся по одному в цикле. Каждый комментарий выводится в блоке с классом «comment». Блок содержит

информацию о имени пользователя ($username), дате комментария ($commentDate), оценке фильма ($rating) и тексте комментария ($comment).

2. Добавление комментариев:

– Создается контейнер с классом «add-comments-section».

– Выводится заголовок «Добавить комментарий».

– Создается форма с методом POST.

– Если в URL передается параметр «videoId» ($_GET['videoId']), то создается скрытое поле ввода типа «hidden» с именем «movie_id» и значением «$videoId».

– Форма содержит поле ввода для оценки фильма с использованием звездочек (radio-кнопки) и поле ввода для комментария (<textarea>).

– Форма также включает кнопку отправки комментария с классом «send».

– Код JavaScript обрабатывает выбор оценки фильма (подсвечивает звезды) при наведении и клике.

3. Обработка отправки комментария:

– Если метод запроса ($_SERVER['REQUEST_METHOD']) равен «POST», то выполняется следующий код.

– Получаются данные из формы: идентификатор фильма ($movieId), имя пользователя ($username), комментарий ($comment) и оценка ($rating).

– Подготавливается SQL-запрос для вставки комментария в таблицу «comments» с использованием подготовленных выражений.

– Выполняется SQL-запрос для добавления комментария.

– Закрывается подготовленное выражение и соединение с базой данных.

– Перенаправление на текущую страницу с обновленными параметрами URL, чтобы предотвратить повторную отправку формы при обновлении страницы.

Для отображения плеера на странице с фильмом используется PHP-код, который проверяет наличие параметра «videoId» в URL-адресе. Если параметр

присутствует, то извлекается его значение и генерируется код iframe для вставки YouTube видео. Сгенерированный код iframe выводится на странице. Если параметр «videoId» отсутствует, то выводится сообщение об ошибке. Сам код представлен в приложении Г.

Таким образом создана серверная часть, которая состоит из PHP-запросов, обеспечивающих взаимодействие с базой данных и добавление фильмов на веб-страницу. С помощью PHP были написаны функции и запросы для получения информации о фильмах из базы данных. При поступлении запроса от клиентской части, PHP код обрабатывает этот запрос, выполняет соответствующие действия, например, выбирает фильмы определенного жанра, и возвращает результаты обратно клиентской части. Кроме того, PHP отвечает за динамическую генерацию контента на веб-странице, вставляя информацию о фильмах из базы данных в нужные элементы HTML-шаблона.

2.4 Разработка клиентской части приложения

Клиентская часть приложения представляет собой интерфейс для взаимодействия с пользователем. С клиентской стороны страница имеет стили как для статичных блоков, которые находятся на странице независимо от серверной части, так и стили для генерируемых блоков с помощью PHP и MySQL. Для написания header главной страницы web-приложения, как на рисунке 10, можно использовать комбинацию языков HTML, CSS и JavaScript. HTML и CSS будут отображать «шапку» (header) сайта с навигацией, логотипом и полем поиска. Также он будет включать элементы для навигации по разделам сайта и выпадающим списком с жанрами.

– Шапка сайта имеет фиксированную позицию и фоновый цвет, определенный как var(--body-color).

– Логотип отображается с помощью элемента <a> с классом «logo». Он имеет размер шрифта 2.5rem и использует шрифт Poppins с жирным

начертанием. Цвет логотипа определен как var(--text-color), а вторая часть логотипа имеет цвет var(--main-color).

– Поле поиска представлено формой с идентификатором «search-form». Оно содержит элемент <input type="search"> для ввода текста поиска и иконку поиска, представленную иконкой из библиотеки boxicons. Форма имеет фоновый цвет var(--container-color) и круглые углы.

– Навигация представлена блоком с классом «navbar». Он содержит «якоря» (href=”#Text”), ссылки на разделы сайта, каждая из которых имеет иконку и заголовок.

– Жанры представлены в виде выпадающего списка. Список имеет элемент с классом «dropdown» и кнопкой с классом «dropdown-btn». При нажатии на кнопку, список с жанрами отображается благодаря JavaScript. Каждый жанр представлен ссылкой.

Код интерфейса header представлен в приложении Д.

