

Реферат

Пояснительная записка дипломного проекта выполнена в объеме 80 страниц, 49 рисунков, 8 таблиц, 17 листингов, 21 использованного источника, 2 приложений.

ОНЛАЙН-СЕРВИС, АНАЛОГ, ЦЕЛЕВАЯ АУДИТОРИЯ, БАЗА ДАННЫХ, ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ, ПРОТОТИП, ДИЗАЙН-МАКЕТ, ЛОГОТИП, ВЕРСТКА, JAVASCRIPT, VUE.JS, PHP, LARAVEL, ТЕСТИРОВАНИЕ

Объектом разработки является онлайн-сервис «Moviq».

Целью разработки онлайн-сервиса было привлечение новых пользователей, облегчение поиска фильмов по своим предпочтениям и настроению.

К задачам, решаемым в ходе выполнения проекта, относятся: постановка целей, анализ предметной области и целевой аудитории, проектирование структуры онлайн-сервиса, выбор стиля дизайна, типографики и цветовой схемы, верстка сервиса, программная реализация сервиса, тестирование, обоснование экономической целесообразности проекта.

Объем графического материала:

- логическая схема базы данных – 1 лист А3;
- структурная схема онлайн-сервиса – 1 лист А3;
- диаграмма вариантов использования – 1 лист А3;
- прототипы страниц сервиса – 1 лист А3;
- дизайн-макеты основных страниц сервиса – 1 лист А3;
- элементы дизайна – 1 лист А3.

					БГТУ 00.00.ПЗ						
Изм	Лист	№ документа	Подп.	Дата							
Разраб.	Помоз				Реферат				Лит.	Лист	Листов
Пров.	Осоко								у	1	1
Консульт.									74319008, 2024		
Н. контр.	Осоко										
Утв.	Романенко										

Abstract

Explanatory letter of degree project is performed in the content of 80 pages, 49 figures, 8 tables, 17 listings, 21 literary sources, 2 annexes.

ONLINE SERVICE, ANALOGUE, TARGET AUDIENCE, DATABASE, USE-CASE DIAGRAM, PROTOTYPE, DESIGN LAYOUT, LOGO, GRAPHICAL LAYOUT, JAVASCRIPT, VUE.JS, PHP, LARAVEL, TESTING

The object of development is the online service «Moviq».

The purpose of the online service development was to attract new users, make it easier to find movies according to their preferences and mood.

The tasks to be solved during the project include: setting goals, analyzing the subject area and target audience, designing the structure of the online service, choosing the design style, typography and color scheme, layout of the online service, program implementation of the online service and the admin panel, testing, justification of the economic feasibility of the project.

Content of graphic matter:

- logical schema of the database – 1 sheet of A3;
- block diagram of the online service – 1 sheet A3;
- use case diagram – 1 sheet of A3;
- service page prototypes – 1 sheet of A3;
- layouts of the main pages of the service – 1 sheet of A3;
- elements of design – 1 sheet of A3.

					БГТУ 00.00.ПЗ						
Изм	Лист	№ документа	Подп.	Дата							
Разраб.	Помоз				Abstract				Лит.	Лист	Листов
Пров.	Осоко								у	1	1
Консульт.									74319008, 2024		
Н. контр.	Осоко										
Утв.	Романенко										

Содержание

	стр.
Введение.....	7
1. Аналитический обзор	8
1.1. Анализ предметной области.....	8
1.2. Анализ аналогичных решений	8
1.2.1. Анализ онлайн-сервиса «PairMovie».....	9
1.2.2. Анализ онлайн-сервиса «Film-Like»	11
1.2.3. Анализ онлайн-сервиса «filmpro»	13
1.2.4. Анализ онлайн-сервиса «Кинопоиск».....	15
1.3. Выбор стиля дизайна, методов и технологий.....	18
1.4. Выводы по разделу	19
2 Проектирование интерфейсов и структуры данных онлайн-сервиса	20
2.1 Цели и задачи проекта	20
2.2 Целевая аудитория	20
2.3 Разработка персонажей и создание пользовательских сценариев	21
2.3.1 Ключевой персонаж №1	21
2.3.2 Ключевой персонаж №2	22
2.3.3 Ключевой персонаж №3	23
2.4 Проектирование взаимодействия пользователей	24
2.5 Информационная структура проекта	25
2.6 Построение интерактивного прототипа интерфейса	25
2.7 Проектирование базы данных	29
2.8 Алгоритмы основных процессов	30
2.9 Выводы по разделу	32
3 Дизайн онлайн-сервиса	33
3.1 Выбор и обоснование стиля	33
3.2 Цветовое решение проекта	34
3.3 Типографика.....	34
3.4 Создание логотипа	36
3.5 Разработка дизайна страниц.....	37
3.6 Выводы по разделу	39
4 Реализация проекта.....	40
4.1 Создание и обработка мультимедийного контента.....	40
4.2 Верстка и стилевое оформление проекта	41
4.3 Программная реализация визуальных эффектов и элементов дизайна	42
4.4 Реализация функциональных возможностей.....	45
4.4.1 Реализация функционала авторизации	45
4.4.2 Реализация расширенной формы регистрации.....	46

					БГТУ 00.00.ПЗ			
Изм	Лист	№ документа	Подп.	Дата	Содержание			
Разраб.	Помоз							
Пров.	Осоко							
Консульт.								
Н. контр.	Осоко							
Утв.	Романенко				74319008, 2024			

4.4.3 Реализация личного кабинета пользователя	46
4.4.4 Реализация административной панели	47
4.4.5 Реализация панели модератора	48
4.4.6 Реализация модуля добавления описания фильмов	49
4.4.7 Реализация модуля актуализации базы данных	49
4.4.8 Расчет рейтинга фильмов.....	51
4.4.9 Реализация формирования рекомендаций для пользователей.....	52
4.5 Выводы по разделу	53
5 Особенности использования и продвижения проекта	54
5.1 Руководство пользователя	54
5.2 Тестирование.....	57
5.2.1 Тестирование навигации.....	57
5.2.2 Корректная работа с формами и валидация.....	58
5.2.3 Кроссбраузерное тестирование	59
5.2.4 Тестирование адаптивности	60
5.2.5 UX/UI тестирование	62
5.3 Особенности продвижения проекта.....	64
5.4 Выводы по разделу	65
6 Экономический раздел	66
6.1 Общая характеристика разрабатываемого программного средства	66
6.2 Исходные данные для проведения расчётов и маркетинговый анализ.....	77
6.3 Обоснование цены программного средства	68
6.3.1 Затраты рабочего времени на разработку программного средства	68
6.3.2 Расчёт основной заработной платы.....	69
6.3.3 Расчёт дополнительной заработной платы	70
6.3.4 Расчёт отчислений в Фонд социальной защиты населения по обязательному страхованию.....	71
6.3.5 Расчёт суммы прочих прямых затрат.....	71
6.3.6 Расчёт суммы накладных расходов.....	72
6.3.7 Сумма расходов на разработку программного средства.....	72
6.3.8 Расходы на реализацию	73
6.3.9 Расчёт полной себестоимости	73
6.3.10 Определение цены, оценка эффективности	77
6.4 Расчёт стоимости хостинга и доменного имени	75
6.5 Выводы по разделу	75
Заключение	77
Список использованных источников.....	78
Перечень графического иллюстративного материала	80
ПРИЛОЖЕНИЕ А. Полная структура разработанного проекта	81
ПРИЛОЖЕНИЕ Б. Основной программный код сервиса	82

Введение

В современном мире цифровые технологии развиваются стремительными темпами. За несколько десятилетий интернет охватил практически все сферы человеческой деятельности, стал важной частью повседневной жизни миллионов людей и продолжает интегрироваться во все аспекты нашей повседневности.

Для качественного онлайн-сервиса важны не только содержание и функциональность, но и визуальное оформление. Веб-дизайн – это этап веб-разработки, подразумевающий проектирование и визуализацию макета сайта или веб-приложения, т. е. его прототипа, который в будущем будет перенесен в интернет. В процессе его создания планируется структура и навигация, komponуются блоки и контент, подробно прорабатывается внешний вид отдельных элементов [1].

Веб-дизайн оказывает значительное влияние на эмоциональные впечатления пользователя и часто определяет, останется ли он на сайте. Важную роль играют эстетика и гармоничность восприятия страниц: слишком яркие иллюстрации могут отвлекать от основного контента, а отсутствие дизайнерских элементов может создать впечатление пустоты. Качественный и оригинальный веб-дизайн особенно важен в условиях высокой конкуренции.

Сегодня все больше владельцев бизнеса осознают, что успешное ведение дел невозможно без использования интернет-ресурсов. Онлайн-сервис по подбору фильмов не является исключением. Некоторые владельцы акцентируют внимание на привлечении пользователей через социальные сети и контекстную рекламу, другие уделяют больше времени созданию удобного и функционального сервиса.

Интернет-сервисы, связанные с предоставлением информации о кинематографии стали неотъемлемой частью жизни многих людей. В условиях современного изобилия контента пользователи часто сталкиваются с трудностями в выборе фильма, который соответствовал бы их вкусу и предпочтениям. Разработка онлайн-сервиса, который может помочь пользователям в выборе фильмов, становится актуальной задачей.

Таким образом, целью данного дипломного проекта является создание онлайн-сервиса по подбору фильмов, соответствующего современным технологиям и сочетающего удобство и комфорт использования. Этот сервис будет помогать пользователям находить фильмы по их предпочтениям, предоставляя информацию о каждом фильме. Интерфейс сервиса должен быть интуитивно понятен, а контент – легко восприниматься.

					БГТУ 00.00.ПЗ									
Изм	Лист	№ документа	Подп.	Дата										
Разраб.	Помоз				Введение					Лит.	Лист	Листов		
Пров.	Осоко									у		1	1	
Консульт.										74319008, 2024				
Н. контр.	Осоко													
Утв.	Романенко													

1 Аналитический обзор

1.1 Анализ предметной области

Интернет стал неотъемлемой частью нашей жизни. Он предлагает бесконечное количество возможностей и может стать как полезным инструментом, так и средством развлечения [2]. Кино – один из самых популярных способов провести время. Однако, не всегда легко выбрать фильм, который подойдет под настроение и предпочтения. В этом контексте, создание эффективного онлайн-сервиса, способного предложить пользователям рекомендации и обеспечить удобный поиск и выбор фильмов, становится стратегически важной задачей.

Создание онлайн-сервиса «Moviq» – это возможность предоставить удобный и доступный инструмент для всех любителей кино, помогающий им в поиске фильмов без лишних затрат времени. Благодаря современному и интуитивно понятному интерфейсу, а также широким возможностям, «Moviq» станет надежным помощником в мире кинематографии для всех его пользователей.

Темой дипломного проекта является создание современного онлайн-сервиса по подбору фильмов «Moviq», который предоставит пользователям возможность быстро и удобно находить фильмы по своим предпочтениям и настроению, делая процесс выбора кино более простым, быстрым и удобным.

1.2 Анализ аналогичных решений

Для того, чтобы разработать актуальный и востребованный дизайн и функционал сайта, необходимо рассмотреть аналогичные веб-сайты, после чего сделать выводы по каждому из разделов, выделив их достоинства и недостатки.

Анализ аналогов будет проводиться по следующим критериям:

- цель сайта;
- мобильная версия;
- главная страница;
- дизайн;
- навигация;
- функционал;
- актуальность контента.

Для проведения анализа были выбраны несколько сервисов, предназначенных для предоставления пользователю возможности выбирать кино в соответствии с их интересами. Каждый из этих ресурсов предлагает свои особенные алгоритмы подбора фильмов. Важно было проанализировать, как выбранные сервисы удовлетворяют потребности своих пользователей, а также какие преимущества и недостатки они имеют.

					БГТУ 01.00.ПЗ									
Изм	Лист	№ документа	Подп.	Дата										
Разраб.		Помоз			Аналитический обзор				Лит.		Лист		Листов	
Пров.		Осоко							у		1		12	
Консульт.									74319008, 2024					
Н. контр.		Осоко												
Утв.		Романенко												

1.2.1 Анализ онлайн-сервиса «PairMovie»

«PairMovie» [3] – это онлайн-сервис, разработанный для любителей кино, которые ценят время и хотят посмотреть фильм в паре или группой. Целевая аудитория – молодые люди в возрасте от 18 до 35 лет, проживающие в городских и пригородных районах. Аудитория обладает средним и выше среднего достатком. Интересуются разнообразными жанрами кино и ценят возможность находить качественные фильмы для просмотра в паре или группе друзей.

Главная страница сайта изображена на рисунке 1.1.

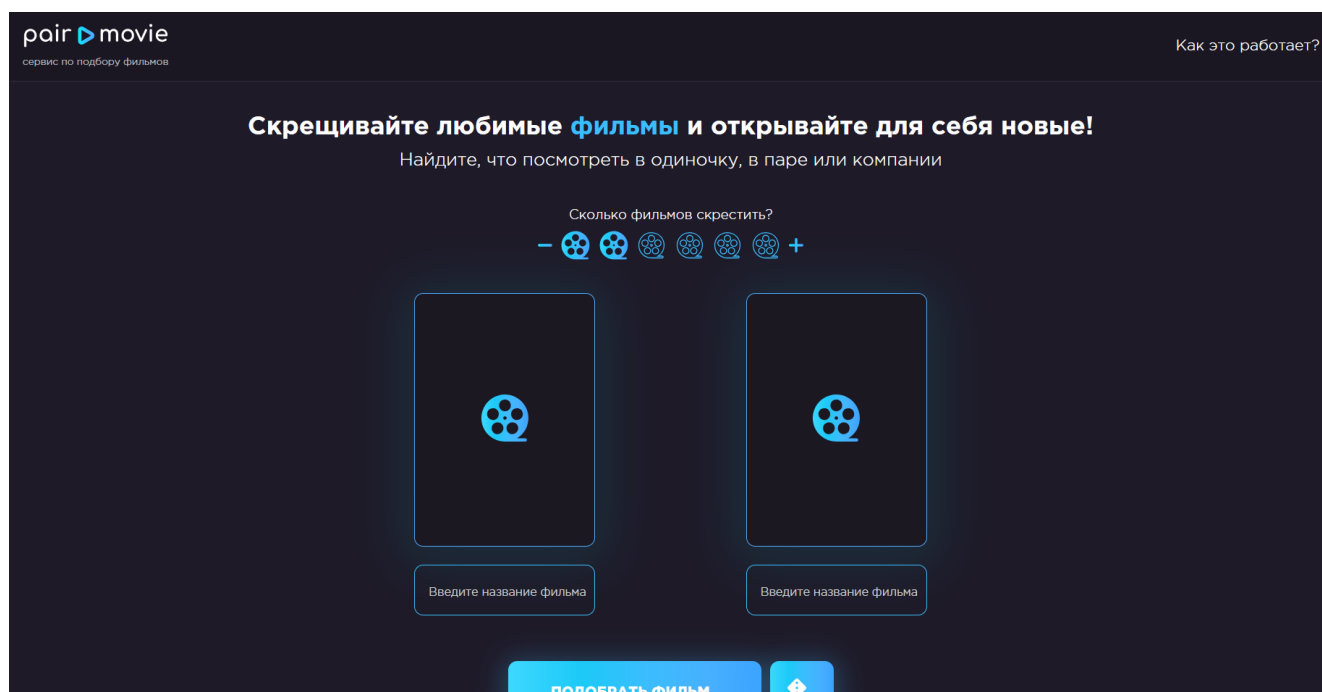


Рисунок 1.1 – Главная страница сайта «PairMovie»

Цель сервиса «PairMovie» заключается в создании удобной платформы, которая поможет пользователям сделать выбор фильмов для совместного просмотра проще и приятнее. Он стремится стать онлайн-пространством, где каждый участник сможет легко найти подходящие фильмы для просмотра в компании своего партнера, учитывая их предпочтения и настроение. Кроме того, PairMovie нацелен на создание дружественного сообщества кинолюбителей, где пользователи могут обмениваться рекомендациями, отзывами и идеями о фильмах, делая процесс выбора более интерактивным и вдохновляющим.

На главной странице сайта расположены ключевые элементы, такие как меню навигации, логотип и кнопка «Как это работает?», которая предоставляет дополнительную информацию о функционале сервиса. Дополнительно представлен текстовый блок с описанием сервиса, который помогает посетителям быстро понять его основные принципы и возможности. На главной странице также имеются два окна с полями для ввода названия фильмов и кнопка «Подобрать фильм», которая запускает процесс подбора и анализа предложений.

Важно отметить, что на сайте реализована мобильная версия, которая обеспечивает корректное отображение контента как на экранах компьютеров, так

и на смартфонах. Адаптивная верстка позволяет элементам интерфейса автоматически перестраиваться и оптимизироваться для различных размеров экранов, обеспечивая пользователям комфортное взаимодействие с сервисом в любых условиях. Мобильная версия представлена на рисунке 1.2.



Рисунок 1.2 – Мобильная версия сайта «PairMovie»

Логотип, представленный на рисунке 1.3, представляет собой стилизованное текст и кнопку «play», объединенных в одном композиционном целом. Каждые из слов, располагающихся по обоим сторонам кнопки «play», символизируют двух различных пользователей, объединенных одним общим интересом – кино. Пользователю сразу будет понятно, чем занимается сервис. Логотип легко заметен, при изменении адаптивности сайта логотип масштабируется.

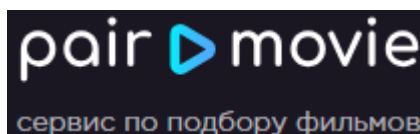


Рисунок 1.3 – Логотип онлайн-сервиса «Moviq»

Навигации на сайте нет, так как основная функциональность сервиса располагается на главной странице. Вся необходимая информация о рекомендованных фильмах и инструментах для поиска доступна непосредственно на главной странице, что упрощает и ускоряет процесс использования сервиса. Логотип является ссылкой на главную страницу.

Возможность регистрации и вход на сайт отсутствуют.

и различные подборки фильмов, обеспечивая пользователям широкий выбор и удобный доступ к контенту.

Главная страница сайта представлена на рисунке 1.5.

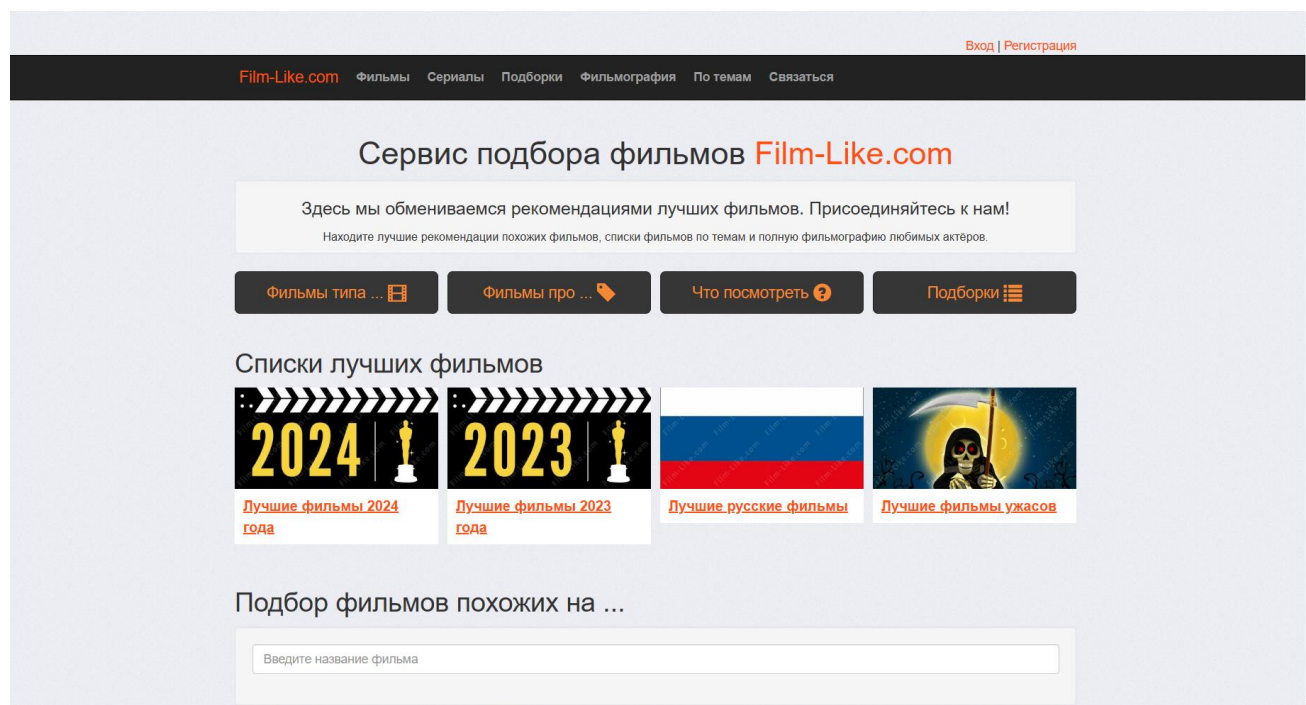


Рисунок 1.5 – Главная страница сервиса «Film-Like»

Главная страница объединяет в себе весь функционал сайта, она демонстрирует часть каждой из второстепенных страниц, таким образом, пользователю нет необходимости кликать на все пункты меню в поисках необходимой информации, с большой вероятностью, он найдет ее на главной странице сайта, при необходимости перейдет на другую страницу.