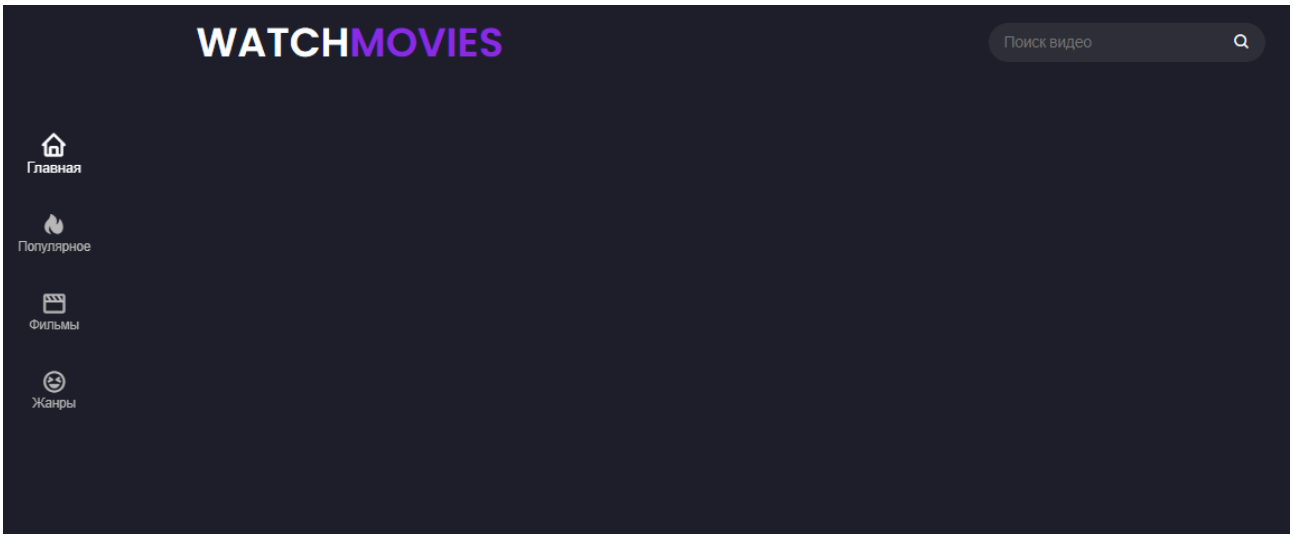

Рисунок 10 – header всех страниц приложения

Для реализации новостной панели, как на рисунке 11, используется язык разметки HTML и стилей CSS. Код используется для отображения раздела на главной странице сайта. Раздел содержит изображение и текст новостной панели, а также динамический видео-трейлер.

HTML код создаст раздел, представленный внутри <section> элемента с классом «home» и классом контейнера. Внутри раздела расположено изображение с путем к файлу изображения. Также присутствует блок с классом «home-text», содержащий заголовок, абзац и ссылку на видео трейлер.

CSS код устанавливает стили для этого раздела. Класс «home» будет определять его позицию, высоту, ширину, отображение, выравнивание, отступы, радиус границы и свойство переполнения. Класс «home-img» будет устанавливать позицию изображения, его ширину, высоту, отступы, z-индекс и свойство object-fit. При наведении на раздел «home» изображение будет увеличиваться с использованием эффекта перехода. Класс «home-tex» задает отступы для текста, а класс «home-title» устанавливать размер шрифта и начертание для заголовка. Класс «watch-btn» будет стилизовать ссылку, задавая свойства отображения, выравнивания и цвета. Класс «.bx» будет определять стили для иконки, включая размеры, фон и радиус границы. При наведении на кнопку ссылки, фон иконки будет изменяться, а цвет текста будет меняться на указанный цвет. Текст внутри ссылки будет иметь определенный цвет, размер шрифта и начертание.

Код новостной панели представлен в приложении Д.

Рисунок 11 – форма новостной панели

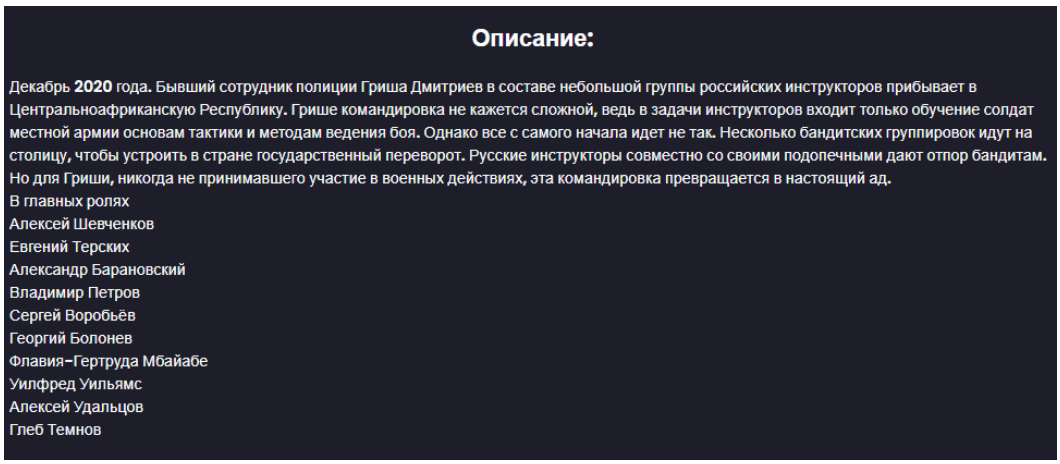

Рисунок 12 – Форма, содержащая описание фильма

Страница для просмотра фильма содержит плеер с самим фильмом и его описанием. В отличие от плеера описание, представленное на рисунке 12, генерируется не с помощью извлечения данных из MySql, а с помощью использования API-ключа YouTube, ID канала и JavaScript, который выполняет следующие действия:

1. Определяется функция обратного вызова onYouTubeApiLoad, которая будет вызвана после загрузки YouTube API. В функции проверяется наличие описания видео и, если оно есть, получается описание и выводится на странице.

2. Определяется функция getVideoIdFromUrl, которая получает значение параметра «videoId» из URL.

3. Определяется функция buildYouTubeApiUrl, которая формирует URL для запроса к YouTube API на основе переданного значения «videoId» и API-ключа.

4. Определяется функция loadVideoDescription, которая загружает описание видео. Она получает значение «videoId» из URL, строит URL для запроса к YouTube API и создает элемент <script> с указанным URL и функцией обратного вызова onYouTubeApiLoad. Этот элемент добавляется в <body> документа.

5. Функция loadVideoDescription вызывается для загрузки описания видео.

Код добавления описания содержится в приложении Е.

Таким образом создана клиентская часть, которая представляет собой весь необходимый пользовательский интерфейс. В этой части приложения реализована интуитивно понятная навигация, позволяющая пользователям легко перемещаться по различным разделам и функциональным возможностям приложения. Был разработан удобный и эффективный механизм поиска, который позволяет пользователям быстро находить интересующие их фильмы по различным критериям.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной выпускной квалификационной работы были изучены и исследованы теоретические основы создания и сопровождения web-приложений для просмотра фильмов.

В первой главе работы проведен анализ существующих web-приложений данной сферы, что позволило выявить их особенности и функциональные возможности. Рассмотрено понятие «web-приложение» и представлены выбранные инструменты разработки и область их применения.

Во второй главе разработано собственное web-приложение для просмотра фильмов. Проведено проектирование дизайна интерфейса с учетом требований пользователей и современных тенденций в дизайне. Разработана база данных, которая обеспечивает хранение и управление информацией о фильмах и пользователях. Серверная и клиентская части приложения были разработаны, что позволило реализовать функциональность web-приложения и обеспечить взаимодействие с базой данных.

В результате работы создано полноценное web-приложение для просмотра фильмов, которое предоставляет пользователям удобный интерфейс, позволяет осуществлять поиск и просмотр фильмов, а также вести данные о просмотрах, оценках и комментариях пользователей. Разработанное приложение отвечает современным требованиям к web-приложениям данной сферы и может быть успешно применено в реальных условиях.

Таким образом, выполнение данной дипломной работы позволило углубить знания о создании и сопровождении web-приложений для просмотра фильмов, а также приобрести практические навыки разработки, проектирования и реализации web-приложений. Разработанное приложение может быть использовано в качестве основы для дальнейших исследований и разработок в данной области.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Аквино, К. Front-end. Клиентская разработка для профессионалов / К. Аквино. – СПБ : Питер, 2017. – 512 с.

2. Андерсон, С. Приманка для пользователей. Создаем привлекательный сайт / С. Андерсон. – М. : Питер, 2021. – 793 с.

3. Браун, Э. Изучаем JavaScript. Руководство по созданию современных веб-сайтов / Э. Браун. – Москва : Альфа-книга, 2017. – 368 с.

4. Веллинг, Л. В. Разработка веб-приложений с помощью PHP и MySQL / Л. В. Веллинг, Л. Т. Томпсон. – Москва : Вильямс, 2020. – 768 с.

5. Вора, П. В. Шаблоны проектирования веб-приложений / П. В. Вора. – Москва : Эксмо, 2011. – 870 с.

6. Губин, М. С. PHP 8. Новинки языка и программы для работы с ним / М. С. Губин, Д. Котеров. – Екатеринбург : Издательские Решения, 2020. – 19 с. – ISBN 9785005138330.

7. Дакетт, Д. Основы веб-программирования с использованием HTML / Джон Дакетт. – Москва : Эксмо, 2020. – 239 с.

8. Дакетт, Д. Разработка и дизайн веб-сайтов / Джон Дакетт. – Москва : Эксмо, 2018. – 250 с.

9. Дронов, В.А. HTML5, CSS3 и web2.0. Разработка современных Web-сайтов / В.А. Дронов. – Спб : БХВ-Петербург, 2011. – 416 с.

10. Емелин, Н.В. Способы оценки и анализа конкурентоспособности / Н. В. Емелин. – М : Форсайт, 2018. – 355 с.

11. Ешану, А. А. Отличия веб-приложения от веб-сайта / А. А. Ешану // Современное программирование. – Нижневартовск : НВГУ, 2020. – С. 41-43.

12. Журавлев, А. Е. Корпоративные информационные системы. Администрирование сетевого домена. Учебное пособие для СПО / А. Е. Журавлев, А. В. Макшанов, Л. Н. Тындыкарь. – Санкт-Петербург : Лань, 2021. – 172 с.

13. Ильин, И. В. Базы данных : учебное пособие / И. В. Ильин, О. Ю. Ильяшенко. – Санкт-Петербург : СПбГПУ, 2020. – 96 с.

14. Карпов, Александр. Создание и продвижение сайтов. Непрофессионал для Непрофессионала / Александр Карпов, Джи Ким. – Москва : ., 2016. – 280 с.