Навигация представлена в виде горизонтального меню, которое содержит пункты для перехода на другие страницы сайта. Навигация сайта понятна и удобна. Хлебный крошки отсутствуют. Логотип сайта располагается в хедере и является ссылкой на главную страницу.

В футере сайта расположена только политика конфиденциальности.

Логотип сайта текстовый, представлен на рисунке 1.6. Логотип представляет собой название сервиса. Цвет логотипа – оранжевый. Фон – прозрачный. Логотип и сайт выполнены в одном стиле, поэтому логотип хорошо сочетается с другими элементами сайта.



Рисунок 1.6 – Логотип сервиса «Film-Like»

Сайт выдержан в едином стиле.

Цветовая гамма – светло-серый, оранжевый, серый.

Не все графические элементы, используемые на сайте, высокого качества, но все соответствуют информации, к которой относятся и интуитивно понятны.

По результатам анализа можно выделить слабые стороны сайта: отсутствие «хлебных крошек» и недостаточно понятная навигация и сильные стороны сайта: имеющаяся фильтрация, удачная цветовая гамма, полная адаптивность, единый стиль сайта, большой выбор фильмов.

1.2.3 Анализ онлайн-сервиса «filmpro»

Сервис «filmpro» [5] представляет собой онлайн-платформу для подбора фильмов, разработанную для широкого круга аудитории, увлеченной кинематографом. Его главная цель – помочь пользователям находить фильмы на основе их предпочтений, интересов и настроения.

Целевая аудитория «filmpro» охватывает взрослых пользователей в возрасте от 18 до 45 лет, проживающих в различных географических регионах. Это в основном люди со средним и выше среднего уровнем дохода, увлеченные кинематографом и стремящиеся провести свободное время качественно и эффективно. Среди аудитории могут быть как студенты, так и трудовые работники, проявляющие интерес к кино и желающие расширить свой кругозор в данной области.

Сайт сервиса «filmpro» предлагает пользователям обширный выбор разнообразных фильмов, а также разнообразные подборки, включающие график премьер, афиши кинотеатров, новости, интервью и другой контент, связанный с кино.

Главная страница сайта демонстрируется на рисунке 1.7.

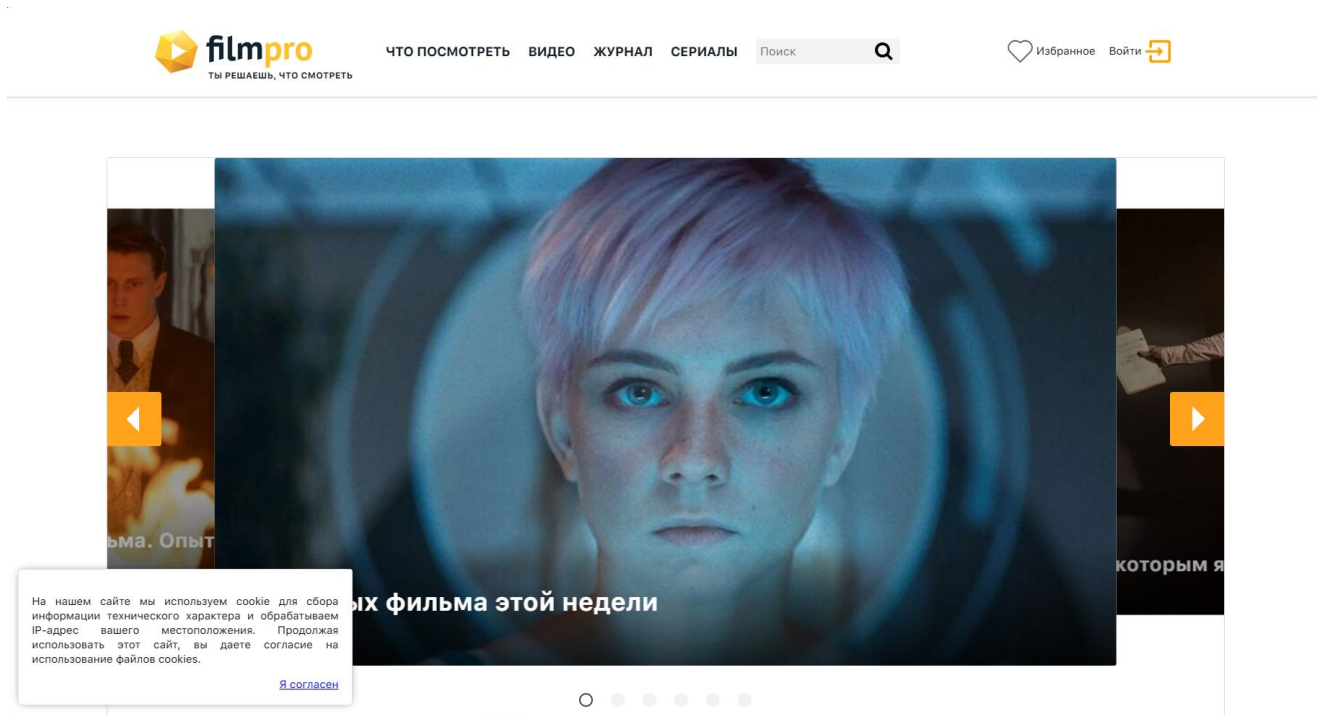


Рисунок 1.7– Главная страница сервиса «filmpro»

Мобильная версия сайта полностью поддерживается. Она отображается оптимально как на экранах компьютеров, так и на смартфонах. Верстка сайта адаптивна, что означает, что элементы сайта автоматически перестраиваются и изменяются в соответствии с различными устройствами, на которых открывается сайт. Мобильная версия сайта может быть просмотрена на рисунке 1.8.

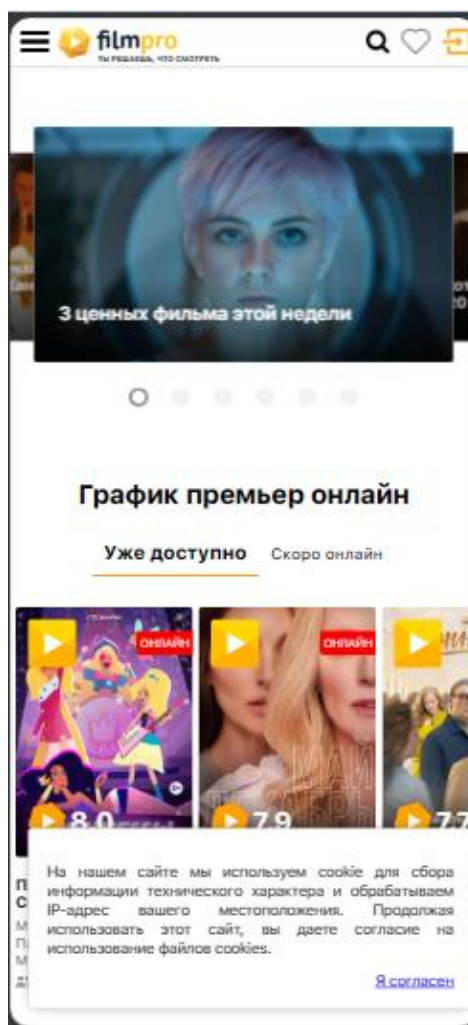


Рисунок 1.8– Мобильная версия сервиса «filmpro»

Хедер на главной странице большой и содержит логотип, меню, поле для поиска, а также кнопки авторизации и регистрации. На главной странице располагается слайдер с различными подборками, блок новостей, трейлеры.

В футере сайта находится полное меню, информация о сервисе.

Страницы с описанием конкретного фильма содержат информацию о фильме: название, трейлер, постер, описание, а также рейтинг.

Навигация осуществляется только с помощью меню в шапке, которое закреплено при скролле, или футере. Пользователь, находящийся на середине страницы сможет перейти на главную или другие страницы сайта. Отсутствуют хлебные крошки.

Логотип является текстово-графическим и содержит название компании. Слева от названия – рисунок кнопки «play». Цвет логотипа – желтый с черным. Выполнен в стиле сайта. Логотип представлен на рисунке 1.9.



Рисунок 1.9 – Логотип сайта «filmpro»

Цветовая гамма – белый, светло-серый и желтый. На сайте присутствует уместное количество анимации: слайдер, анимации наведения и другие.

По результату анализа сайта можно сделать выводы и обозначить слабые стороны сайта, а именно отсутствие «хлебных крошек». К сильным сторонам веб-сайта относится единая стилистика, адаптивность, удобная навигация, множество контента, удобный выбор фильмов, а также подборок. Сервис полностью выполняет свою функцию удобного выбора фильмов.

1.2.4 Анализ онлайн-сервиса «Кинопоиск»

Сервис «Кинопоиск» [6] – это онлайн-платформа, предназначенная для поиска информации о фильмах, сериалах, актерах и других кинематографических произведениях. Он предоставляет пользователям возможность получить доступ к обзорам, оценкам, рецензиям, трейлерам и другой полезной информации о кино.

Целевая аудитория сервиса «Кинопоиск» включает в себя людей разного возраста, уровня дохода и места проживания, которые интересуются кино и хотят быть в курсе последних новостей и обзоров о фильмах. Основной сегмент аудитории – это активные кинолюбители в возрасте от 18 до 45 лет, проживающие как в городских, так и в пригородных районах. У данной аудитории обычно средний или выше среднего достаток.

Цель сервиса «Кинопоиск» заключается в предоставлении пользователю все-сторонней информации о фильмах, а также создании удобной платформы для поиска и выбора контента на основе его предпочтений. Кроме того, «Кинопоиск» стремится создать активное сообщество кинолюбителей, которые могут обмениваться мнениями, оценками и рекомендациями о фильмах.

Главная страница сайта изображена на рисунке 1.10.

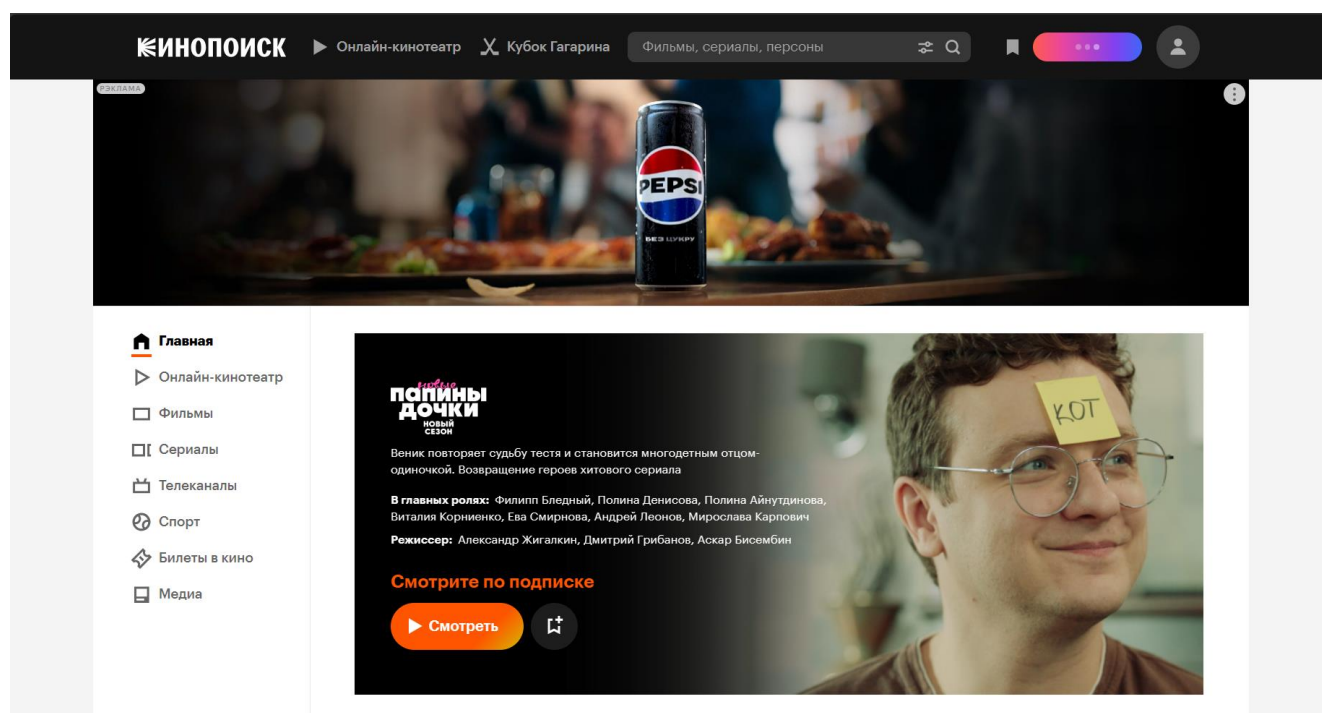


Рисунок 1.10 – Главная страница сервиса «Кинопоиск»

На сайте есть мобильная версия (рисунок 1.11). Верстка адаптивная, что означает, что сайт оптимизирован для корректного отображения как на экранах компьютеров, так и на смартфонах. Благодаря адаптивной верстке, пользователи могут легко находить и просматривать информацию о фильмах, читать рецензии, просматривать рейтинги и многое другое, используя смартфон или планшет.

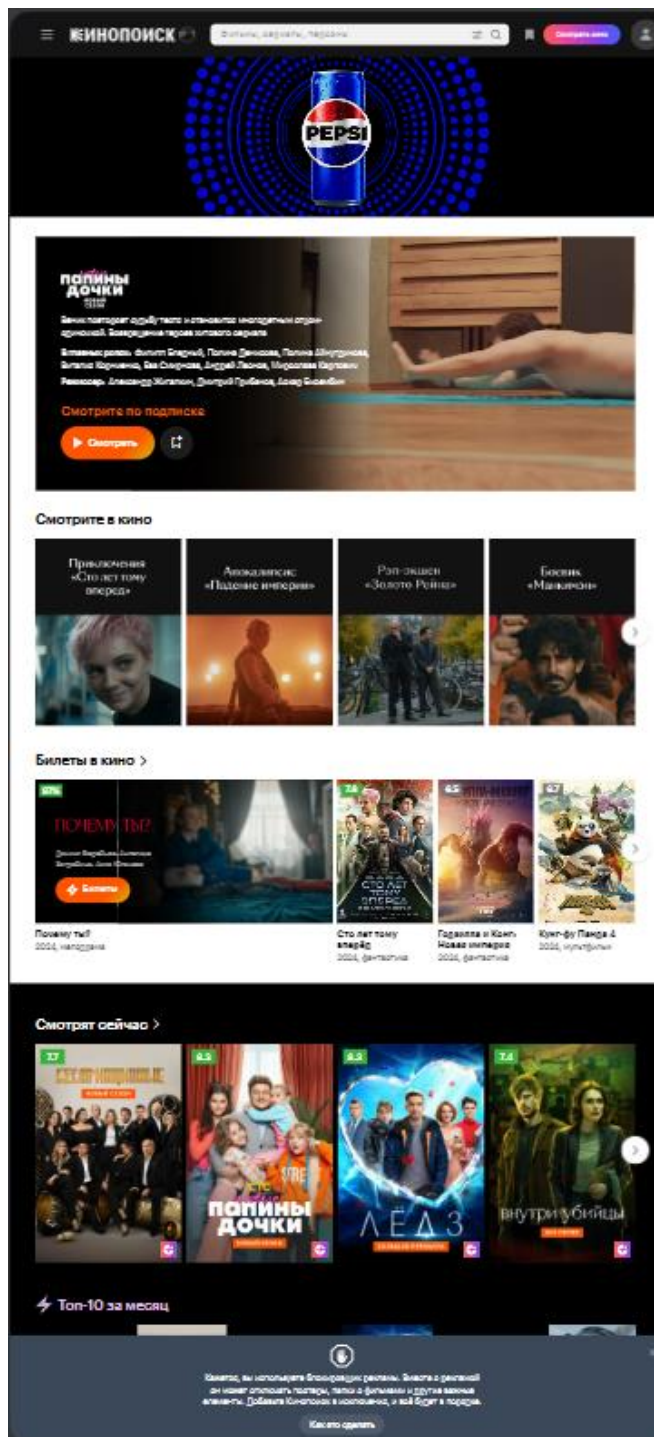


Рисунок 1.11 – Мобильная версия сервиса «Кинопоиск»

На главной странице сайта располагается меню, логотип, поле для поиска, кнопки для авторизации и регистрации. Далее вертикальная навигация по страницам, а также различные рекомендации, блок новостей, календарь релизов, сборы по фильмам и так далее.

Логотип является уникальным, пользователю сразу понятно, что делает сервис. Логотип является символьным, однако первая буква в логотипе стилизована под лучи проектора.

При изменении адаптивности сайта логотип масштабируется. Логотип выполнен в белом цвете на черном фоне.

Сам логотип представлен на рисунке 1.12.



Рисунок 1.12 – Логотип сервиса «Кинопоиск»

Навигация представлена горизонтальным меню в верхней части страницы. Выбранный пункт меню, то, на какой из страниц находится сейчас пользователь, подсвечивается другим цветом. Меню фиксируется при скролле. Логотип на других на страницах сайта является ссылкой на главную страницу. «Хлебные крошки» отсутствуют, однако это не мешает удобной навигации по сервису.

Футер имеет рекламу подписки на сервис, а также меню, информация о правах, ссылки на социальные сети, ссылки на онлайн-магазины приложений.

Стоит также заметить, что на странице подбора фильмов, фильмы можно фильтровать по многим параметрам: названиям, жанрам, странам, годам, критике, сборам, премиям.

Основными цветами являются белый, черный, оранжевый, серый. Шрифты подобраны грамотно и вписываются в общий стиль. На главной странице сайта преобладает качественная графическая информация с элементами анимации.

На страницах используются графические элементы – иконки, фотографии. Текст на всех страницах структурирован и легко воспринимается.

По результату анализа можно сделать вывод, что сервис «Кинопоиск» успешно реализует свою цель, предоставляя пользователям подробную информацию о фильмах и создавая удобное место для поиска и выбора фильмов на основе их предпочтений. В целом, «Кинопоиск» представляет собой полноценную и информативную онлайн-платформу для кинолюбителей, обладающую высоким уровнем удобства использования и широким набором функций.

На основании проведенного анализа конкурентов можно сделать вывод, что для данной темы веб-сайта можно использовать различные стилевые решения. Больше всего использовался стиль «Минимализм». Благодаря этому стилю можно качественно преподнести информацию, при этом дизайн будет удобен для пользователей сервиса.

Цветовая гамма может содержать яркие элементы для привлечения внимания, однако она не должна отвлекать от графических элементов (обложек фильмов). Кроме того, фотографии должны быть хорошего качества.

Шрифты должны способствовать быстрому визуальному поиску необходимой информации и ее восприятию.

В логотипах найденных аналогов использованы простые шрифты без засечек, стилизована кнопка «play» или название сервиса целиком. Стоит отметить, что во всех сайтах логотип сочетался со стилевым решением сайта.

1.3 Выбор стиля дизайна, методов и технологий

Онлайн-сервис «Moviq» будет выполнен в стиле «Минимализм». Главные принципы этого стиля:

- фокусировка пользователя на функциональности;
- много свободного пространства;
- ограниченная цветовая палитра;
- интерфейс чистый и незагроможденный.

Каждый дизайн создаваемого сайта должен иметь свою «изюминку», которая будет отличать его от остальных. Из этого следует, что дизайн веб-сайта должен:

- иметь понятную структуру;
- владеть максимально удобным интерфейсом;
- использовать анимацию;
- иметь качественный графический контент;
- иметь уникальный и узнаваемый логотип.

Сайт будет содержать следующие элементы: хедер с логотипом; горизонтальное меню; вертикальное меню; футер, содержащий логотип, информацию о правах, ссылки на социальные сети. При просмотре с монитора ноутбука или компьютера структура будет двенадцатиколончатая. При просмотре с мобильного устройства – четкая двухколоночная структура.

Следуя принципам юзабилити, основные навигационные блоки и логотип компании при входе в сервис будут в вертикальном меню для того, чтобы посетитель без труда нашел необходимую ему информацию. Также следует добавить «Хлебные крошки» для удобства навигации между контентом. Шрифты, используемые в этом стиле, хорошо читаемы при различном разрешении экрана.

При разработке логотипа следует использовать простые шрифты без засечек, однако можно и использовать декоративные шрифты (в том числе собственные шрифты). Логотип может быть как символьно-текстовый (рисование стилизованной буквы «М»), так и просто символьный.

Для разработки логотипа будет использоваться графический редактор CorelDraw. CorelDraw имеет мощные инструменты для редактирования и манипулирования векторной графикой, делая процесс разработки быстрым и эффективным.

Цветовая гамма может содержать яркие и акцентные цвета для коммерческих элементов сайта, однако она не должна отвлекать пользователя от его цели прихода на онлайн-сервис.

Дипломный проект будет выполнен в смешанном стиле минимализм и метро, в классических цветах с ярким акцентом. Предусматривается большое количество пустого пространства для придания легкости в восприятии информации.

Для разработки онлайн-сервиса будет оправданным использование следующих веб-технологий:

- V

u
e
j
s

– это JavaScript-фреймворк, предоставляющий мощные инструменты для разработки клиентской части приложения. Vue.js обеспечивает гибкую архитектуру, легко масштабируемую и обеспечивающую плавную загрузку страниц. Благодаря своей простоте и производительности, Vue.js идеально подходит для создания интерактив-

Он предоставляет возможность использования переменных, вложенных правил, миксинов и других функций, что обеспечивает более организованный и гибкий CSS код.