15. Кевин, Янк. PHP и MySQL. От новичка к профессионалу / Янк Кевин. – Москва. : Эксмо, 2018. – 384 с.

16. Кириченко, А. В. Web на практике. CSS, HTML, JavaScript, MySQL, PHP для fullstack-разработчиков / А. В. Кириченко, А. П. Никольский, Е. В. Дубовик. – Санкт-Петербург : Наука и техника, 2021. – 432 с. – ISBN 978-5-94387-271-6.

17. Климов, А. JavaScript на примерах / А. Климов. – СПБ: БХВ-Петербург : Эксмо, 2018. – 336 с.

18. Котеров, Дмитрий. PHP 8 / Дмитрий Котеров, Игорь Симдянов. – Москва : БХВ, 2023. – 992 с. – ISBN 978-5-9775-1692-1.

19. Коэн, Исси. Полный справочник по HTML, CSS и JavaScript / Исси Коэн. – Паблишерз : Эксмо, 2017. – 246 с.

20. Локхарт, Джош. Современный PHP. Новые возможности и передовой опыт / Джош Локхарт. – М. : ДМК Пресс, 2016. – 304 с.

21. Лоре, А. Проектирование веб-API / А. Лоре. – Москва : ДМК-Пресс, 2020. – 440 с. – ISBN 978-5-97060-861-6.

22. Мехта, Ч. MySQL 8 для больших данных / Ч. Мехта. – Москва : ДМК Пресс, 2017. – 228 с.

23. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript / Р. Никсон. – СПБ : Питер, 2010. – 768 с.

24. Прохоренок, Н. А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера, 5 изд / Н.А. Прохоренок. – СПб. : БХВ-Петербург, 2019. – 912 с.

25. Пьюривал, Сэмми. Основы разработки веб-приложений / Сэмми Пьюривал. – Москва : Питер, 2015. – 272 с.

26. Роббинс, Д. Н. Веб-дизайн для начинающих. HTML, CSS, JavaScript и веб-графика Подробнее: https://www.labirint.ru/books/797578/ / Д. Н. Роббинс. – Санкт-Петербург : BHV, 2021. – 956 с.

27. Робсон, Элизабет. Изучаем HTML, XHTML и CSS / Элизабет Робсон, Эрик Фримен. – Питер : -, 2019. – 720 с.

28. Сафронов, М. Разработка веб-приложений с помощью Yii 2 и PHP / М. Сафронов. – Москва : ДМК Пресс, 2015. – 377 с.

29. Ткаченко, В. MySQL по максимуму / В. Ткаченко, П. Зайцев, Ш. Бэрон. – М. : Питер, 2018. – 864 с.

30. Фримен, Э. Изучаем программирование на JavaScript / Э. Фримен, Э. Робсон. – Москва : Эксмо, 2007. – 2010 с.

31. Хоффман, Э. Безопасность веб-приложений. Разведка, защита, нападение / Э. Хоффман. – Санкт-Петербург : Питер, 2020. – 336 с. – ISBN 978-5-4461-1786-4.

32. Хэррон, Д. Node.js Разработка серверных веб-приложений на JavaScript / Д. Хэррон. – Москва : ДМК, 2014. – 144 с.

33. Эспозито, Д. Разработка современных веб-приложений. Анализ предметных областей и технологий / Д. Эспозито – Москва : Вильямс, 2017. – 464 с. – ISBN 978-5-9908910-3-6.

34. GitHub // phpmyadmin : сайт. – URL: https://github.com/phpmyadmin/phpmyadmin/wiki (дата обращения: 15.05.2023)

35. HTML5BOOK : сайт. – URL: https://html5book.ru/ (дата обращения: 03.04.2023)

36. PHP : официальный сайт. – URL: https://www.php.net/ (дата обращения: 15.06.2023)

37. w33schools // PHP tutorial: сайт. – URL: https://www.w3schools.com/php/default.asp (дата обращения: 08.05.2023)

38. w3schools // CSS Tutorial : сайт. – URL: https://www.w3schools.com/css/default.asp (дата обращения: 22.04.2023)

39. w3schools // HTML Tutorial : сайт. – URL: https://www.w3schools.com/html/default.asp (дата обращения: 22.04.2023)

40. w3schools // JavaScript Tutorial : сайт. – URL: https://www.w3schools.com/js/default.asp (дата обращения: 27.04.2023)

41. w3schools // MySQL Tutorial : сайт. – URL: https://www.w3schools.com/mysql/default.asp (дата обращения: 15.05.2023)