- Laravel – фреймворк языка PHP. Laravel предоставляет удобные инструменты для обработки запросов, работы с базой данных и обеспечения безопасности приложения. Этот фреймворк также позволяет эффективно организовать код и упростить процесс разработки.

— M

y

S

Q

L - A

ж для хранения данных приложения будет использоваться база данных MySQL. MySQL обеспечивает надежное хранение данных и эффективное выполнение запросов, что позволяет обеспечить корректное функционирование онлайн-сервиса и быстрый доступ к необходимой информации.

— это WebRTC для JavaScript, обеспечивающий простоту, эффективность и способностью действовать в браузере, что позволяет разработчикам внедрять в свои веб-приложения технологию peer-to-peer для передачи данных между клиентом и сервером без перезагрузки страницы, что улучшает производительность и отзывчивость приложения.

1.4 Выводы по разделу

На основе изучения предметной области и анализа аналогичных сервисов было принято решение о создании онлайн-сервиса под названием «Moviq».

Исходя из проведенного анализа, были сформулированы следующие требования к проекту:

- Четкая и понятная структура онлайн-сервиса для удобства пользователей.

- Максимально удобный и стильный интерфейс, обеспечивающий приятный пользовательский опыт.

- Наличие высококачественного графического контента для привлечения внимания посетителей.

- Адаптивный дизайн, который обеспечивает корректное отображение сайта на различных устройствах.

– Разработка функционала для пользователей, включая предоставление рекомендованных фильмов, фильтрацию по различным параметрам, личный кабинет, а также регистрацию и авторизацию.

- Разработка функционала для администратора, включая возможность управления фильмами, таких как добавление, удаление и редактирование.

- Создание уникального и запоминающегося логотипа для легкого узнаваемости бренда.

Для создания серверной части проекта был выбран фреймворк Laravel, а для работы с базой данных будет использоваться MySQL. Для клиентской части приложения основным инструментом будет JavaScript-фреймворк Vue.js, а также будет использоваться SASS для управления стилями.

2 Проектирование интерфейсов и структуры данных онлайн-сервиса

2.1 Цели и задачи проекта

Для того, чтобы онлайн-сервис был удобен для пользователей, необходимо грамотно организовывать структуру сайта, а также иметь интуитивно понятный интерфейс. Кроме того, помимо дизайна, сервис должен обладать обширным функционалом, чтобы удовлетворять потребности пользователей.

Цели сервиса прямым или косвенным образом способствуют увеличению прибыльности компании. Целями создания онлайн-сервиса «Moviq» являются:

- привлечение большего количества пользователей;
- формирование постоянной пользовательской базы;
- формирование положительного имиджа в интернет-сообществах;
- увеличение количества подписок и просмотров.

Разработка сервиса позволит пользователям иметь доступ к обширной базе данных фильмов и получать рекомендованные фильмы. Потенциальные пользователи будут иметь возможность также предложить добавить свои фильмы, отправив данные по ним администратору сервиса. Разработка административной панели будет подразумевать упрощение работы с базой данных, то есть возможность быстрого управления динамическими частями контента сайта, используя удобный интерфейс.

С учетом целей онлайн-сервиса были выделены следующие основные задачи:

- разработка логотипа;
- проектирование и реализация функциональной части проекта;
- регистрация и авторизация пользователей;
- разработка простой навигации по сайту;
- создание базы данных;
- предоставление рекомендованных фильмов;
- разработка удобного и привлекательного пользовательского интерфейса;
- реализация поиска и фильтрации фильмов;
- создание личного кабинета для пользователей;
- разработка административной панели для управления информацией на сайте.

Выполнение данных задач позволит достигнуть поставленных целей и разработать качественный продукт, отвечающий всем современным требованиям.

2.2 Целевая аудитория

Целевая аудитория подразумевает под собой совокупность потенциальных потребителей услуги, которые могут изменить свои предпочтения в пользу данной

					БГТУ 02.00.ПЗ			
Изм	Лист	№ документа	Подп.	Дата	Проектирование интерфейсов и структуры данных онлайн-сервиса	Лит.	Лист	Листов
Разраб.	Помоз					у	1	13
Пров.	Осоко					74319008, 2024		
Консульт.								
Н. контр.	Осоко							
Утв.	Романенко							

услуги под воздействием маркетинговых мероприятий. Целевая аудитория включает круг пользователей, для которых предназначен конкретный сервис.

Онлайн-сервис «Moviq» будет иметь целевую аудиторию, которая включает людей, заинтересованных в простом и удобном поиске фильмов.

Это могут быть любители кино, ищущие новые фильмы для просмотра, семейные пары, желающие найти фильм для совместного вечера, а также молодые люди, интересующиеся новинками киноиндустрии.

Кроме того, могут быть пользователи, которые ищут специфические жанры или фильмы по настроению.

Целевая аудитория также может включать в себя киноманов, которые ценят удобный интерфейс. Основной фокус будет сделан на людей со средним и высоким уровнем дохода, активно пользующихся интернетом и стриминговыми сервисами.

Для описания целевой аудитории можно использовать следующие социально-демографические характеристики:

- пол;
- возраст;
- семейное положение;
- предпочтения и интересы;
- география проживания.

Таким образом, можно выделить три группы пользователей, подходящих под целевую аудиторию выбранной тематики проекта:

- любители кино, ищущие новые фильмы для просмотра;
- семьи, которые хотят найти фильм для совместного времяпрепровождения;
- молодые люди, интересующиеся новинками кино и рекомендациями.

Проект будет ориентирован на создание удобной платформы, способствующей удовлетворению потребностей каждой из этих групп пользователей.

2.3 Разработка персонажей и создание пользовательских сценариев

Были разработаны персонажи и пользовательские сценарии для каждой из целевых аудиторий, определенных в разделе 2.2.

Пользовательский сценарий представляет собой выдуманную историю о том, как пользователь использует продукт для достижения своих целей. Сценарий обращает особое внимание на мотивацию пользователя в ходе взаимодействия с дизайном и документирует особенности поведения на сайте [7].

2.3.1 Ключевой персонаж №1

Первый ключевой персонаж – мужчина Александр, 38 лет. Александр работает в сфере информационных технологий. У него есть семья, и он ценит время, проведенное с близкими после работы. Любит отдыхать за просмотром фильмов различных жанров в свободное время.

Александр является активным пользователем интернета, вечерами он часто ищет новые фильмы для просмотра или читает отзывы о них. Он предпочитает использовать как сайты, так и мобильные приложения, он не испытывает сложности в понимании структуры сайта и с лёгкостью привыкает к новым интерфейсам.

Изображение Александра представлено на рисунке 2.1.



Рисунок 2.1 – Персонаж Александр

Он предпочитает использовать как сайты, так и мобильные приложения, он не испытывает сложности в понимании структуры сайта и с лёгкостью привыкает к новым интерфейсам.

Важные для Александра функции сервиса:

- скорость работы;
- интуитивно понятный интерфейс;
- простая навигация по сервису.

В ближайшее время у Александра день рождения его жены, и он хочет приготовить приятный сюрприз – вечер с просмотром её любимых фильмов. Он ищет сервис, который поможет ему подобрать фильмы по предпочтениям его супруги.

Сценарий использования. Александр вводит запрос «подобрать фильм» в поисковую строку. Среди результатов поиска он выбирает сервис «Moviq» и переходит на главную страницу. После того как он ознакомится с описанием сервиса, Александр переходит на страницу регистрации и заполняет форму, после этого попадает на страницу авторизации и заполняет форму. Александр попадает на страницу «Подборки», переходит на страницу «Рекомендации». В фильтрах Александр указывает предпочитаемые женой жанры. После этого он просматривает рекомендованные фильмы и выбирает подходящий.

Добившись своей цели, Александр покидает сайт, закрыв вкладку браузера.

2.3.2 Ключевой персонаж №2

Второй ключевой персонаж Марина, 37 лет. Она работает в области международной консалтинговой фирмы и постоянно находится в командировках. Марина

любит проводить время за просмотром кинофильмов различных жанров. Изображение Марины представлено на рисунке 2.2.



Рисунок 2.2 – Персонаж Марина

Интернет-активность Марины высокая, в свободное время она часто пользуется Интернетом, в основном сидит в социальных сетях со смартфона или использует мобильные приложения, но при необходимости также пользуется и браузером. Однако, иногда она испытывает сложности с навигацией на сайтах, особенно если интерфейс запутанный.

Важные функции для Марины:

- быстрая и удобная навигация;
- скорость работы сайта;
- четкая структура сайта.

Марина любит расслабиться, просматривая кино. Однако, она не хочет тратить время на поиск каждого отдельного фильма. Она ищет сервис, который сможет предложить ей широкий выбор интересных фильмов, а также сохранить эти выбранные фильмы в удобном формате.

Сценарий пользователя: Марина находит сервис «Moviq», авторизуется с помощью Google. Она попадает на страницу «Подборки», создает подборку и переходит на страницу «Рекомендации». Пролистывая предложенные фильмы, она находит интересующие и добавляет их в свой созданный список для просмотра. Теперь у неё есть много вариантов для просмотра в свободное время, и она может наслаждаться просмотром без необходимости долгих поисков.

2.3.3 Ключевой персонаж №3

Третий ключевой персонаж – Анна, 30 лет, работающая аналитиком в крупном рекламной агентстве в Москве. Анна любит кино и каждую неделю ходит в кинотеатр или смотрит фильмы дома. На этот раз у нее планируется выходной, и она решила устроить домашний киномарафон.

Изображения Анны представлено на рисунке 2.3.



Рисунок 2.3 – Персонаж Анна

Интернет-активность Анны высока, на работе она проводит много времени за компьютером, поэтому привыкла к быстрой навигации по сайтам и особенно ценит интуитивно понятный интерфейс.

Функции, которые важны пользователю:

- удобный и понятный интерфейс;
- быстрая работа сайта;
- минимум лишней информации;
- качественный контент.

Сценарий пользователя. Так как Анна уже не первый раз пользуется сервисом, она авторизуется, открывает свою подборку, вписывает в поле поиска названия интересующих фильмов и подбавляет их в свою подборку. Далее она хочет найти еще несколько фильмов, переходит на страницу «Рекомендации» и фильтрует фильмы по своим предпочтениям. Она находит интересующие фильмы и также добавляет их в свою подборку. Один из фильмов она уже просматривала ранее, поэтому она решила оставить оценку фильму.

2.4 Проектирование взаимодействий пользователей

Диаграммы прецедентов представляют собой один из пяти типов диаграмм, применяемых в UML для моделирования динамических аспектов системы.

На диаграмме использования изображаются:

- актеры – группы лиц или систем, взаимодействующих с нашей системой;
- варианты использования (прецеденты) – сервисы, которые наша система предоставляет актерам;
- комментарии;
- отношения между элементами диаграммы [8].

Для онлайн-сервиса была разработана UML диаграмма прецедентов (вариантов использования), представленная на плакате БГТУ 03.00 РР.

Представленная схема демонстрирует все возможные сценарии, которые могут происходить при использовании веб-сайта как обычным пользователем (потен-

циальным клиентом), так и администратором при использовании панели администратора. Для использования панели администратора пользователю необходимо пройти авторизацию под определенным email и паролем.

2.5 Информационная структура проекта

Структуры сайта является иерархической. Пользователь, начиная с главной страницы, последовательно переходит на все последующие страницы. Переход на элементы второго уровня может быть осуществлен с любой страницы сайта через горизонтальное или вертикальное меню, не требуется возвращение на главную страницу веб-сайта для этого.

Сервис включает в себя следующие страницы:

- «Главная страница». Содержит информацию о сервисе, а также кнопку для перехода на страницу авторизации. Функционал страницы: переход на страницу авторизации, регистрации из меню, переход на страницу авторизации посредством нажатия кнопки.

- «Авторизация». Содержит форму для входа в аккаунт с помощью формы, а также с помощью Google. При нажатии на кнопку «Создать аккаунт» осуществляется переход на страницу «Регистрация».

- «Регистрация». На странице содержатся форма с полями для регистрации, есть кнопка «У вас уже есть аккаунт?» для перехода на страницу авторизации.

- «Личный кабинет». Для пользователя страница содержит его данные, а также информацию о предложенных фильмах.

- «Панель администратора». Включает в себя те же страницы, что и обычный пользователь, однако в личном кабинете у администратора есть возможность управления данными. Для администратора сервиса страница содержит возможности поиска, добавления, удаления и редактирования фильмов, возможность выдачи прав для пользователей.

- «Предложенные фильмы». Список предложенных фильмов пользователей.

- «Предложенный фильм». Содержит данные о предложенном фильме.

- «Рекомендации». Страница содержит данные о рекомендованном фильме, кнопки для навигации между фильмами, а также возможности настройки предпочтений по рекомендациям, включая сортировку, фильтрацию и поле поиска.

- «Подборки». Страница содержит пользовательские подборки, созданные пользователем.

- «Подборка». Страница содержит список фильмов в текущей подборке.

- «Фильм». Данная страница содержит подробные данные о фильме, выбранном пользователем.

В структурной схеме отражены страницы, которые будут присутствовать на веб-сайте. Структурная схема представлена на чертеже БГТУ 02.00 С1.

2.6 Построение интерактивного прототипа интерфейса

Для создания прототипов была использована программа Figma, которая является одной из самых известных и популярных для разработки прототипов, макетов

и графического материала. Она достаточно мощная для создания сложных элементов, но при этом интуитивно понятная, позволяя быстро подготовить макет. Figma поддерживает создание как статических, так и динамических прототипов и предлагает множество встроенных элементов.

При разработке макетов сайта были учтены принципы композиции, особенно симметрия, которая легла в основу прототипов. Все страницы сайта оформлены в едином стиле, при этом симметричное равновесие достигается за счет равномерного распределения объектов по визуальной массе, расположенных на одинаковом расстоянии от центральной оси. Это создает ощущение формальности и элегантности, что полностью соответствует тематике сайта. Такая композиция позволяет пользователям легко ориентироваться на сайте, обеспечивая удобство.

Прототип сайта был создан с использованием горизонтальной сетки, что обеспечило четкую и упорядоченную структуру. Каждая страница состоит из трех основных элементов: хедера, основного контента и футера. Хедер является сквозным элементом, повторяющимся на каждой странице, и содержит логотип и выпадающее меню с данными пользователя, что позволяет пользователям быстро получить доступ к важной информации и навигационным элементам. Футер также является сквозным элементом, включающим контактную информацию, ссылки на социальные сети и другие полезные ресурсы. Кроме того, на каждой странице присутствует вертикальное меню, обеспечивающее переход между различными разделами сайта.

Прототип главной страницы представлен на рисунке 2.4.

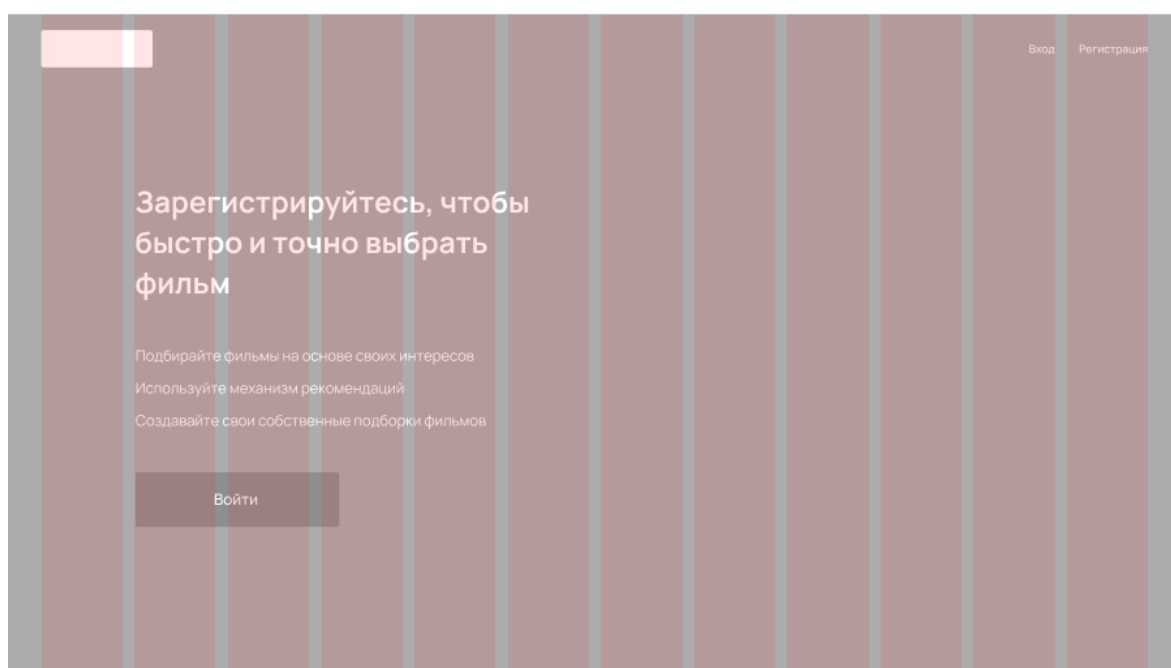


Рисунок 2.4 – Прототип главной страницы сайта

Футер сайта также является сквозным элементом, он включает в себя логотип, а также блок с ссылками на социальные сети.

Основной контент каждой страницы уникален, как по наполнению, так и по внешнему виду. На главной странице располагается призыв к регистрации в сервисе, блок с текстом об основном функционале, а также кнопка для перехода на страницу авторизации.

Страницы с подборками и фильмами в этих подборках являются однотипными, содержат текстовую и графическую информацию. Страница с подборками включает вертикальное меню, текст с приветствием и блоки с подборками. Также на странице имеется кнопка для создания новой подборки. Прототип данной страницы представлен на рисунке 2.5.

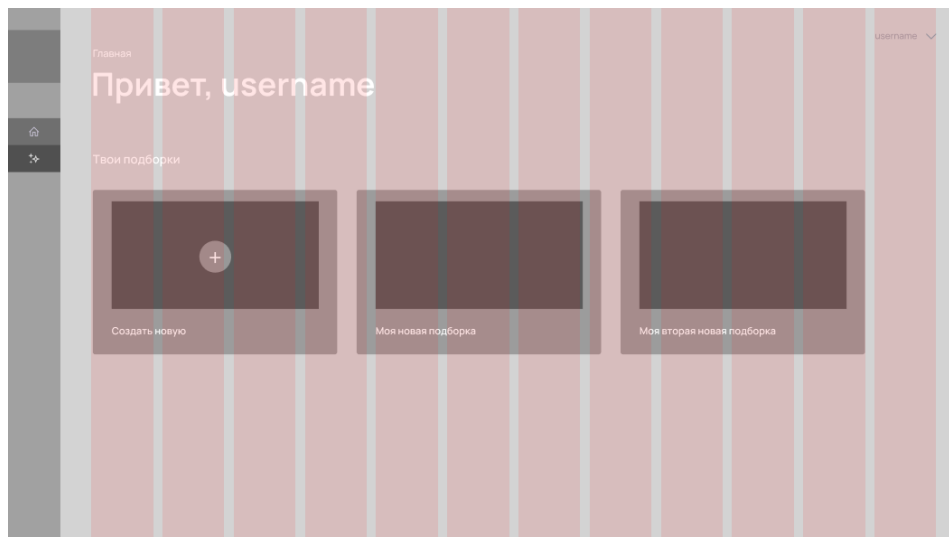


Рисунок 2.5 – Прототип страницы «Подборки»

Страница «Подборка» содержит набор фильмов, которые пользователь добавил в данную подборку. Также страница имеет кнопку для добавления фильмов в текущую подборку. Прототип представлен на рисунке 2.6.

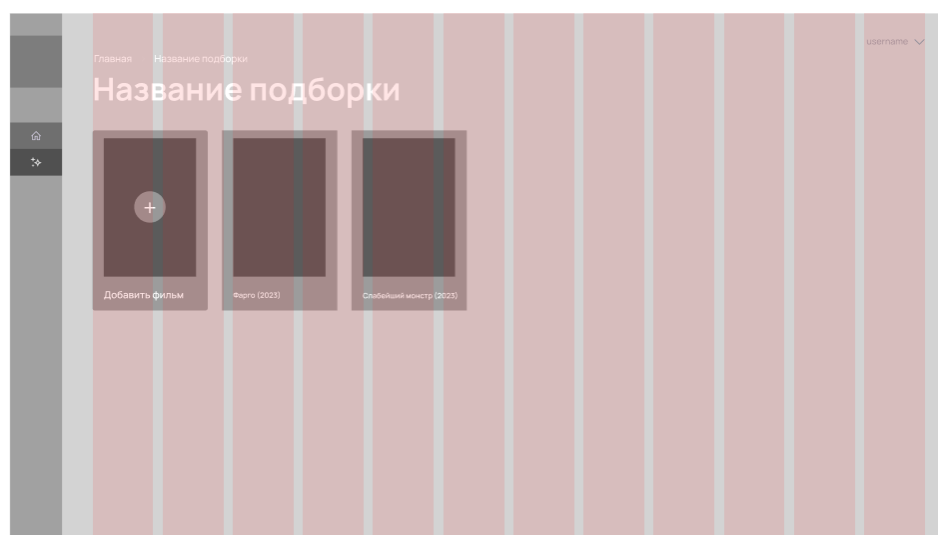


Рисунок 2.6 – Прототип страницы «Подборка»

Страница с рекомендациями имеет содержит один большой блок с описание фильма, а именно постер, название, краткое описание, характеристика, рейтинг, а также возможность поставить свой рейтинг. Кроме того, на блоке расположены кнопки для навигации между предложенными фильмами. Есть возможность перелистнуть фильм на следующий или предыдущий, а также добавить данный фильм к себе в подборку. Прототип страницы представлен на рисунке 2.7.

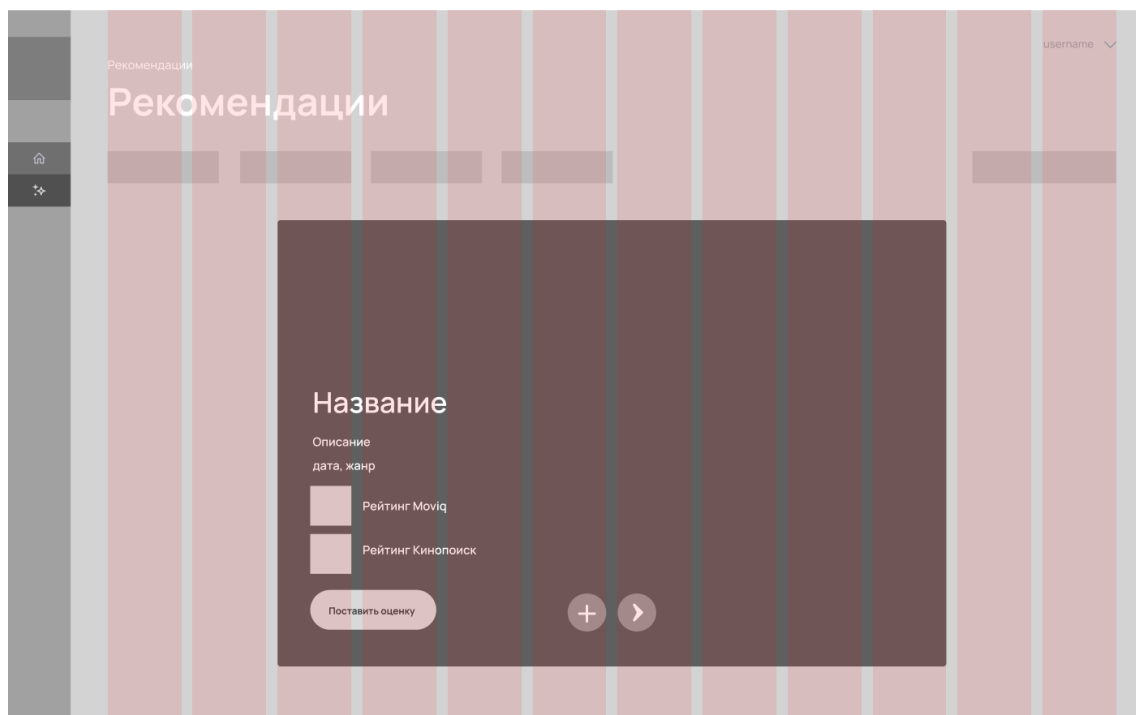


Рисунок 2.7 – Прототип страницы «Рекомендации»

Страницы «Авторизация» и «Регистрация» выполнены в едином стиле и являются однотипными. Содержат форму с полями для входа/регистрации. Фрагмент страницы представлен на рисунке 2.8.

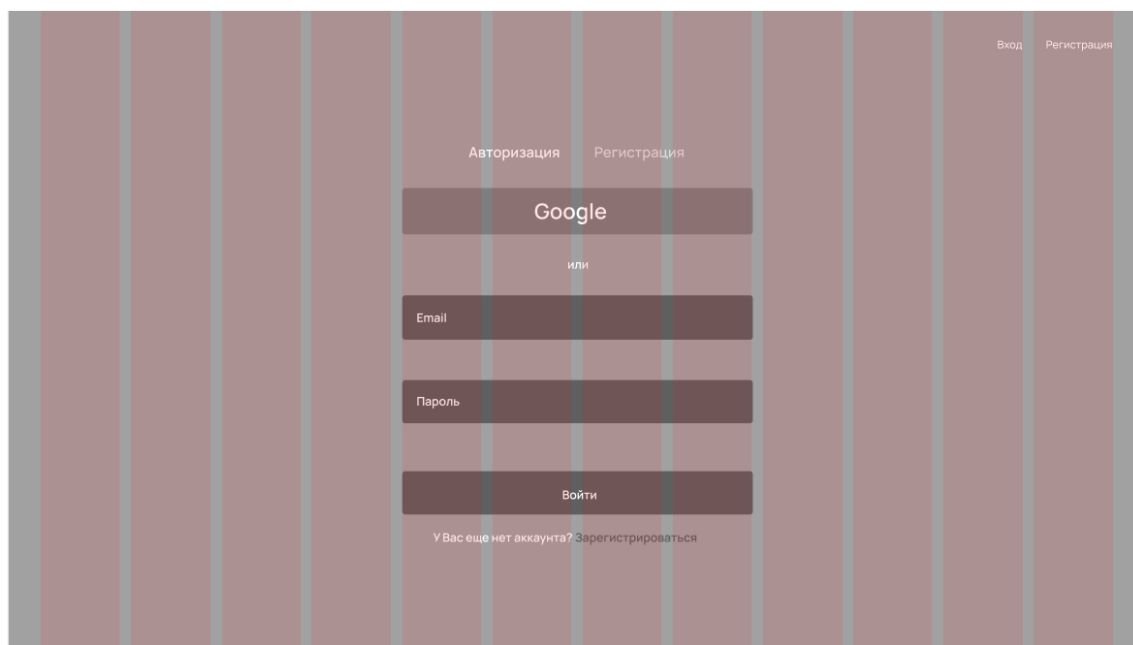


Рисунок 2.8 – Фрагмент прототипа страницы «Авторизация»

К уникальным страницам сайта также относятся следующие страницы: фильм, личный кабинет. Прототипы страниц представлены на плакате БГТУ 04.00 РР.

Разработанные прототипы страниц веб-сайта и панели администратора позволяют понять, как будут располагаться функциональные элементы на страницах сайта, какую структуру будут иметь страницы.

2.7 Проектирование базы данных

Каждая таблица имеет свой первичный ключ, который является уникальным идентификатором. Благодаря структуре этих таблиц, можно понять, как связаны между собой таблицы и их содержимое на сайте.

Для проекта создана двадцать одна таблица:

- `users` (информация о пользователях);
- `movies` (хранится информация о фильмах);
- `add_movies` (информация о предложенных фильмах от пользователей);
- `api_page_numbers` (информация о текущей странице фильмов в api);
- `backdrops` (хранится информация о фонах для фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`);
- `collections` (хранится информация о подборках);
- `collection_movies` (хранится информация о фильмах, добавленных в коллекцию, связана с таблицами `movies` по внешнему ключу `movie_id` и с таблицей `collections` по внешнему ключу `collection_id`);
- `countries` (хранится информация о странах производства);
- `feedbacks` (хранится информация о отзывах пользователей о фильмах, связана с таблицами `movies` по внешнему ключу `movie_id` и с таблицей `users` по внешнему ключу `user_id`);
- `fees` (хранится информация о сборах фильмов по странам, связана с таблицами `movies` по внешнему ключу `movie_id`);
- `film_countries` (хранится информация о фильмах, произведенных в соответствующих странах, связана с таблицами `movies` по внешнему ключу `movie_id` и с таблицей `countries` по внешнему ключу `country_id`);
- `film_genres` (хранится информация о фильмах и соответствующих им жанрах, связана с таблицами `movies` по внешнему ключу `movie_id` и с таблицей `genres` по внешнему ключу `genre_id`);
- `genres` (хранится информация о жанрах);
- `logos` (хранится информация о лототипах для фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`);
- `persons` (хранится информация о актерах фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`);
- `posters` (хранится информация о постерах для фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`);
- `ratings` (хранится информация о рейтингах для фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`);
- `restricts` (хранится информация о ограничениях по возрасту для фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`);
- `roles` (хранится информация о ролях пользователей, связана с таблицей `users` по внешнему ключу `user_id`);
- `types` (хранится информация о типах фильмов, связана с таблицей `movies` по внешнему ключу `movie_id`).

Схема логической структуры базы данных, разработанная в рамках выполнения дипломного проекта, представлена на чертеже БГТУ 01.00 С1.

2.8 Алгоритмы основных процессов

Алгоритм процесса авторизации представлен на рисунке 2.9.

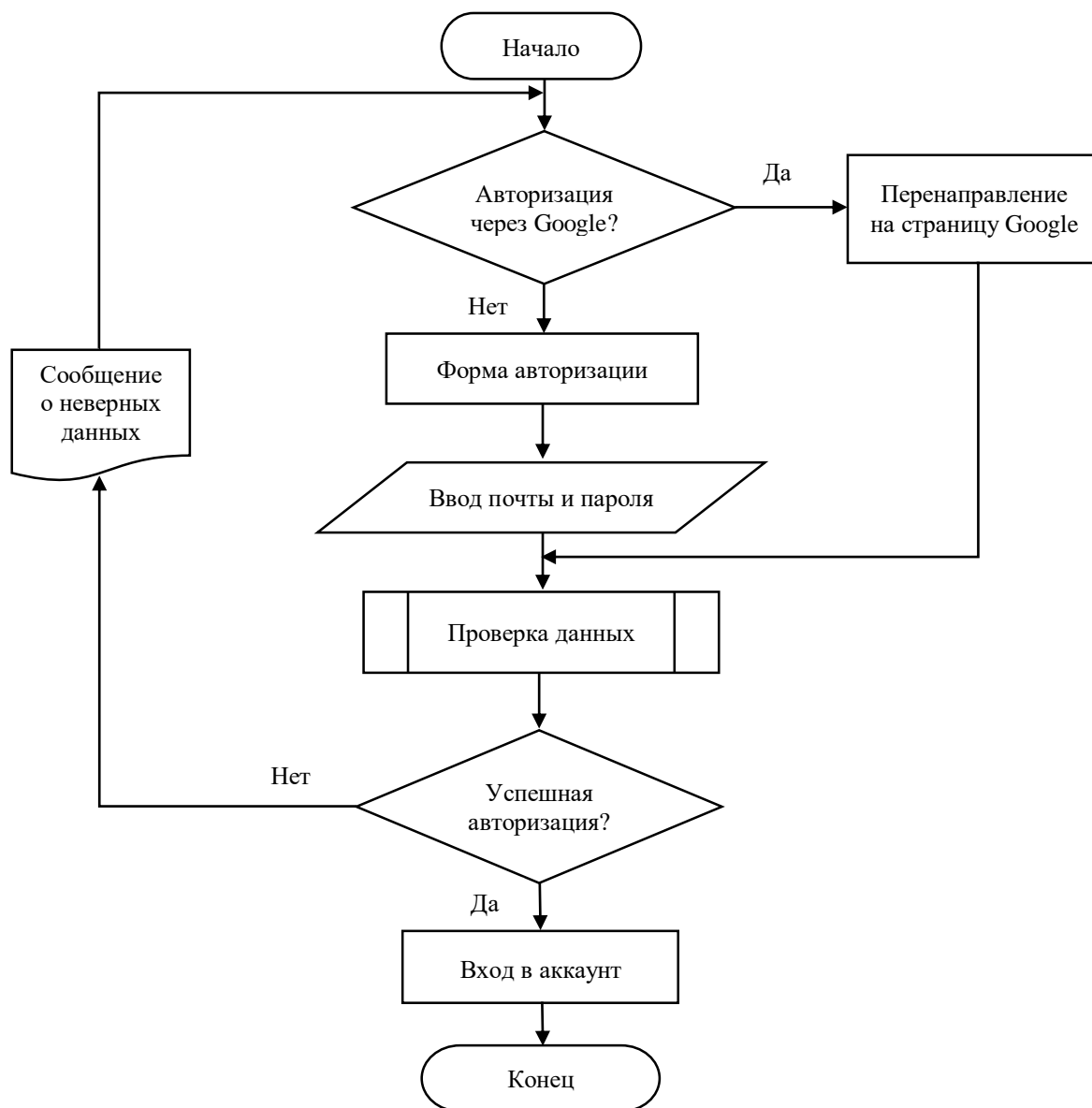


Рисунок 2.9 – Алгоритм процесса авторизации

Первым делом пользователь попадает на страницу входа и выбирает способ авторизации. Если пользователь выбирает авторизацию с помощью Google, то он перенаправляется на страницу авторизации Google. Если пользователь выбирает стандартную авторизацию, то он вводит email и пароль. Далее происходит проверка данных. Если проверка вернула отрицательный результат, пользователю показывается соответствующее сообщение, процесс возвращается к пункту отображения формы авторизации.

Алгоритм процесса добавления новой подборки пользователем представлен на рисунке 2.10.

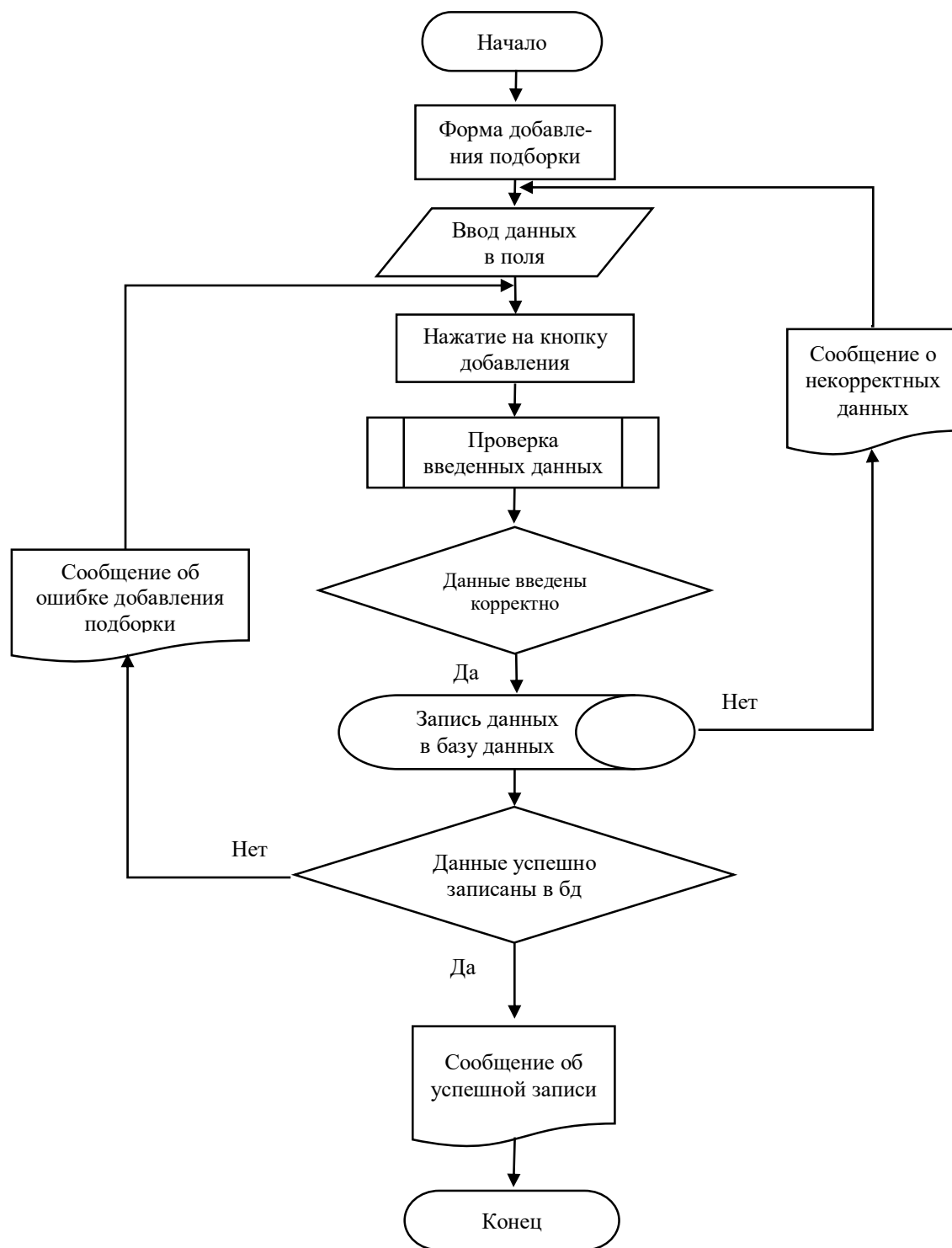


Рисунок 2.10 – Алгоритм процесса добавления подборки

Разработанный алгоритм процесса добавления подборки показывает, что первым делом пользователь попадает на форму, в поля которой он вводит требующиеся данные. Далее он нажимает на кнопку «Добавить», происходит проверка данных на корректность. Если данные не корректны, то выводится соответствующее сообщение, пользователь перенаправляется на форму. Если же данные корректны, происходит следующий этап – добавление данных в таблицу базы данных. В случае, если данные не удалось записать, пользователю отображается соответствующее сообщение, процесс завершается. Если же данные добавляются в базу, пользователь получает сообщение об успехе.

Алгоритм расчета рейтинга фильма основывается на среднем значении рейтинга, вычисленном на основе всех записей в базе данных о фильме. Для этого система суммирует все значения рейтингов, затем делит эту сумму на общее количество записей о фильме. Таким образом, рейтинг фильма определяется как среднее арифметическое всех рейтингов, оставленных пользователями для данного фильма в базе данных.

2.9 Вывод по разделу

В ходе разработки дипломного проекта были установлены цели и задачи, на основании которых определены необходимые функциональные модули.

Для определения целевой аудитории потенциальные пользователи были сгруппированы по общим характеристикам. Из каждой группы был выделен персонаж, что позволило определить и сформировать их пользовательские сценарии на сайте. Это также способствовало созданию списка наиболее важных и популярных разделов сайта.

Для разработки интерактивных прототипов страниц сайта использовался графический редактор Figma, учитывая будущий стиль дизайна сайта. Прототипы были разработаны на основе существующих данных, чтобы обеспечить удобство и понятность интерфейса для пользователей.

Созданные блок-схемы иллюстрируют пошаговый процесс входа в панель управления контентом и процесс создания новой записи, включая этап добавления контента в таблицу базы данных. В блок-схемах также рассматриваются случаи некорректного ввода данных и показано, как система реагирует на такие ситуации.

В результате была сформирована структура веб-сайта и четко определены функциональные возможности.

3 Дизайн онлайн-сервиса

Главная цель дизайна сайта заключается в том, чтобы предоставить пользователям информацию в привлекательной и удобной форме. Хороший дизайн должен сразу давать понять посетителям, что их ждет на сайте. Он должен быть визуально привлекательным, но не перегружать интерфейс. Приоритет отдается информации, поэтому дизайн должен подчеркивать её и облегчать доступ к ней, а не мешать.

Для разработки веб-сайта применяются следующие шаги:

- создание дизайна главной страницы. От нее будет зависеть весь проект;
- формирование шаблонов для остальных страниц;
- разработка дизайна второстепенных страниц.

Применяя прототипы сайта и учитывая вышеперечисленные моменты, были созданы макеты для страниц веб-сайта.

3.1 Выбор и обоснование стиля

Проанализировав аналоги, был выбран стиль минимализм. Минималистичный дизайн сайта используется для создания современного, функционального и чистого внешнего вида.

Зачастую дизайнеры воспринимают минимализм как просто набор основных элементов на сайте без добавления лишних деталей. Однако, этим стиль не ограничивается. Основная идея минимализма в дизайне – акцент на контент сайта, а не на его оформление [9].

Он характеризуется использованием простых цветовых блоков, лаконичных шрифтов и чётких линий, что эффективно привлекает внимание посетителей и создает положительное впечатление.

Для онлайн-сервиса использование минималистичного стиля может подчеркнуть удобство и легкость использования. Это также делает сайт более доступным и понятным, особенно для пользователей, которые могут быть не знакомы с подобными сервисами. В целом, минималистичный стиль помогает создать визуально привлекательный и функциональный сайт, выделяющийся на фоне конкурентов.

Главные принципы стиля минимализм:

- ориентация на пользователей и их потребности;
- использование простых цветов и графических элементов для создания привлекательного внешнего вида и легкости восприятия информации;
- простота и прямолинейность в дизайне;
- создание современного и удобного интерфейса.

Тренд простоты в дизайне пришел к нам недавно, но быстро стал культовым направлением. Если раньше было модно делать яркие дизайны с множеством графики, то сейчас стремятся все упрощать.

					БГТУ 03.00.ПЗ			
Изм	Лист	№ документа	Подп.	Дата	Дизайн онлайн-сервиса	Лит.	Лист	Листов
Разраб.	Помоз					у	1	7
Пров.	Осоко					74319008, 2024		
Консульт.								
Н. контр.	Осоко							
Утв.	Романенко							

3.2 Цветовое решение проекта

При определении основного цвета для веб-дизайна необходимо учитывать различные аспекты. Выбранный цвет должен соответствовать концепции, идее и цели проекта, а также оставить первое впечатление на пользователей. Кроме того, цвет может влиять на восприятие пространства: светлые тона создают ощущение воздушности, в то время как темные могут уменьшить визуальное пространство. Поэтому выбор цвета требует особой внимательности.

Эффективное цветовое оформление имеет глубокий смысл и значимость. Использование цвета в дизайне страниц веб-сайта может сыграть ключевую роль в достижении целей проекта, воздействуя на восприятие пользователей веб-страницами.