ПРИЛОЖЕНИЕ А

Скрипт для наполнения главной страницы фильмами

```
<?php
    // Конфигурация подключения к базе данных
    $user = 'root';
    $password = 'root';
    $db = 'inventory';
    $host = 'localhost';
    $port = 3306;
    // Подключение к базе данных
    $link = mysqli_init();
    $success = mysqli_real_connect(
       $link,
       $host,
       $user,
       $password,
       $db,
       $port
    );
    if (!$success) {
       die("Ошибка подключения к базе данных: " . mysqli_connect_error());
    }
    // Получение данных из базы данных
    $limit = 21; // Количество видео на странице
    $page = isset($_GET['page']) ? $_GET['page'] : 1; // Текущая страница
    $offset = ($page - 1) * $limit; // Смещение
    $sql = "SELECT * FROM movies ORDER BY added_date DESC LIMIT $limit
OFFSET $offset";
    $result = mysqli_query($link, $sql);
```

```
// Отображение данных
echo '<div class="thumbsnail">';
while ($row = mysqli_fetch_assoc($result)) {
   $videoId = $row['id'];
   $videoTitle = $row['moviename'];
   $thumbnailUrl = $row['thumbnail'];
   echo '<div class="video-block">';
   echo '<a href="play-page.php?videoId=' . $videoId . '">';
   echo '<img src="' . $thumbnailUrl . '" alt="' . $videoTitle . '">';
   echo '<p>' . $videoTitle . '</p>';
   echo '</a>';
   echo '</div>';
}
echo '</div>';
// Кнопки предыдущая страница и следующая страница
$prevPage = $page - 1;
$nextPage = $page + 1;
echo '<div class="pagination">';
if ($prevPage >= 1) {
   echo '<a href="index.php?page=' . $prevPage . '" style="
   margin-right: 10px;
   background-color: #8A2BE2;
   color: #FFFFFF;
   padding: 8px;
">Предыдущая страница</a>';
}
echo '<span>Страница ' . $page . '</span>';
$sqlCount = "SELECT COUNT(*) as total FROM movies";
$countResult = mysqli_query($link, $sqlCount);
```

```
$rowCount = mysqli_fetch_assoc($countResult);
$totalVideos = $rowCount['total'];
$totalPages = ceil($totalVideos / $limit);
if ($nextPage <= $totalPages) {
   echo '<a href="index.php?page=' . $nextPage . '" style="
   margin-left: 10px;
   background-color: #8A2BE2;
   color: #FFFFFF;
   padding: 8px;
">Следующая страница</a>';
}
echo '</div>';
// Закрытие соединения с базой данных
mysqli_close($link);
?>
```

ПРИЛОЖЕНИЕ Б

Скрипт для выполнения поискового запроса

```
<?php
// Конфигурация подключения к базе данных
$user = 'root';
$password = 'root';
$db = 'inventory';
$host = 'localhost';
$port = 3306;
// Подключение к базе данных
$link = mysqli_init();
$success = mysqli_real_connect(
   $link,
   $host,
   $user,
   $password,
   $db,
   $port
);
if (!$success) {
   die("Ошибка подключения к базе данных: " . mysqli_connect_error());
}
// Получение значения из поля поиска
$searchQuery = isset($_GET['query']) ? $_GET['query'] : '';
// Защита от SQL-инъекций (рекомендуется использовать подготовленные выражения)
$searchQuery = mysqli_real_escape_string($link, $searchQuery);
// Выполнение SQL-запроса для поиска видео по названию
$sql = "SELECT * FROM movies WHERE moviename LIKE '%$searchQuery%'";
```

```
$result = mysqli_query($link, $sql);

// Отображение результатов поиска
echo '<div class="thumbsnail">';
while ($row = mysqli_fetch_assoc($result)) {
    $videoId = $row['id'];
    $videoTitle = $row['moviename'];
    $thumbnailUrl = $row['thumbnail'];
    echo '<div class="video-block">';
    echo '<a href="play-page.php?videoId=' . $videoId . '">';
    echo '<img src="' . $thumbnailUrl . '" alt="' . $videoTitle . '">';
    echo '<p>' . $videoTitle . '</p>';
    echo '</a>';
    echo '</div>';
}
echo '</div>';
// Закрытие соединения с базой данных
mysqli_close($link);
?>
```