Для этого сайта были выбраны пять основных цветов, которые представлены на рисунке 3.1.



Рисунок 3.1 – Цветовая схема

Сочетание темно-синего, фиолетового и розового цветов – это стильное и современное решение. Использование этих оттенков придает веб-сайту выразительность и привлекательность. Выбор цвета в дизайне сайта основан на анализе целевой аудитории, конкурентов и предназначении ресурса. Темно-синий и фиолетовый дополняются светлыми оттенками, что придает стильный и элегантный вид. Этот нестандартный подход вызывает положительные эмоции у посетителей.

Данный спектр цветов прекрасно сочетается с общим стилем сайта, придавая ему современность и легкость.

3.3 Типографика

Типографика – это искусство оформления текста, которое базируется на работе со шрифтами и версткой, то есть распределении текста и визуала. Типографика влияет на читабельность и восприятие текста, управляя вниманием читателя [10].

Качество представления текста играет ключевую роль в успехе веб-сайта. Внешний вид ресурса во многом зависит от выбранных шрифтов. Правильно подобранный шрифт способствует ускоренному чтению, превращает текст в элемент композиции, делает его более выразительным и способствует передаче идеи не только содержанием, но и графически.

Для сайта был выбран шрифт Мангоре. Мангоре является шрифтом без засечек, поддерживает латинские и кириллические символы, легко воспринимается и гармонирует со стилистикой сайта. Заголовки будут выглядеть легкими и не перегруженными. Сам шрифт представлен на рисунке 3.2.

		!	"	#	\$	%	&	'	()	*	+	,	-	.	/	0	5	6	7
8	9	:	;	<	=	>	?	@	A	B	C	D	E	F	G	H	I	J	K	L
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_	`	a
b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
w	x	y	z	{		}	~		ı	ç	£	¤	¥	¦	§	¨	©	ª	«	¬
	®	¯	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿	ë	÷
А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф
Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	а	б	в	г	д	е	ж	з	и	й
к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю
я	ё	—	'	'	,	“	”	„	”	†	‡	•	...	‰	<	>	/		0	4

Рисунок 3.2 – Шрифт Мангоре

Начертание шрифта для заголовков является «Bold» сорок восьмого кегля, для подзаголовков – «Medium» тридцать шестого кегля, для основного текста – «Regular», «Light» шестнадцатого кегля.

Начертания представлены на рисунке 3.3.

ExtraLight 200	Whereas recognition of the inherent dignity
Light 300	Whereas recognition of the inherent dignity
Regular 400	Whereas recognition of the inherent dignity
Medium 500	Whereas recognition of the inherent dignity
SemiBold 600	Whereas recognition of the inherent dignity
Bold 700	Whereas recognition of the inherent dignity
ExtraBold 800	Whereas recognition of the inherent dignity

Рисунок 3.3 – Начертания шрифта Мангоре

Был выбран именно этот шрифт, потому что он лучше всего вписался в композицию разрабатываемого ресурса.

3.4 Создание логотипа

Логотип – это фирменный знак, который люди ассоциируют с конкретным брендом [11]. Он играет важную роль в передаче бренду его уникальности, идентифицируя его и выделяя на фоне конкурентов. Логотип может быть сделан на основе изображения, текста или их комбинации, при этом основной целью остается создание узнаваемого и привлекательного образа.

При разработке логотипа для сайта важно учитывать его цели и направление. Простота, привлекательность, читаемость, запоминаемость, универсальность, функциональность и креативность – все эти качества должны быть воплощены в логотипе. Он должен быть несложным, вызывать эмоции, быть легко читаемым и запоминаемым, а также обладать универсальным дизайном и функциональностью. Креативный подход к созданию логотипа позволяет придать ему уникальность и скрытый смысл, делая его визуально привлекательным и запоминающимся.

Логотип был разработан в векторном графическом редакторе – CorelDraw.

Логотип является комбинированным. Логотип состоит из стилизованной буквы «М», которая символизирует онлайн-сервис, а справа от нее расположено название сервиса. Символ «М» является основным элементом, а название сервиса добавлено для уточнения и идентификации. Такой подход позволяет пользователям легко узнавать ваш сервис и ассоциировать его с предоставлением информации о фильмах.

Шрифт был использован Montserrat в начертании Medium. Логотип представлен на рисунке 3.4.



Рисунок 3.4 – Вариации логотипа сайта

Логотип сайта создает ассоциации, соответствующие теме онлайн-сервиса. Важно сохранить свободное пространство вокруг логотипа, чтобы обеспечить его четкое визуальное восприятие и максимальное воздействие на зрителей. Никакие дополнительные элементы не должны пересекать границы свободной зоны логотипа. При размещении логотипа вместе с названием сайта, минимальная ширина для читаемости составляет 40 мм. Так как логотип векторный, его размеры могут быть масштабированы без потери качества, поэтому нет ограничений на максимальные размеры. Минимальные расстояния от логотипа до других элементов представлены на рисунке 3.5.

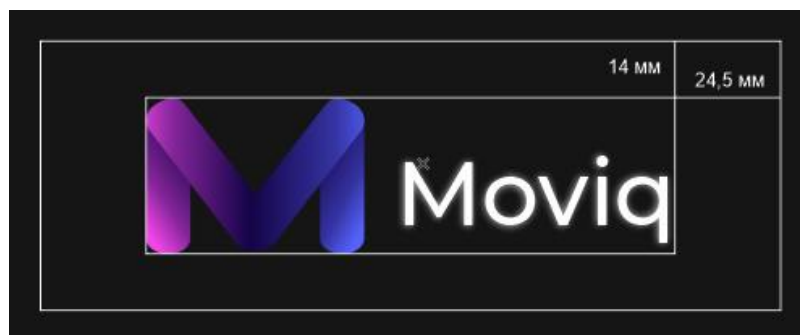


Рисунок 3.5 – Минимальные расстояния от логотипа

Нежелательно использовать фоновые цвета, которые будут сливаться с цветом логотипа, создавая непривлекательный образ и нечитаемый текст, как это представлено на рисунке 3.6.



Рисунок 3.6 – Пример нерекомендуемого фона для логотипа

Логотип лучше всего сочетается с однотонными и темными подложками.

3.5 Разработка дизайна страниц

Дизайн-макет сайта – это визуальный образ будущего сайта, разработанный с учетом технических возможностей HTML верстки. Такой макет является демонстрацией того, как визуально будет выглядеть сайт после верстки и наполнения [12]. Он служит своеобразным чертежом архитектуры сайта, предоставляя представление о том, как будет выглядеть окончательная версия сайта на различных этапах его создания. Использование дизайн-макета сайта обладает несколькими преимуществами, включая возможность выявления визуальных недочетов на ранних стадиях процесса, упрощение работы верстальщика и программиста, предварительное представление о внешнем виде готового сайта, а также сокращение возможных недопониманий между дизайнером и клиентом.

Для создания дизайн-макетов был выбран инструмент Figma, используя имеющиеся прототипы в качестве основы. В палитру цветов были включены оттенки выбранной цветовой схемы онлайн-сервиса.

В начале главной страницы располагается текст, изображение и кнопка для перехода на страницу авторизации. Макет главной страницы веб-сайта представлен на рисунке 3.7. На всех страницах закреплено меню, которое фиксируется при скроллинге страницы.

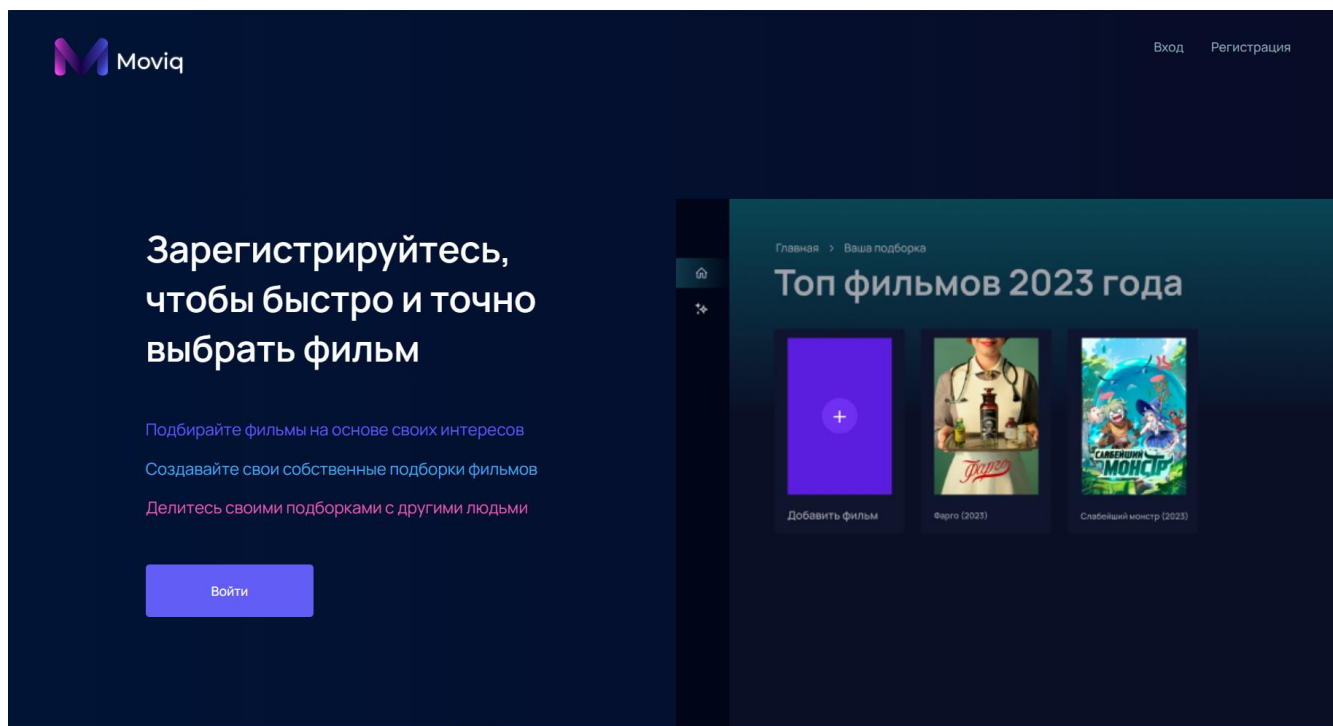


Рисунок 3.7 – Макет главной страницы веб-сайта

Следующей рассматривается страница «Подборки» с набором пользовательских подборок. На странице расположена кнопка с возможностью добавления новой подборки, а также сами подборки. Макет страницы представлен на рисунке 3.8.

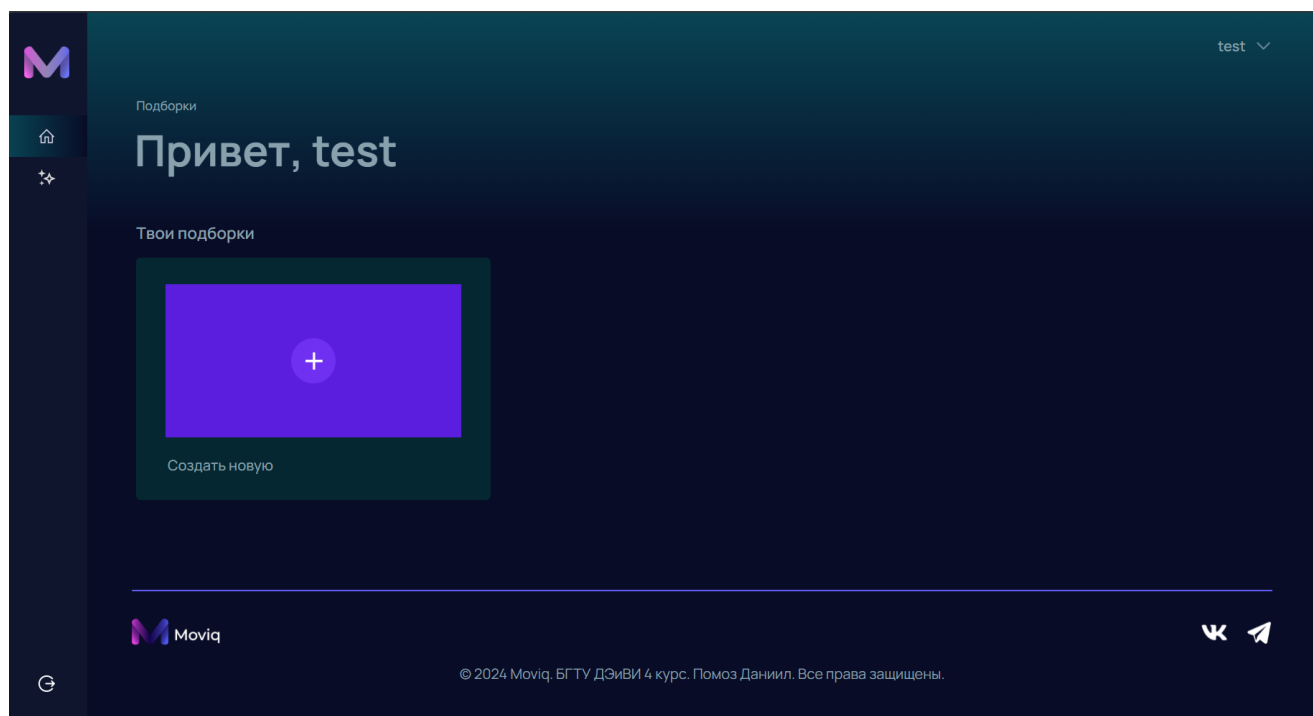


Рисунок 3.8 – Макет страницы «Подборки»

Далее рассмотрена шаблонная страница с отдельно выбранным фильмом, на которой размещена подробная информация, включая постер, название, описание фильма, год, жанр, а также рейтинг.

Также размещена кнопка, при нажатии на которую открывается блок с возможностью выбрать рейтинг. Кроме того, имеется кнопка, при наведении на которую раскрывается список подборок, в которые можно добавить фильм.

Макет страницы с описанием конкретного авто представлен на рисунке 3.9.

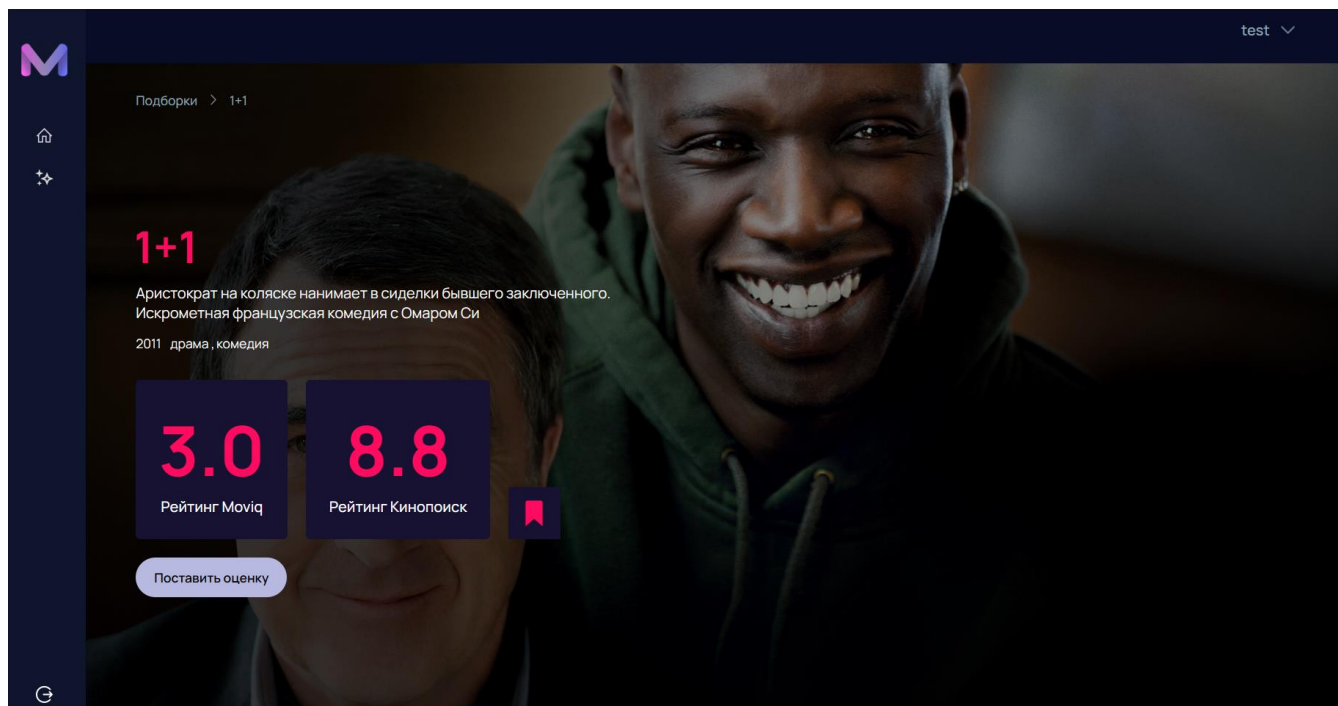


Рисунок 3.9 – Макет страницы с описанием фильма

Макеты основных страниц сайта представлены на плакате БГТУ 05.00 РР.

3.6 Выводы по разделу

В этом разделе был разработан дизайн для веб-сайта, основанный на предварительно созданных прототипах и проведенном анализе конкурентов. Все ключевые элементы навигации на сайте были выделены относительно текста, чтобы привлечь внимание пользователей и улучшить удобство использования. При разработке дизайна был выбран минималистический стиль. Благодаря четкому определению концепции и ее соответствию с логотипом, окончательный дизайн выглядит сбалансированным, гармоничным и завершенным. Этот эффект достигается за счет однородного оформления повторяющихся элементов на сайте, таких как пункты меню, «хлебные крошки», основной текст, заголовки, поля ввода, сообщения об ошибках, а также ссылки и кнопки.

Цветовая схема, логотип и основные элементы дизайна представлены на плакате БГТУ 06.00 РР.

4 Реализация проекта

4.1 Создание и обработка мультимедийного контента

Так как онлайн-сервис предполагает наличие списка фильмов, требовалось подобрать большое количество изображений в хорошем качестве, которые бы были приятны для пользователя.

Изображения были подобраны в одном стиле. Другие изображения, которые присутствуют на сайте, выбирались на стоке freerik.com [13].

Чтобы подобранные изображения сочетались со стилем и оформлением веб-сайта большинство изображений дополнительно обрабатывались в графическом редакторе Adobe Photoshop.

Пример обработанного изображения представлен на рисунке 4.1.

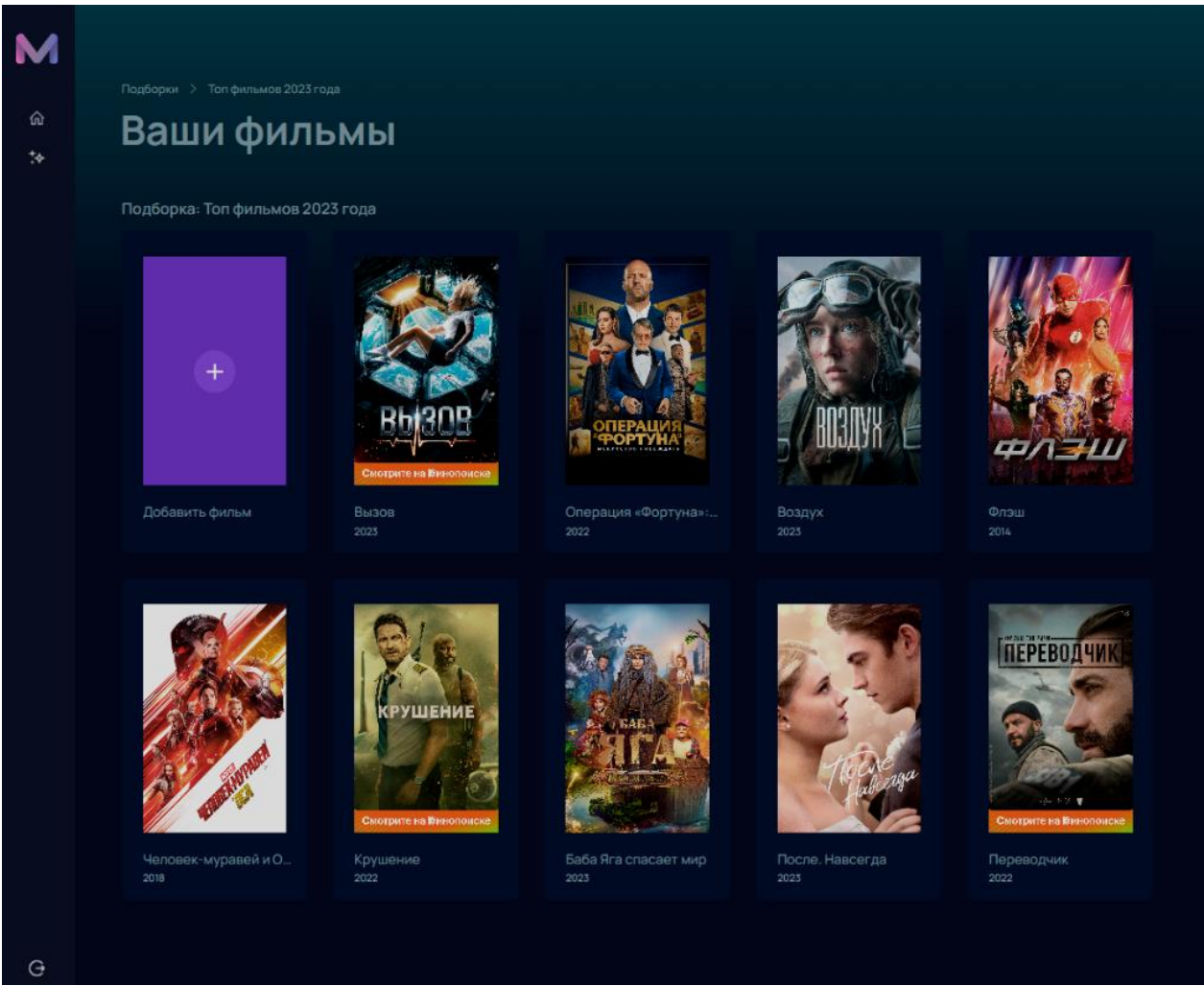


Рисунок 4.1 – Обработанное изображение

					БГТУ 04.00.ПЗ			
Изм	Лист	№ документа	Подп.	Дата	Реализация проекта	Лит.	Лист	Листов
Разраб.	Помоз					у	1	14
Пров.	Осоко					74319008, 2024		
Консульт.								
Н. контр.	Осоко							
Утв.	Романенко							

В основном для редактирования изображений применялись различные фильтры, которые изменяют цветовую схему изображения. Накладывался корректирующий слой «Цветовой тон / Насыщенность».

4.2 Верстка и стилевое оформление проекта

При создании веб-сайта применялся подход семантической верстки, который базируется на значении каждого элемента и структуре документа. Этот подход позволяет создать более понятный и легко поддерживаемый код, так как он основан на использовании семантических тегов, таких как `header`, `footer`, `main` и `section`, для организации различных блоков контента.

Для элементов, которые должны повторно использоваться на странице, такие как меню, футер, страниц фильмов и модальные окна, были созданы отдельные компоненты. Это позволяет упростить и ускорить процесс разработки и обслуживания веб-страниц сервиса.

Маршрутизация между страницами была реализована с помощью пакета `Vue Router`. Маршрутизация описана в файле `index.js`. Она позволяет переключаться между различными страницами в нашем приложении без перезагрузки страницы.

```
const routes = [
  {
    path: '/',
    name: 'Main',
    component: Main,
  },
  {
    path: '/registration',
    name: 'Registration',
    component: Registration,
  },
  {
    path: '/authorization',
    name: 'Authorization',
    component: Authorization
  },
  {
    path: '/collections/:userID?',
    name: 'Collections',
    component: Collections
  }
]
```

Листинг 4.1 – Код файла `index.js`

Для улучшения процесса верстки и оптимизации кода был использован препроцессор `SASS` [14].

Для разметки страниц использовалась двенадцатиколончатая сетка, а также технология `CSS Flexbox` [15].

В листинге 4.2 показано применение технологии `Flexbox`.


```
.collection-container {
  background-color: #052731;
  border-radius: 5px;
  width: 395px;
  height: 240px;
  cursor: pointer;

  display: flex;
  flex-direction: column;
  align-items: center;
  padding-top: 30px;
  transition: .3s;

  &:hover {
    background-color: $mainColor4;
  }
}
```

Листинг 4.2 – Применение Flexbox

Медиа-запросы – это правила CSS, которые позволяют управлять стилями в зависимости от значений технических параметров устройств [16]. Для создания адаптивных блоков с рейтингом на странице фильмов используется свойство `flex-direction: row` для горизонтального расположения элементов. При использовании медиа-запросов это свойство изменяется на такое как `flex-direction: column` для расположения элементов меню по вертикали на мобильных устройствах.

Часть прописанных стилей для мобильной версии представлен на листинге 4.3.

```
@media (max-width: 767px) {

  .rating-container {
    display: flex;
    flex-direction: column;
    margin-top: 20px;
    gap: 10px;
  }
}
```

Листинг 4.3 – Листинг применения медиа-запроса

Благодаря использованию позиционирования по вертикали, получилось создать более удобный и читаемый интерфейс на мобильных устройствах, где вертикальное расположение элементов улучшает доступность и восприятие контента для пользователей.

4.3 Программная реализация визуальных эффектов и элементов дизайна

Основными интерактивными элементами на сайте являются меню и кнопки.

Меню является вертикальным. При наведении на область меню, оно раскрывается вправо и рядом с иконками появляется текст, описывающих их.

Для раскрытия меню требуется на него навести курсор. Для этого для области меню добавлен обработчик события, который при срабатывании изменяет значение переменной, которая отвечает за видимость текста иконок, на true. При отведении курсора из области меню данная переменная изменяет значение на false, скрывая текст иконок. Код разметки для меню представлен на листинге 4.4.

```
<div class="sidebar-menu" v-if="verticalMenu" @mouseover="toggleLabels(true)" @mouseleave="toggleLabels(false)">
  <div class="sidebar-logo">
    </div>
</div>
<div class="menu-icon-container logout">
  <div class="menu-icon">
    <router-link :to="{ name: 'Main'}">
      
      <span class="label" v-if="showRecommendationsLabel">Выйти</span>
    </router-link>
  </div>
</div>
```

Листинг 4.4 – Разметка для меню

Следующим интерактивным элементом является выпадающий список для выбора подборки при добавлении фильма. В раскрытом виде данный список представлен на рисунке 4.2.

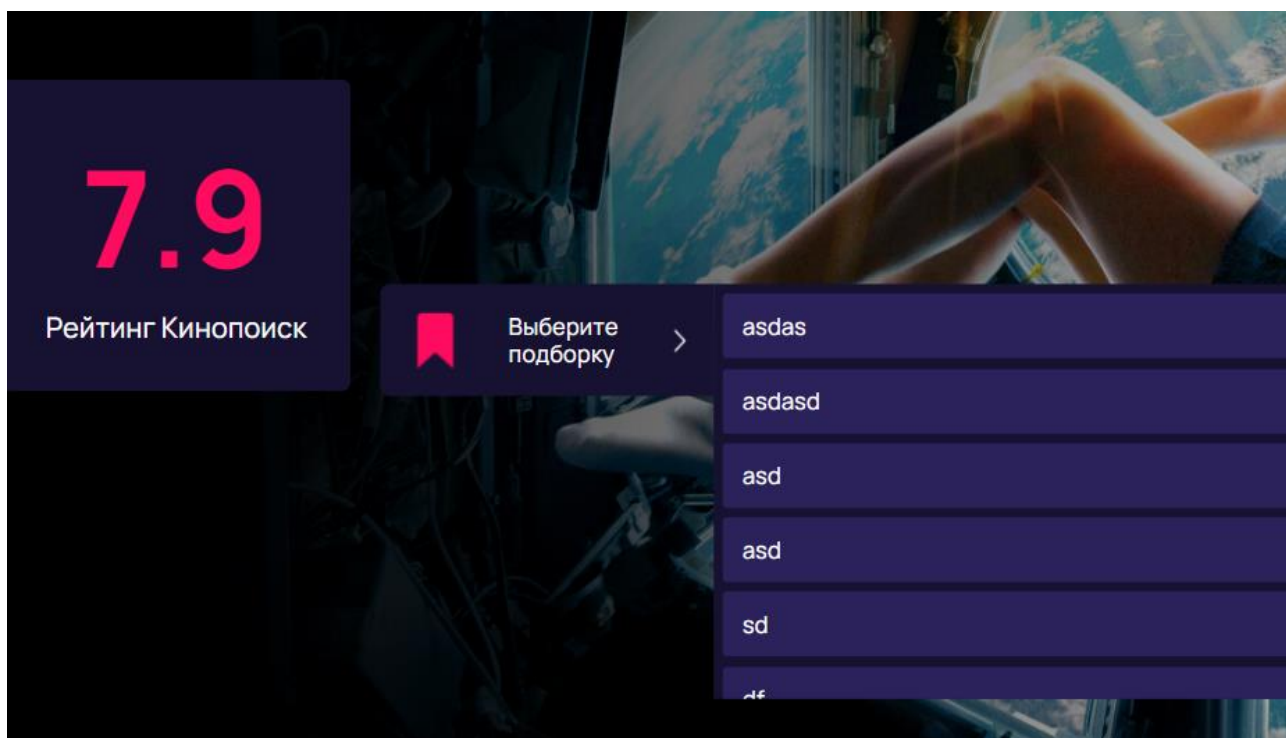


Рисунок 4.2 – Раскрытый выпадающий список

Изначально на блоке с иконкой добавлен обработчик события, который сра-

батывает при наведении. Обработчик события устанавливает для переменной, открывающей следующий дочерний блок, значение `true`. Когда пользователь убирает курсор с блока, переменная устанавливается в `false`. Сами подборки выводятся на страницу с помощью цикла `v-for`, который встроен в Vue фреймворк.

Код выпадающего списка представлен на листинге 4.5.

```
<div class="collection-container" @mouseenter="showTextBlock = true"
@mouseleave="showTextBlock = false">
  
  <div class="text-block-container" v-show="showTextBlock"
@mouseenter="showListBlock = true">
    <div class="list-block" v-show="showListBlock"
@mouseleave="showListBlock = false">
      <div class="collections">
        <div v-for="collectionData in collections"
:key="collectionData.collection.id">
          <div class="collection-data" @click="selectCol-
lection(collectionData.collection.id)">
            <span>{{collectionData.collec-
tion.name}}</span>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

Листинг 4.5 – Разметка для выпадающего меню

При переходе на страницу «Подборки» происходит плавное перемещение блоков с подборками вниз поочередно. Появляется эффект подтягивания блоков. Данная анимация создана путем устанавливания блокам начальных свойств `opacity: 0` и `transform: translateY(-20px)` для их скрытия и перемещения вверх. При загрузке страницы каждому блоку задается новый класс со свойствами `opacity: 1` и `transform: translateY(0)`. Код, позволяющий сделать последовательное появление блоков представлен на листинге 4.6.

```
animateCollections() {
  const collections = document.querySelectorAll('.collection-con-
tainer');
  collections.forEach((collection, index) => {
    setTimeout(() => {
      collection.classList.add('animate');
    }, index * 200);
  });
}
```

Листинг 4.6 – Метод для последовательного появления блоков

Данный метод поочередно добавляет новый класс к каждому блоку с небольшой задержкой для создания эффекта последовательного появления. Для каждого

блока задержка увеличивается на 200 миллисекунд.

Таким образом, каждая из анимаций добавляет интерактивности, что делает сайт более привлекательным. Также анимация помогает пользователю быстрее понять, как и куда переходить. Анимация создает эффект плавности взаимодействия, что в конечном итоге формирует у пользователя положительные эмоции от работы с онлайн-сервисом.

4.4 Реализация функциональных возможностей

При создании серверной части проекта использовался фреймворк Laravel, написанный на языке программирования PHP. Полная физическая структура разработанного проекта представлена в приложении А.

В данном разделе будут подробно рассмотрены несколько ключевых функциональных элементов сайта.

4.4.1 Реализация функционала авторизации

Для авторизации используется метод `getUser` контроллера `UserController.php`. Данный метод представлен на листинге 4.7.

```
public static function getUser(Request $request) {

    $email = $request['email'];
    $password = $request['password'];

    $user = User::where('email', $email)->first();

    $res = [];

    if($user && Hash::check($password, $user->password)) {
        $res[] = [
            "status" => true,
            "type" => 0,
            "message" => "Вход успешен!",
            "user"=>$user
        ];
    } else {
        $res[] = [
            "status" => false,
            "type" => 1,
            "message" => "Пользователя с таким Email не существует
или неправильный пароль",
        ];
    }

    return $res;
}
```

Листинг 4.7 – Метод для авторизации

Данный метод принимает объект запроса `Request`. В методе извлекаются учетные данные `email` и пароль. Проводится проверка существования записи в базе данных, где поле `email` соответствует переменной `$email`. Если аутентификация проходит успешно, сервер возвращает данные пользователя и соответствующее сообщение. В том случае, когда запись в базе данных не найдена, сервер возвращает ответ с сообщением об ошибке.

4.4.2 Реализация расширенной формы регистрации

Расширенная форма регистрации и авторизации представляет собой возможность авторизоваться в сервисе с помощью Google OAuth.

Для этого созданы два метода: `redirectGoogle` и `callbackGoogle`. Метод `redirectGoogle` представлен на листинге 4.8.

```
public static function redirectGoogle() {
    try {
        return Socialite::driver('google')->redirect();
    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}
```

Листинг 4.8 – Метод `redirectGoogle`

Этот метод отвечает за перенаправление пользователя на страницу аутентификации Google. Используется метод `Socialite::driver('google')->redirect()`, который инициирует процесс аутентификации через Google. Если возникает исключение, метод возвращает массив с информацией об ошибке, содержащий статус, тип ошибки и сообщение.

Метод `callbackGoogle` обрабатывает обратный вызов после того, как пользователь аутентифицировался через Google. Получает код авторизации из запроса и использует его для получения ответа с токеном доступа от Google. С помощью токена доступа получает объект пользователя от Google. Далее проверяется, существует ли пользователь в базе данных по `email`. Если пользователь существует и у него нет пароля, считается, что вход успешен. Если пользователь существует и у него есть пароль, выводится сообщение о необходимости стандартного входа. Если пользователь не существует, создается новый пользователь с полученными данными (`nickname` и `email`), и ролью по умолчанию.

Метод `callbackGoogle` представлен на листинге Б.1.

4.4.3 Реализация личного кабинета пользователя

Для реализации редактирования данных пользователя в его личном кабинете, используется код, представленный на листинге Б.2.

Метод принимает объект `Request`, из которого извлекаются данные пользователя, такие как `userID`, `name`, `email`, `old_email`, `password`.

Метод проверяет, существует ли в базе данных другой пользователь с таким же `email`, но с другим `id`. Если такой пользователь существует, возвращается ответ с сообщением об ошибке. Если такого пользователя не существует, метод находит текущего пользователя по `id`. Если пользователь найден, проверяются и обновляются переданные данные. Если пользователь с переданным `id` не найден, возвращается ответ с сообщением об ошибке.

4.4.4 Реализация административной панели

Для управления таблицами в базе используется контроллер `AdminController`, в котором определены методы для добавления, редактирования и удаления данных в сервисе. Код метода для добавления новой страны представлен на листинге 4.9.

```
public static function addCountry(Request $request) {
    try {
        $newCountry = $request['newCountry'];
        $addCountry = Countrie::where('country', $newCountry)-
>first();
        if (!$addCountry) {
            $country = Countrie::create([
                'country' => $newCountry,
            ]);
            if (!$country) {
                throw new \InvalidArgumentException("Произошла
ошибка при добавлении страны");
            }
        } else {
            throw new \InvalidArgumentException("Страна уже суще-
ствует");
        }
        return [
            "status" => true,
            "type" => 0,
            "message" => 'Добавлена новая страна',
        ];
    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}
```

Листинг 4.9 – Код метода для добавления новой страны

Этот метод используется для добавления новой страны в базу данных. Метод начинается с извлечения названия новой страны из объекта запроса `$request`. Это значение присваивается переменной `$newCountry`.

Затем метод проверяет, существует ли уже такая страна в базе данных. Это делается с помощью запроса к модели `Countrie`, где ищется запись с полем `country`, равным `$newCountry`. Если такая запись найдена, результат сохраняется в переменной `$addCountry`. Если страна не найдена в базе данных, создаётся новая запись в таблице `Countrie` с полем `country`, равным `$newCountry`. После этого проверяется, успешно ли была добавлена новая запись. Если добавление не удалось, выбрасывается исключение с сообщением об ошибке. Если страна уже существует в базе данных, также выбрасывается исключение. Если страна успешно добавлена, метод возвращает массив с соответствующими данными.

Остальные методы данного контроллера представлены на листинге Б.3.

4.4.5 Реализация панели модератора

Для реализации обновления статуса на предложенные фильмы используется код, представленный на листинге 4.10.

```
public static function postAnswerForAddMovie(Request $request) {
    try {
        $userID = $request['userID'];
        $ticketID = $request['ticketID'];
        $status = $request['status'];

        $suggestions = AddMovie::where('user_id', $userID)-
>where('id', $ticketID)->first();
        if ($suggestions) {
            $suggestions->status = $status;
            $suggestions->save();

            $res = [
                "status" => true,
                "type" => 0,
                "message" => "200 ok",
            ];
        } else {
            $res = [
                "status" => false,
                "type" => 1,
                "message" => "Произошла ошибка"
            ];
        }
        return $res;
    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}
```

Листинг 4.10 – Код обновления статуса на предложенные фильмы

Метод извлекает `userID`, `ticketID` и `status` из объекта запроса `$request`. Выполняется запрос к модели `AddMovie`, чтобы найти запись с `user_id`, равным `$userID`, и `id`, равным `$ticketID`. Результат сохраняется в переменной `$suggestions`. Если предложение найдено, его поле `status` обновляется значением из запроса `$status`, после чего запись сохраняется в базе данных с помощью метода `save`. В случае успешного обновления, возвращается массив данных с соответствующим сообщением.

4.4.6 Реализация модуля добавления описания фильмов

Пользователь может предложить добавить фильм для сервиса в своем личном кабинете. Для этого реализован метод `suggestMovie`. Код данного метода представлен на листинге Б.4.

Метод извлекает различные параметры фильма из объекта запроса `$request` с использованием оператора `??`, чтобы установить значения по умолчанию на `null`, если параметры отсутствуют. Если параметр `userMovieSelectedTypeOption` присутствует, метод ищет соответствующую запись в таблице `Type`. Если запись найдена, устанавливается `typeID`. В противном случае выбрасывается исключение. Если параметр `userMovieSelectedRestrictOption` присутствует, метод ищет соответствующую запись в таблице `Restrict`. Если запись найдена, устанавливается `restrictID`. В противном случае выбрасывается исключение. Метод создает новую запись в таблице `AddMovie` с переданными параметрами. Поля `type_id`, `restrict_id` и `user_id` также устанавливаются, если они были найдены ранее. Если запись не была создана, выбрасывается исключение. Метод прикрепляет `genreID` и `countryID` к созданной записи, если они существуют. В случае успешного выполнения метод возвращает ответ с сообщением об успешном выполнении. В случае ошибки метод возвращает ответ с сообщением об ошибке.

4.4.7 Реализация модуля актуализации базы данных

База данных строится на основе существующего API сервиса «Кинопоиск» [17]. Для этого производится GET запрос к API, чтобы получить данные о фильмах, используя текущий номер страницы, лимит записей и выбранные поля.

В методе `getAllMovies` на первом этапе метод формирует строку `$fields`, содержащую список полей, которые будут выбраны из внешнего источника данных. Этот список включает в себя идентификатор фильма, его название, описание, рейтинг, жанры, страны производства, постеры, а также другую информацию о фильме. Затем метод получает текущий номер страницы для запроса данных о фильмах из базы данных. Если такой записи не существует, создается новая запись с номером страницы «1». Следующим шагом выполняется HTTP-запрос к внешнему API с использованием сформированного URL, который включает текущий номер страницы и список выбранных полей. Если запрос успешен и API возвращает данные, происходит попытка добавления этих данных в базу данных. Если API сообщает о завершении загрузки данных, это также учитывается.

Фрагмент запроса к API представлен на листинге 4.11.

```
$response =
Http::get("https://api.kinopoisk.dev/v1.4/movie?page={$currentPage}&limit=250&selectFields={$fields}");
```

Листинг 4.11 – Фрагмент запроса к API

Метод `addMoviesToDB` принимает массив фильмов и добавляет каждый из них в базу данных Laravel. Для каждого из фильма из массива происходит извлечение необходимых данных о фильме и связанных с ним объектах, таких как жанры и страны. Далее происходит создание новой записи о фильме в базе данных, а также добавление связанных записей в соответствующие таблицы базы данных. В случае возникновения ошибки при добавлении фильма, процесс продолжается с остальными фильмами из массива.

Метод `extractMovieData` извлекает данные о фильме из массива, полученного от API, и подготавливает их для добавления в базу данных. Он также определяет связанные записи, такие как тип и ограничения.

Кроме того, множество вспомогательных методов используются для обработки различных аспектов данных о фильмах. Например, методы `addGenresToMovie` и `addCountriesToMovie` предназначены для добавления информации о жанрах и странах производства фильма в базу данных. Код метода `addGenresToMovie` представлен на листинге 4.12.

```
private function addGenresToMovie($movie, $createdMovie)
{
    if (isset($movie['genres'])) {
        foreach ($movie['genres'] as $genre) {
            $genreRecord = $this->getOrCreateRelatedRecord(Genre::class, 'name', $genre['name']);
            $createdMovie->genres()->attach($genreRecord->id);
        }
    }
}
```

Листинг 4.12 – Метод `addGenresToMovie`

Есть также дополнительные методы для обновления информации о типах фильмов, странах и жанрах, полученных из внешнего API, и добавления их в базу данных, если они отсутствуют.

Для актуализации базы данных была создана команда консоли Laravel, которая выполняет обновление базы данных. Команда находится в пространстве имен `App\Console\Commands`.

В свойство `signature` добавляется имя команды, в данном случае имя – `app:update-database-command`.

Когда команда запускается, вызывается метод `handle`. Внутри метода создается новый экземпляр класса контроллера `DBController`. Вызывается метод `getAllMovies` этого контроллера для обновления базы данных фильмов. Результат вызова записывается в переменную `result`. Затем вызываются методы `updateMovieTypes`, `updateMovieCountries` и `updateMovieGenres` для обновления

информации о типах фильмов, странах и жанрах. Данная команда выполняется с помощью планировщика заданий Windows в определенное время суток при запущенном сервере. Метод `handle` представлен на листинге 4.13.