ПРИЛОЖЕНИЕ В

Скрипты для вывода и добавления оценок и комментариев

```
<section class="comment-container container">
    <div class="existing-comments-section" style="width: 598px; margin-right: 20px; margin-top: -29px;">
    <h3>Комментарии</h3>
    <?php
    // Подключение к базе данных
    $user = 'root';
    $password = 'root';
    $db = 'inventory';
    $host = 'localhost';
    $port = 3306;
    $link = mysqli_init();
    $success = mysqli_real_connect(
        $link,
        $host,
        $user,
        $password,
        $db,
        $port
    );
    if (!$success) {
        die('Ошибка подключения к базе данных: ' . mysqli_connect_error());
    }
    // Получение комментариев из базы данных
    $sql = "SELECT username, rating, comment, comment_date FROM comments WHERE movie_id = ?";
```

```
    $stmt = mysqli_prepare($link, $sql);
    mysqli_stmt_bind_param($stmt, "s", $videoId);
    mysqli_stmt_execute($stmt);
    $result = mysqli_stmt_get_result($stmt);
    // Проверка наличия комментариев
    if (mysqli_num_rows($result) === 0) {
        echo '<p>Здесь пока нет комментариев :(</p>';
    } else {
        // Отображение комментариев
        while ($row = mysqli_fetch_assoc($result)) {
            $username = $row['username'];
            $rating = $row['rating'];
            $comment = $row['comment'];
            $commentDate = $row['comment_date'];
            echo '<div class="comment" style="
            width: auto;
            border-radius: 10px;
            background-color: var(--container-color);
            margin-bottom: 10px;
            padding:10px;
            overflow: hidden;
            word-wrap: break-word;
            ">';
            echo '<p><b>' . $username . '</b> <b style="font-size:12px;">' .
$commentDate . '</b></p>';
            echo '<p><strong>Оценка:</strong> ' . $rating . '</p>';
            echo '<p style="border">' . $comment . '</p>';
            echo '</div>';
        }
```

```
    }
    // Закрытие подготовленного выражения и соединения
    mysqli_stmt_close($stmt);
    mysqli_close($link);
    ?>
</div>
<div class="add-comments-section" style="background-color: var(--container-color);
width: 470px; padding: 20px; border-radius: 10px;">
    <h3>Добавить комментарий</h3>
    <form action="" method="POST">
        <?php
        if (isset($_GET['videoId'])) {
            $videoId = $_GET['videoId'];
            echo '<input type="hidden" name="movie_id" value="' . $videoId . '">';
        }
        ?>
        <div class="star-wrapper">
        <b style="margin-right: 30px; margin-top:6px; font-size: 20px">Оценка
фильма</b>
            <div>
            <input type="radio" name="rating" id="star-1" value="1" required>
            <label for="star-1" class="fas fa-star"></label>
            <input type="radio" name="rating" id="star-2" value="2" required>
            <label for="star-2" class="fas fa-star"></label>
            <input type="radio" name="rating" id="star-3" value="3" required>
            <label for="star-3" class="fas fa-star"></label>
            <input type="radio" name="rating" id="star-4" value="4" required>
            <label for="star-4" class="fas fa-star"></label>
            <input type="radio" name="rating" id="star-5" value="5" required>
```

```
        <label for="star-5" class="fas fa-star"></label>
        </div>
    </div>
    <textarea style="margin-top:10px; width: 400px; padding:8px; border-radius:10px; font-size: 20px;" name="comment" placeholder="Ваш комментарий..." id="comment" rows="4" required></textarea>
    <button class="send" style="background-color: #8A2BE2; color: #FFFFFF; padding: 8px; cursor:pointer; margin-left: 25%; margin-top:10px; width: 200px; font-size: 18px; border-color: var(--main-color); border-radius:10px" type="submit">Отправить</button>
</form>
<style>
    .star-wrapper {
        display: flex;
        justify-content: center;
        margin-top: 10px;
    }
    .star-wrapper input[type="radio"] {
        display: none;
    }
    .star-wrapper label {
        font-size: 2em;
        color: #fff;
        margin: 0 2px;
        cursor: pointer;
        transition: color 0.5s;
    }
    .star-wrapper input[type="radio"]:checked ~ label,
    .star-wrapper:hover label {
```

```
        color: gold;
        transition: color 0.5s;
    }
</style>
<script>
    const starWrapper = document.querySelector('.star-wrapper');
    const stars = document.querySelectorAll('.star-wrapper label');
    let selectedStarIndex = -1;
    starWrapper.addEventListener('mouseleave', () => {
        if (selectedStarIndex === -1) {
            stars.forEach((star, index) => {
                star.style.color = '#fff';
            });
        }
    });
    stars.forEach((star, index) => {
        star.addEventListener('mouseover', () => {
            if (selectedStarIndex === -1) {
                stars.forEach((star, i) => {
                    if (i <= index) {
                        star.style.color = 'gold';
                    } else {
                        star.style.color = '#fff';
                    }
                });
            }
        });
        star.addEventListener('click', () => {
            selectedStarIndex = index;
```

```
            stars.forEach((star, i) => {
                if (i <= index) {
                    star.style.color = 'gold';
                } else {
                    star.style.color = '#fff';
                }
            });
        });
    });
    </script>
</div>
<?php
$user = 'root';
$password = 'root';
$db = 'inventory';
$host = 'localhost';
$port = 3306;
$link = mysqli_init();
$success = mysqli_real_connect(
    $link,
    $host,
    $user,
    $password,
    $db,
    $port
);
if (!$success) {
    die('Ошибка подключения к базе данных: ' . mysqli_connect_error());
}
```