```
public function handle()
{
    $dbController = new DBController();
    $result = $dbController->getAllMovies();

    $this->info(json_encode($result));

    $dbController->updateMovieTypes();
    $dbController->updateMovieCountries();
    $dbController->updateMovieGenres();
}
```

Листинг 4.13 – Метод `handle`

Таким образом, процесс обновления базы данных фильмов из внешнего API включает в себя получение данных, их обработку и добавление в базу данных Laravel с использованием соответствующих методов контроллера. Создание команды консоли Laravel обеспечивает автоматическое обновление базы данных, что упрощает и автоматизирует процесс обслуживания приложения.

4.4.8 Расчет рейтинга фильмов

На странице фильма у каждого фильма рассчитывается рейтинг. Рейтинг выводится как от сервиса «Кинопоиск», так и рассчитывается для разрабатываемого сервиса. Каждый пользователь может поставить только одну оценку на один фильм, при этом пользователь может изменить свою оценку.

В листинге 4.14 представлен фрагмент действия в методе контроллера `MoviesController.php` для расчета рейтинга фильма.

```
$existingRatingsCount = Rating::where('movie_id', $id)->count();

if(!$existingRatingsCount) {
    return [
        "status" => true,
        "type" => 0,
        "message" => "Рейтинга еще нет",
        "data" => 'Рейтинга еще нет',
    ];
}

$existingRatingsSum = Rating::where('movie_id', $id)
    ->sum('rating');

$newRatingSum = $existingRatingsSum;
$newRatingsCount = $existingRatingsCount;
```

```

$averageRating = $newRatingSum / $newRatingsCount;
$averageRating = round($averageRating, 1);

return [
    "status" => true,
    "type" => 0,
    "message" => "Рейтинг успешно просчитан",
    "data" => $averageRating,
];

```

Листинг 4.14 – Расчет рейтинга фильма

В данном коде осуществляется подсчет записей, которые записываются в переменную. Если рейтинга еще нет, то функция возвращает соответствующее сообщение и завершает выполнение. Если записи в таблице имеются, то сумма всех значений рейтингов для фильма сохраняется в отдельной переменной.

Далее среднее значение рейтинга рассчитывается путем деления суммы всех рейтингов на количество рейтингов. Затем результат округляется до одного знака после запятой, и функция возвращает сообщение с данными о рейтинге и завершает выполнение.

4.4.9 Реализация формирования рекомендаций для пользователей

При переходе на страницу «Рекомендации» пользователю отображается фильм. Для этого реализован отдельный метод в контроллере `MoviesController.php`. Фрагмент реализации данной функциональности представлен на листинге 4.15.

```

$recommendedMovies = Movie::select('movies.*', 'posters.url
as poster', 'backdrops.url as backdrop', 'logos.url as logo')
->leftJoin('ratings', 'movies.id', '=', 'ratings.movie_id')
->leftJoin('posters', 'movies.id', '=', 'posters.movie_id')
->leftJoin('backdrops', 'movies.id', '=', 'backdrops.movie_id')
->leftJoin('logos', 'movies.id', '=', 'logos.movie_id')
->with('genres')
->with('restrict')
->with('ratingKp')
->with('type')
->with('countries')
->groupBy('movies.id', 'posters.url', 'backdrops.url',
'logos.url')
->havingRaw('COUNT(ratings.id) > 10 OR SUM(movies.votesKp) >
100')
->orderByDesc('votesKp')
->orderByDesc('ratingKp')