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Получение данных из формы
    $movieId = $_POST['movie_id'];
    $username = $_POST['username'];
    $comment = $_POST['comment'];
    $rating = $_POST['rating'];
    // Подготовка SQL-запроса для вставки комментария в таблицу comments
    $sql = "INSERT INTO comments (movie_id, username, comment, rating)
VALUES (?, ?, ?, ?)";
    $stmt = mysqli_prepare($link, $sql);
    mysqli_stmt_bind_param($stmt, "sssi", $movieId, $username, $comment, $rating);
    // Выполнение SQL-запроса
    $result = mysqli_stmt_execute($stmt);
    // Закрытие подготовленного выражения и соединения
    mysqli_stmt_close($stmt);
    mysqli_close($link);
    // Подготовка videoId для сохранения в ссылке
    $currentURL = $_SERVER['PHP_SELF'];
    $urlParts = parse_url($currentURL);
    parse_str($urlParts['query'], $queryParams);
    $queryParams['videoId'] = $movieId;
    $newQuery = http_build_query($queryParams);
    // Формирование нового URL с обновленными параметрами
    $newURL = $urlParts['path'] . '?' . $newQuery;
    // Перенаправление на новый URL
    header("Location: " . $newURL);
    exit();
}
?>
```

</section>

ПРИЛОЖЕНИЕ Г

Скрипт для добавления плеера на страницу

```
<?php
    if (isset($_GET['videoId'])) {
        $videoId = $_GET['videoId'];
        // Генерация кода iframe с YouTube видео
        $iframeCode = '<iframe width="1068" height="600"
src="https://www.youtube.com/embed/' . $videoId . '" frameborder="0"
allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"
allowfullscreen></iframe>';
        // Отображение iframe
        echo $iframeCode;
    } else {
        // Если videoId не получен, отображаем сообщение об ошибке
        echo "Ошибка: Не удалось получить идентификатор видео.";
    }
    ?>
```

ПРИЛОЖЕНИЕ Д

Код системы навигации по страницам сайта

```
<header>
    <!--Блок навигации-->
    <div class="nav container">
      <!--Логотип-->
      <a href="index.php" class="logo"  id="home">
        Watch<span>Movies</span>
      </a>
      <!--Поле поиска-->
      <div class="search-box">
      <form id="search-form" action="search.php" method="GET">
        <input type="search" name="query" id="search-input"
placeholder="Поиск видео" required="">
        <i class="bx bx-search" id="search-icon"></i>
      </form>
      </div>
        <script>
          // Получаем ссылки на элементы формы и иконку поиска
          const searchForm = document.getElementById('search-form');
          const searchInput = document.getElementById('search-input');
          const searchIcon = document.getElementById('search-icon');
          // Обработчик события клика по иконке поиска
          searchIcon.addEventListener('click', function(event) {
            // Отменяем стандартное поведение формы
            event.preventDefault();
            // Проверяем, что поле поиска не пустое
            if (searchInput.value.trim() !== '') {
            // Отправляем форму
```

```
                searchForm.submit();
                }
              });
          </script>
        <!--Пользователь-->
        <div style="position: relative;">
          <div class="user-icon">
            <i class='bx bx-user'></i>
          </div>
          <div class="login-form" id="loginForm">
            <h2 id="welcomeMessage">Авторизация</h2>
            <i class="close-icon bx bx-x" onclick="hideLoginForm()"></i>
            <form id="loginForm" onsubmit="login(event)">
              <input type="text" name="login" placeholder=" Введите логин..."
required><br>
              <input type="password" name="password" placeholder=" Введите
пароль..." required><br>
              <button class="auth-button" type="submit">Войти</button>
              <button class="auth-button" type="button"
onclick="showRegisterForm()">Регистрация</button>
            </form>
            <p id="loginError" class="error-message"></p>
          </div>
          <div class="register-form">
            <h2>Регистрация</h2>
            <i class="close-icon bx bx-x" onclick="hideRegisterForm()"></i>
            <form action="php/process_form.php" method="POST">
              <input type="text" name="name" placeholder=" Ваше Имя..."
required><br>
```

```
                <input type="text" name="login" placeholder=" Введите логин..." required><br>
                <input type="password" name="password" placeholder=" Введите пароль..." required><br>
                <button class="auth-button" type="submit">Зарегистрироваться</button>
            </form>
        </div>
    <!--Навигация-->
    <div class="navbar">
        <a href="#home" class="nav-link nav-active">
            <i class='bx bx-home'></i>
            <span class="nav-link-title">Главная</span>
        </a>
        <a href="#popular" class="nav-link">
            <i class='bx bxs-hot' ></i>
            <span class="nav-link-title">Популярное</span>
        </a>
        <a href="#movies" class="nav-link">
            <i class='bx bx-movie' ></i>
            <span class="nav-link-title">Фильмы</span>
        </a>
        <div class="dropdown">
            <a class="dropdown-btn" onclick="toggleDropdown()">
              <i class='bx bx-laugh'></i>
              <span class="dropdown-btn-title">Жанры</span>
            </a>
            <div class="dropdown-content" id="genre-list">
                <a href="search.php?query=Комедия">Комедии</a>
```

```
                    <a href="search.php?query=Боевик">Боевики</a>
                    <a href="search.php?query=Драма">Драмы</a>
                    <a href="search.php?query=Триллер">Триллеры</a>
                    <a href="search.php?query=Ужасы">Ужасы</a>
                    <a href="search.php?query=Фантастика">Фантастика</a>
                    <a href="search.php?query=Мультфильм">Мультфильмы</a>
                    <a href="search.php?query=Фэнтези">Фэнтези</a>
                    <a href="search.php?query=Детектив">Детективы</a>
                    <a href="search.php?query=Драма">Драмы</a>
                    <a href="search.php?query=Криминал">Криминал</a>
                  </div>
                </div>
              </div>
            </div>
        </div>
    </header>
    <!--Главная-->
    <section class="home container">
        <!--Изображение на главной-->
        <img src="img/home-background.png" alt="" class="home-img">
        <!--Текст на главной-->
        <div class="home-text">
            <h1 class="home-title">Джон Уик 4</h1>
            <p>В кино с 23 марта</p>
            <a href="https://www.youtube.com/watch?v=edKjGI5W3AM" class="watch-
btn">
                <i class='bx bx-right-arrow' ></i>
                <span>Смотреть трейлер</span>
            </a>
```

```
    </div>
</section>
```

ПРИЛОЖЕНИЕ Е

Скрипт, добавляющий описание к фильму

```
<script>
  // Функция обратного вызова, вызываемая после загрузки YouTube API
  function onYouTubeApiLoad(response) {
    // Проверяем наличие описания видео
    if (response.items.length > 0) {
      // Получаем описание видео
      const videoDescription = response.items[0].snippet.description;
      // Отображаем описание на странице
      const descriptionContainer = document.getElementById('video-description');
      descriptionContainer.innerHTML = videoDescription;
    }
  }
  // Получение значения videoId из URL
  function getVideoIdFromUrl() {
    const urlParams = new URLSearchParams(window.location.search);
    return urlParams.get('videoId');
  }
  // Формирование URL для запроса к YouTube API
  function buildYouTubeApiUrl(videoId) {
    const apiKey = 'AIzaSyCxTGDwu19Zcj6MUfFJ_AZjh8og6wnrsJ8';
```

```
    return
`https://www.googleapis.com/youtube/v3/videos?part=snippet&id=${videoId}&key=
${apiKey}`;
  }
  // Загрузка описания видео
  function loadVideoDescription() {
    const videoId = getVideoIdFromUrl();
    if (videoId) {
      const apiUrl = buildYouTubeApiUrl(videoId);
      const scriptElement = document.createElement('script');
      scriptElement.src = apiUrl + '&callback=onYouTubeApiLoad';
      document.body.appendChild(scriptElement);
    }
  }
  // Вызов функции загрузки описания видео
  loadVideoDescription();
</script>
```

ПРИЛОЖЕНИЕ Ж

Скрипт для наполнения главной страницы фильмами

```
<?php
// Конфигурация подключения к базе данных
$user = 'root';
$password = 'root';
$db = 'inventory';
$host = 'localhost';
$port = 3306;
// Подключение к базе данных
$link = mysqli_init();
$success = mysqli_real_connect(
   $link,
   $host,
   $user,
   $password,
   $db,
   $port
);
if (!$success) {
   die("Ошибка подключения к базе данных: " . mysqli_connect_error());
}
// YouTube канал и API ключ
$channelId = 'UC3N4p9X5DjPCH184X-izRwA';
$apiKey = 'AIzaSyCxMzNMvJYKZQrWbwzQyJSWFfOqoi42Dw0';
// Получение списка видео с YouTube канала
$apiUrl =
"https://www.googleapis.com/youtube/v3/search?part=snippet&channelId=$channelI
d&maxResults=50&key=$apiKey";
```

```
$videos = [];

// Получение всех страниц результатов
do {
  $response = file_get_contents($apiUrl);
  $data = json_decode($response, true);

  if ($data && isset($data['items'])) {
    foreach ($data['items'] as $video) {
      $videos[] = $video;
    }
  }
  if (isset($data['nextPageToken'])) {
    $nextPageToken = $data['nextPageToken'];
    $apiUrl =
"https://www.googleapis.com/youtube/v3/search?part=snippet&channelId=$channelI
d&maxResults=50&pageToken=$nextPageToken&key=$apiKey";
  } else {
    $nextPageToken = null;
  }
} while ($nextPageToken !== null);
// Обработка полученных данных и добавление в базу данных
foreach ($videos as $video) {
  $videoId = $video['id']['videoId'];
  $videoTitle = $video['snippet']['title'];
  $thumbnailUrl = $video['snippet']['thumbnails']['medium']['url']; // Исправлено
здесь
  $publishedAt = $video['snippet']['publishedAt'];
  // Преобразование формата даты
```

```
    $formattedDate = date('Y-m-d H:i:s', strtotime($publishedAt));

    // Подготовка SQL запроса
    $sql = "INSERT INTO movies (id, moviename, thumbnail, added_date) VALUES
('$videoId', '$videoTitle', '$thumbnailUrl', '$formattedDate')";
    // Выполнение SQL запроса
    if (mysqli_query($link, $sql)) {
        echo "Видео успешно добавлено в базу данных.";
    } else {
        echo "Ошибка при выполнении запроса: " . mysqli_error($link);
    }
}
```