->paginate(1, ['*'], 'page', $currentPage);

```

Листинг 4.15 – Реализация рекомендаций фильмов

Этот код выполняет запрос к базе данных, чтобы выбрать рекомендованные фильмы с определенными критериями.

Изначально осуществляется выборка всех столбцов из таблицы с фильмами. Далее используются `LeftJoin` для объединения таблиц, а также используется метод `with` для загрузки связанных данных.

Результаты группируются, чтобы выбрать только те записи, для которых количество рейтингов больше 20 или сумма голосов на сервисе «Кинопоиск» больше 100. Записи сортируются по убыванию значений столбцов с голосами и происходит пагинация результатов. Результаты разбиваются на страницы и выбирается одна страница по номеру текущей страницы.

4.5 Выводы по разделу

В данном разделе представлены основные этапы создания онлайн-сервиса «Moviq». Особое внимание уделено верстке и стилевому оформлению проекта, используя семантическую верстку, компонентный подход, маршрутизацию и препроцессор SASS для удобства разработки и поддержки кода. Применение технологий Flexbox и CSS3 медиа-запросов обеспечивает адаптивность и гибкость интерфейса на различных устройствах.

Программная реализация визуальных эффектов и элементов дизайна добавляет интерактивности и привлекательности, включая анимацию меню, кнопок и блоков с подборками фильмов.

Создание серверной части проекта с использованием фреймворка Laravel обеспечивает удобство работы с базой данных, маршрутизацию и автоматизацию повторяющихся задач. Процесс обновления базы данных из внешнего API с помощью команды консоли Laravel обеспечивает актуальность данных и упрощает обслуживание приложения.

В результате выполнения всех этапов проекта был создан функциональный и привлекательный онлайн-сервис «Moviq», соответствующий современным технологиям и требованиям.

5 Особенности использования и продвижения проекта

5.1 Руководство пользователя

Руководство пользователя является важным инструментом для обеспечения эффективного использования программного продукта. В зависимости от целей и особенностей пользователя, такие руководства могут быть представлены в двух основных форматах: сценарном и описательном.

Для онлайн-сервиса «Moviq» было разработано руководство пользователя сценарного типа.

Для того, чтобы пользоваться сервисом, пользователь должен авторизоваться. Авторизоваться пользователь может как с помощью Google, так и стандартным способом. Для авторизации с помощью Google, пользователю необходимо нажать на кнопку, представленную на рисунке 5.1.

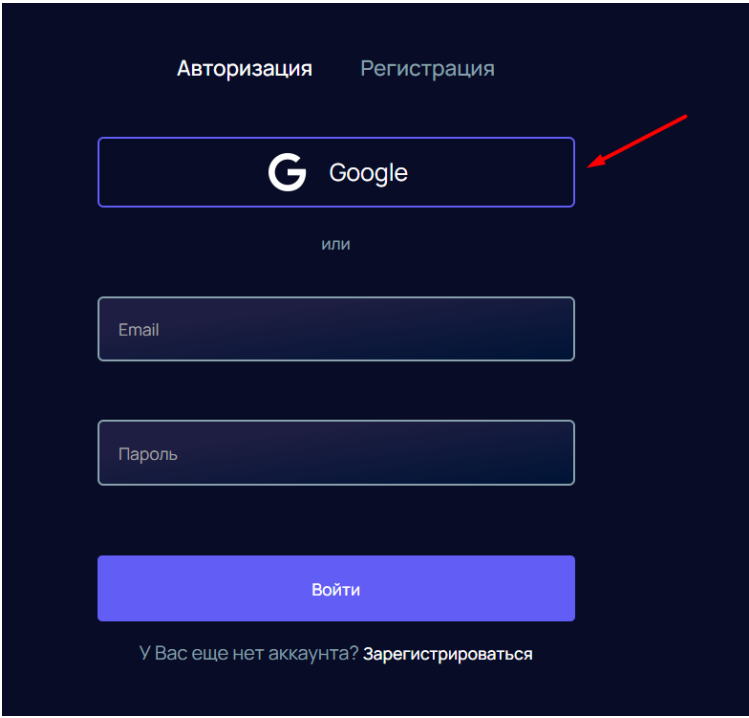
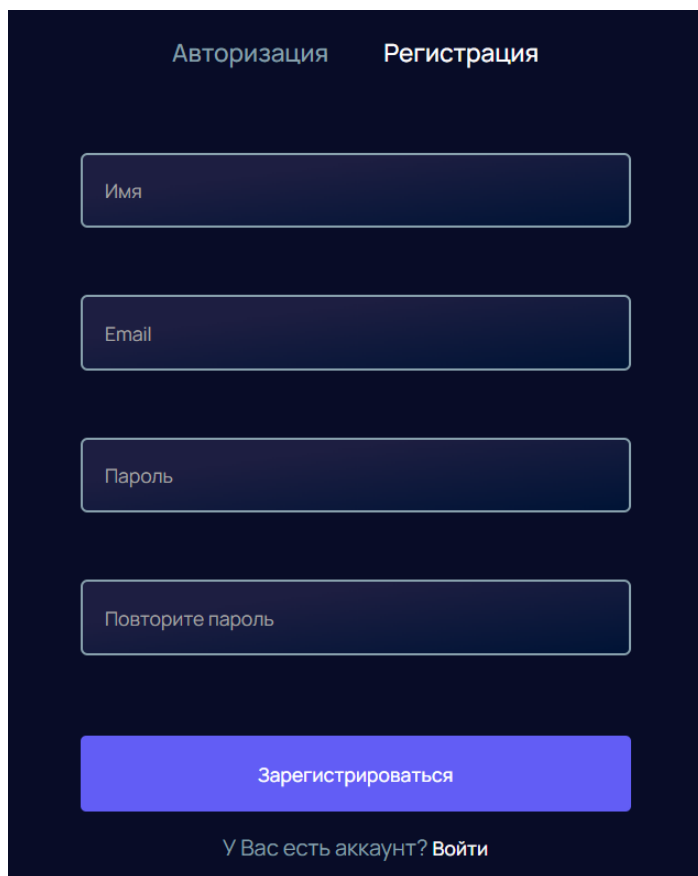


Рисунок 5.1 – Кнопка авторизации через Google

Для авторизации стандартным способом, пользователь должен ввести email и пароль в соответствующие поля на странице авторизации.

Если пользователь еще не зарегистрирован в сервисе, он может зарегистрироваться стандартным способом на странице регистрации аккаунта. Для этого пользователь должен заполнить поля с именем, email, паролем, а также поле с повторением пароля (рисунок 5.2).

					БГТУ 05.00.ПЗ			
Изм	Лист	№ документа	Подп.	Дата	Особенности использования и продвижения проекта	Лит.	Лист	Листов
Разраб.	Помоз					у	1	11
Пров.	Осоко					74319008, 2024		
Консульт.								
Н. контр.	Осоко							
Утв.	Романенко							



Авторизация Регистрация

Имя

Email

Пароль

Повторите пароль

Зарегистрироваться

У Вас есть аккаунт? Войти

Рисунок 5.2 – Поля для заполнения при регистрации

Для того, чтобы добавить фильм в подборку пользователю требуется попасть на страницу «Подборки». Далее требуется нажать кнопку «Создать новую». Данная кнопка представлена на рисунке 5.3.

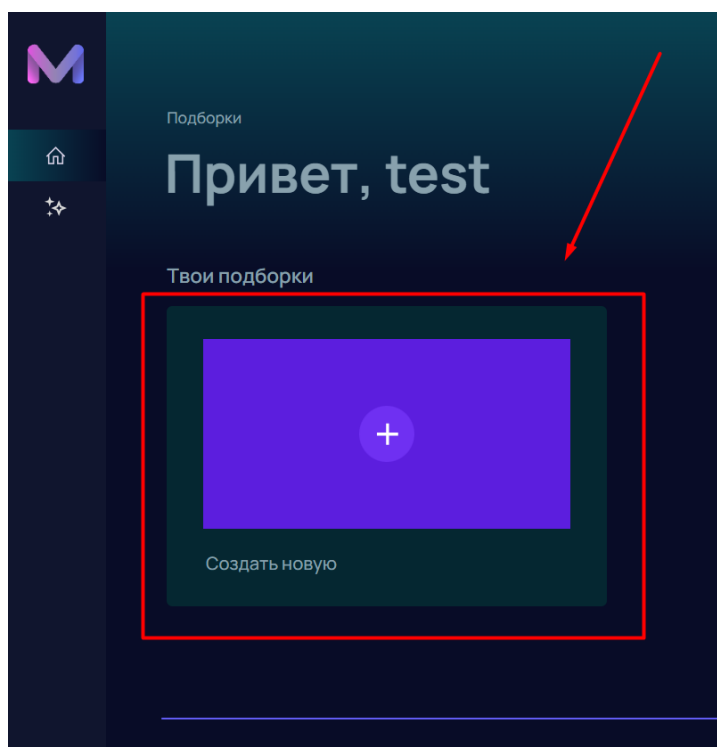


Рисунок 5.3 – Кнопка создания новой подборки

Далее требуется заполнить поля названия и описания и нажать кнопку «Добавить». Для добавления фильма в подборку требуется перейти в подборку на странице «Подборки», затем нажать по кнопке «Добавить фильм». В открывшемся модальном окне требуется выбрать необходимый фильм и нажать кнопку «Добавить». Данное модельное окно представлено на рисунке 5.4.

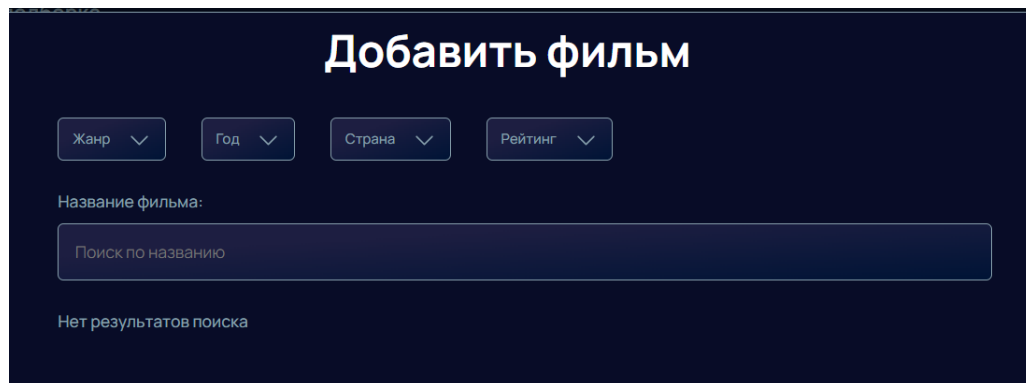


Рисунок 5.4 – Модальное окно для добавления фильма в подборку

Добавить фильм в подборку можно как со страницы требуемой подборки, так и со страницы «Рекомендации».

Чтобы добавить фильм в подборку на странице «Рекомендации», требуется навести курсор на соответствующую кнопку и выбрать необходимую подборку.

Данная кнопка представлена на рисунке 5.5.

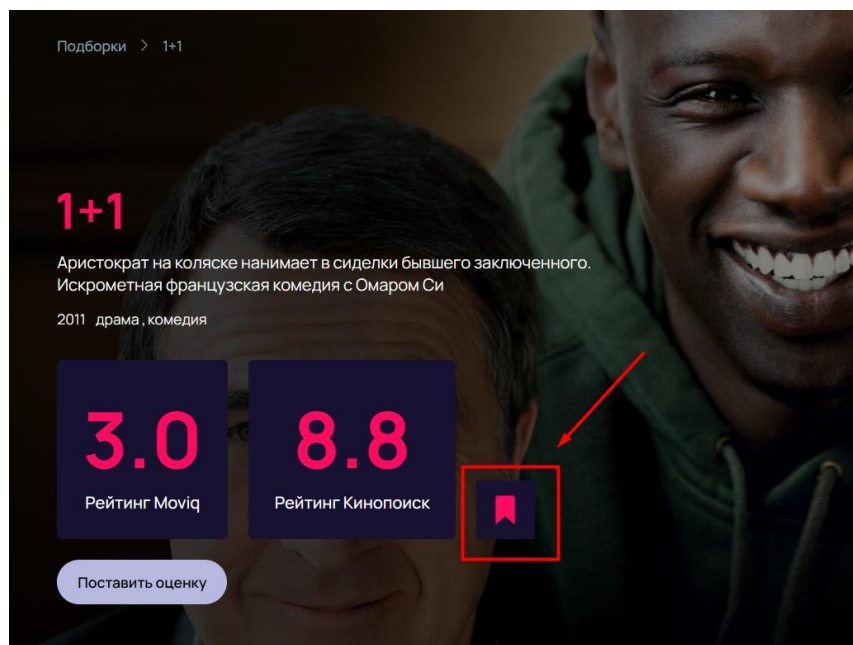


Рисунок 5.5 – Кнопка для добавления фильма в подборку

После добавления фильма в подборку данный фильм можно добавить также в другую подборку, созданную пользователем.

Администратор может управлять данными на сервисе. Для перехода на страницу администратора пользователь должен нажать соответствующую кнопку в меню. На данной странице администратор может управлять данными. Например,

для добавления нового фильма, пользователь должен заполнить необходимые поля данными о фильме. Панель администратора представлена на рисунке 5.6.

Рисунок 5.6 – Панель администратора

Управление другими данными на сервисе происходит аналогичным образом. Для этого пользователь должен выбрать необходимый пункт и заполнить поля.

5.2 Тестирование

Тестирование – это процесс проверки программного обеспечения, системы или приложения на соответствие требованиям и оценки их качества [18]. Тестирование включает различные виды проверок. Каждая проверка оценивает работу системы при определённых условиях и с определёнными входными данными.

5.2.1 Тестирование навигации

На сайте была проведена тщательная проверка всех ссылок и кнопок, и каких-либо недочётов выявлено не было. Все интерактивные элементы сайта, с которыми пользователь должен взаимодействовать, ведут себя предсказуемо и информативно. Например, при наведении курсора на ссылку фон кнопки изменяется, а курсор меняет свой вид. На рисунке 5.7 показана кнопка, при наведении на которую изменяется фон и увеличивается размер, демонстрируя данный функционал.



а

б

Рисунок 5.7 – Реагирование кнопки на наведение курсора:
а – до наведения; *б* – после наведения.

Для упрощения навигации по страницам сайта были созданы хлебные крошки, как показано на рисунке 5.8. Хлебные крошки позволяют пользователю легко отслеживать свое местоположение на сайте и быстро переходить к предыдущим страницам. Важно отметить, что последний элемент цепочки хлебных крошек не является ссылкой, что соответствует лучшим практикам веб-дизайна.

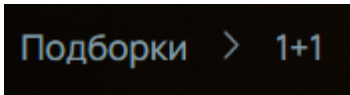
The image shows a dark rectangular box containing the text 'Подборки > 1+1' in a light blue font. This represents a breadcrumb trail on a website.

Рисунок 5.8 – Хлебные крошки

Вертикальное меню сайта закреплено при прокрутке страницы, чтобы облегчить пользователям взаимодействие с интерфейсом. Меню остается видимым и доступным независимо от положения на странице, что повышает удобство навигации.

5.2.2 Корректная работа с формами и валидация

Валидация форм на веб-сайте играет ключевую роль в обеспечении правильности вводимых пользователем данных. Она позволяет убедиться, что данные, передаваемые через формы, соответствуют заданным критериям и ограничениям. Это обеспечивает безопасность и производительность приложения.

На проекте было принято решение внедрить валидацию форм для облегчения работы с ними. Были добавлены проверки формата email и пароля, а также дополнительные проверки, например, проверка длины введенного названия.

Для каждого поля была введена проверка на наличие пробела вначале введенных данных. Система сама удаляет пробел в начале текстового поля, если его находит. Ошибки валидации отображаются пользователю красным цветом под соответствующим полем ввода, что делает процесс ввода данных для пользователей более прозрачным и понятным.

Форма авторизации изображена на рисунке 5.9.

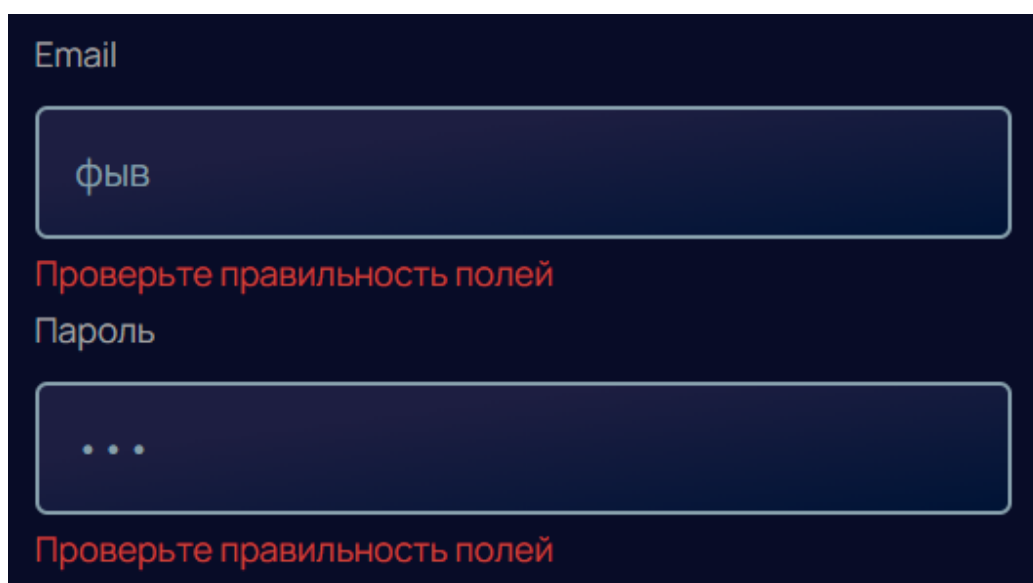
The image shows a login form on a dark background. It has two input fields: 'Email' and 'Пароль'. The 'Email' field contains the text 'фыв' and has a red error message 'Проверьте правильность полей' below it. The 'Пароль' field contains three dots and also has a red error message 'Проверьте правильность полей' below it.

Рисунок 5.9 – Подсвечивание полей с ошибками

В данном случае пользователь заполнил поля в неправильном формате, которые являются обязательными к заполнению, поэтому все неверно заполненные поля формы сопровождаются текстовым пояснением.

5.2.3 Кроссбраузерное тестирование

Для проведения кроссбраузерного тестирования веб-сайта были использованы браузеры Microsoft Edge, Google Chrome и Яндекс.

На рисунке 5.10 показано отображение сайта в браузере Microsoft Edge.

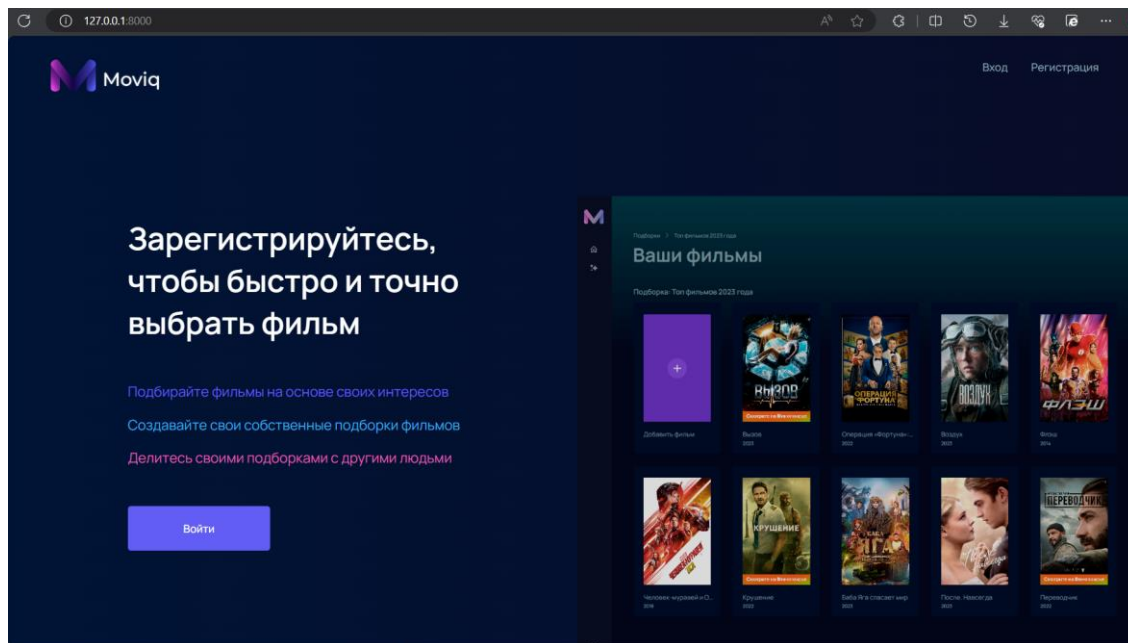


Рисунок 5.10 – Отображение сайта в Microsoft Edge

Отображение сайта в Google Chrome представлено на рисунке 5.11.

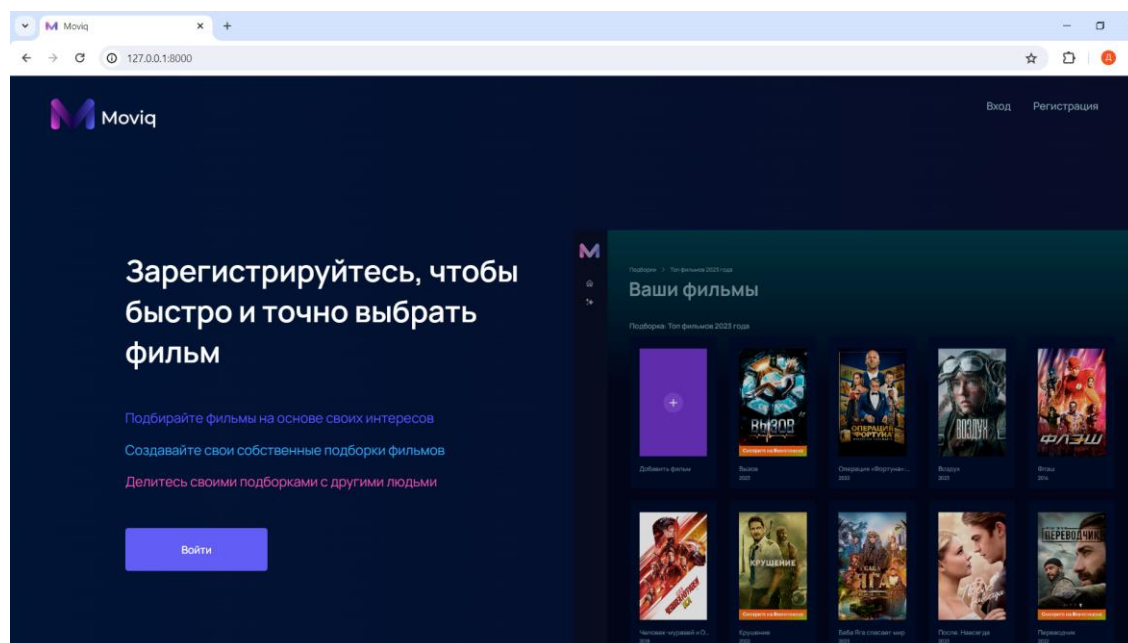


Рисунок 5.11 – Отображение сайта в Google Chrome

Отображение сайта в Яндексе представлено на рисунке 5.12.

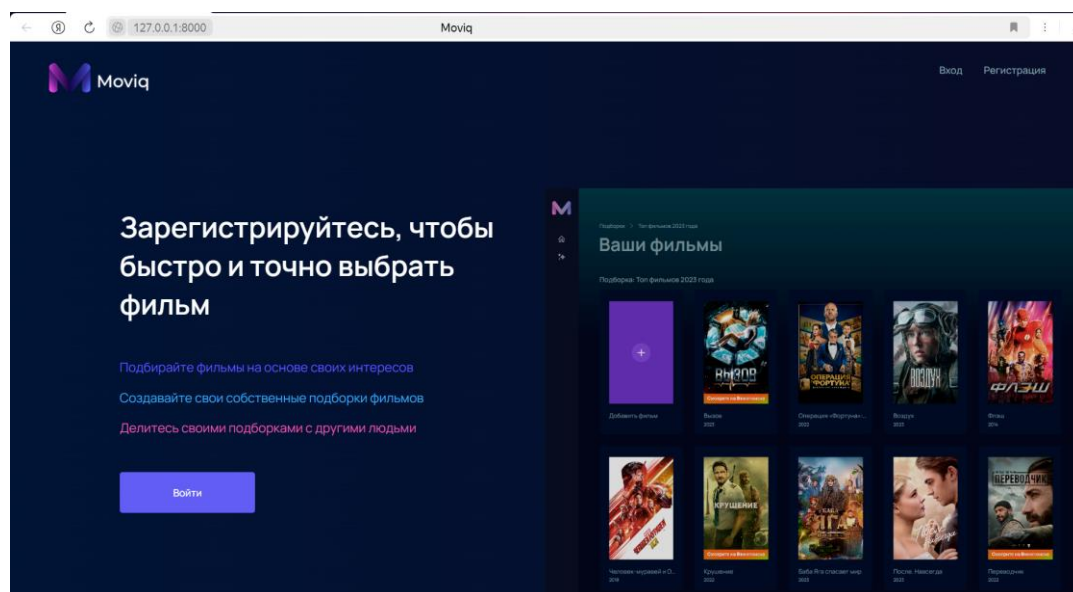


Рисунок 5.12 – Отображение сайта в Яндексе

Как видно на изображениях, шрифты и блоки остаются на своих местах, блоки одинаковы на всех трех браузерах, а, значит, сайт является кроссбраузерным.

5.2.4 Тестирование адаптивности

Разработанный сайт является адаптивным, что позволяет ему корректно отображаться на устройствах с различными разрешениями экрана. Это достигается за счёт динамического изменения позиционирования и размеров блоков на сайте в зависимости от размеров экрана пользователя.

Тестирование адаптивности онлайн-сервиса проводилось для экрана с разрешением 1920 на 1080 пикселей (рисунок 5.13), 768 на 1024 пикселей (рисунок 5.14) и для разрешения 360 на 640 пикселей (рисунок 5.15).

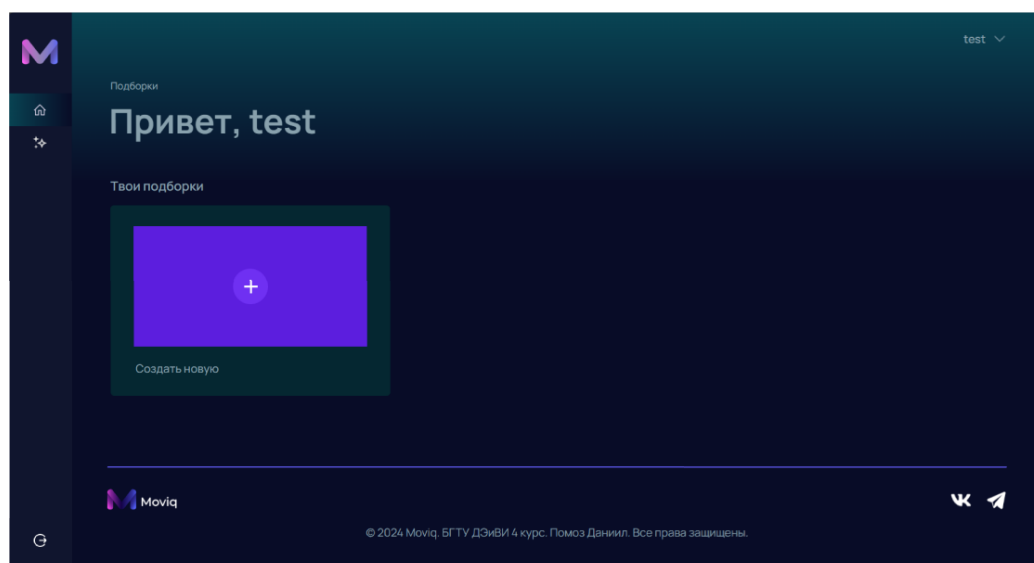


Рисунок 5.13 – Отображение для экрана с разрешением 1920 на 1080 пикселей.

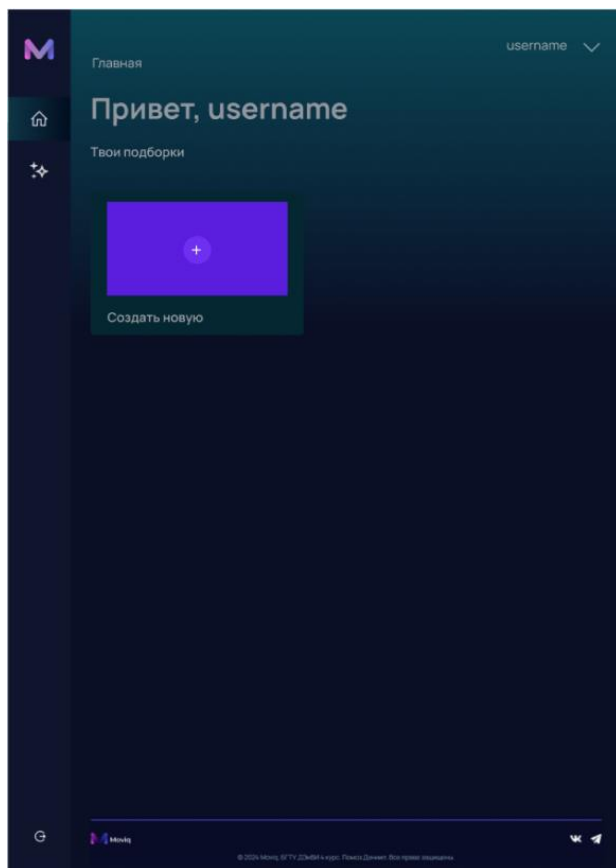


Рисунок 5.14 – Отображение для экрана с разрешением 768 на 1024 пикселей.

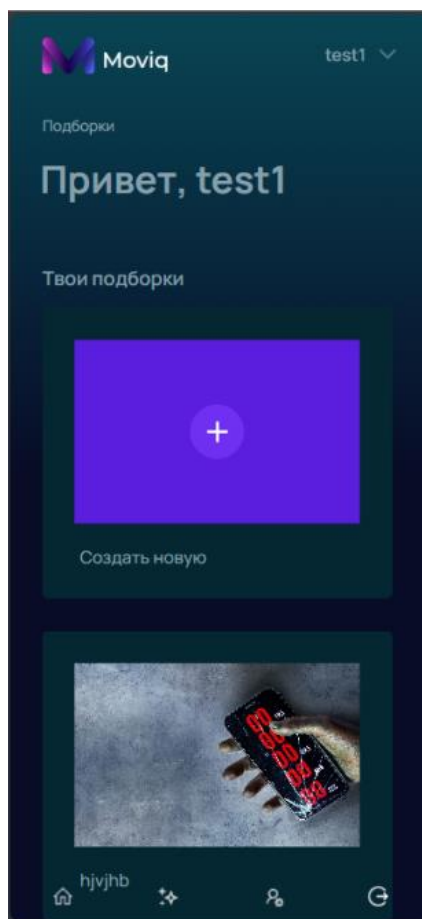


Рисунок 5.15 – Отображение для экрана с разрешением 360 на 640 пикселей.

При просмотре сайта на устройствах с меньшим разрешением меню автоматически позиционируется снизу по горизонтали, что улучшает навигацию и обеспечивает доступ ко всем разделам сайта. Такое решение гарантирует, что пользователи смогут комфортно работать с сайтом, независимо от того, используют ли они настольный компьютер, планшет или смартфон.

5.2.5 UX/UI тестирование

UX/UI тестирование – это один из этапов тестирования, направленный на оценку удобства и эффективности использования веб-приложения или сайта с точки зрения пользователя.

В ходе этого тестирования проверяются все аспекты взаимодействия пользователя с интерфейсом, включая навигацию, расположение элементов, визуальную привлекательность и общий пользовательский опыт.

Для оценки дизайна сайта были задействованы пять человек с разным возрастом и уровнем компьютерной грамотности.

Их описание приведено в таблице 5.1.

Таблица 5.1 – Описание респондентов

Респондент	Уровень компьютерной грамотности	Возраст	Уровень эмоциональной открытости
Александр	3	55	4
Мария	4	35	2
Станислав	5	20	5
Кирилл	2	26	4
Владислав	4	18	2

Метод анализа пользовательского восприятия дизайна помогает определить, вызывает ли дизайн нужные эмоции у пользователей.

Вместе с макетом интерфейса респондентам предоставляется список прилагательных, которые должны характеризовать дизайн сайта и описывать его визуальные и эмоциональные аспекты.

В опросе использовались такие характеристики, как: эффективный, непредсказуемый, понятный, глупый, дружелюбный, чистый, недружелюбный, яркий, удобный, ненадежный, управляемый, раздражающий, непоследовательный, теплый, хороший, плохой, интуитивный, круглый, холодный, комфортабельный, умный, нестандартный, безопасный, современный, любительский, радостный, неприятный, запутанный, последовательный, тусклый, опасный, простой, гибкий, красивый, некрасивый, легкий, привлекательный, нечеткий, жесткий, профессиональный, темный, стандартный, халтурный, интересный, надежный, тяжелый, эффективный, печальный, неудобный, радостный, ясный, треугольный, загадочный, привлекательный, сложный, светлый, неэффективный, медленный.

Результаты анкетирования представлены в таблице 5.2.

Таблица 5.2 – Результат анкеты

Респондент	Прилагательные
Александр	Гибкий, понятный, современный, привлекательный, удобный, ясный
Мария	Полезный, интуитивный, красивый, непредсказуемый
Станислав	Легкий, удобный, интересный, чистый, надежный
Кирилл	Современный, прямой, качественный, безопасный, тяжелый
Владислав	Любительский, скучный, сложный, умный, управляемый

По итогам проведенного анализа сайт был описан пятнадцатью положительными прилагательными, шестью нейтральными и четырьмя отрицательным. Из чего можно сделать вывод, что сайт создает хорошее впечатление у посетителей.

Дальнейшее исследование функциональности системы включает проведение тестовых сценариев. Каждый тестовый сценарий представляет собой проверку определенного аспекта системы, задачу, которую респондент должен выполнить.

В рамках данного тестирования были назначены три респондента. Каждому из них была предложена конкретная задача, формулировка которой была ясной, лаконичной и лишена подсказок. Каждая задача также имеет определенные критерии начала выполнения и успешного завершения.

Сам сценарий включает в себя описание задачи пользователя, определение метрик и выполнение тестового задания.

Задача 1. Респондентам необходимо добавить новую подборку. Метрики: успешность, удовлетворенность.

Сценарий №1 для выполнения задания:

Пользователь проходит регистрацию, после чего попадает на страницу «Подборки». Далее он нажимает кнопку «Создать новую», заполняет необходимые поля. Нажимает кнопку «Создать». Новая подборка создана, все поставленные метрики были достигнуты.

Задача 2. Респондентам необходимо найти фильм с помощью рекомендаций. Метрики: скорость, удовлетворенность.

Сценарий №1 для выполнения задания:

Пользователь авторизуется с помощью Google, после чего попадает на страницу «Подборки». Далее нажимает кнопку «Рекомендации». Пользователь пролистывает предложенные фильмы и просматривает их описание. Пользователь не находит интересующие фильмы и раскрывает список жанров, после чего выбирает необходимый жанр. Пользователь находит необходимый фильм. Все поставленные метрики достигнуты.

Задача 3. Респондентам необходимо добавить фильм в подборку. Метрики: успешность, удобство.

Сценарий №1 для выполнения задания:

Пользователь уже находится на странице «Подборки». Он выбирает необходимую подборку, которую создал заранее и переходит на ее страницу. Пользователь нажимает кнопку «Добавить фильм» и вводит данные в модальном окне. Выбирает

необходимый фильм и добавляет его в подборку, после чего переходит на страницу данного фильма. Все поставленные метрики достигнуты.

После проведения тестирования онлайн-сервиса «Moviq» можно утверждать, что не было обнаружено никаких явных ошибок и несоответствий поставленным задачам. Все функции, включая навигацию, обработку контента и взаимодействие с пользователем, работают корректно и без сбоев.

5.3 Особенности продвижения проекта

Особенности продвижения проекта «Moviq» заключаются в использовании различных методов SEO. Основная цель SEO – это повышение видимости сайта в результатах поисковых систем, что напрямую влияет на количество посетителей и их взаимодействие с контентом.

Важным этапом стала разработка семантического ядра, включающего ключевые слова и фразы, которые наиболее точно отражают содержание сайта и запросы потенциальных пользователей. Семантическое ядро стало основой для создания релевантного контента и улучшения структуры сайта.

Были грамотно настроены мета теги, включая теги заголовков `title` и описаний `description`, что позволило улучшить видимость сайта в результатах поиска и повысить кликабельность.

Заголовки страниц включали основные ключевые слова, что способствовало улучшению индексации сайта поисковыми системами и привлечению большего числа посетителей.

Пример добавления тегов `title` и `description` представлен на листинге 5.1.

```
methods: {
  async updateMetaTags() {
    document.title = 'Ваши подборки фильмов| Moviq';
    const description = 'На этой странице вы можете управлять своими подборками фильмов: добавлять новые и редактировать существующие.'

    const metaDescription = document.querySelector('meta[name="description"]');

    if (metaDescription) {
      metaDescription.setAttribute('content', description);
    } else {
      const newMetaDescription = document.createElement('meta');
      newMetaDescription.name = 'description';
      newMetaDescription.content = description;
      document.head.appendChild(newMetaDescription);
    }
  },
  async created() {
```

```
await this.updateMetaTags();
},
```

Листинг 5.1 – Пример добавления тегов title и description

Для улучшения пользовательского опыта и повышения ранжирования сайта также были настроены человеко-понятные URL (ЧПУ). Вместо длинных и неинформативных ссылок использовались короткие и понятные URL, которые отражают структуру сайта. Это не только помогает улучшить SEO, но и делает навигацию по более удобной для пользователей. Пример ЧПУ представлен на рисунке 5.16.

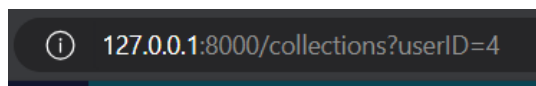


Рисунок 5.16 – ЧПУ

Особое внимание было уделено оптимизации изображений на сайте. Все изображения были снабжены атрибутами alt и title для обеспечения их корректного отображения в случае проблем с загрузкой и для улучшения индексации сайта поисковыми системами. Атрибуты alt и title также способствуют улучшению доступности сайта для пользователей с ограниченными возможностями. Пример добавления атрибутов alt и title для тега img представлен на листинге 5.2.

```

```

Листинг 5.2 – Пример добавления атрибутов alt и title

Благодаря этим мерам, сайт «Moviq» был оптимизирован для поисковых систем как Google, так и Яндекс.

5.4 Выводы по разделу

В рамках данного раздела было произведено тестирование сайта, включающее выполнение тестовых заданий для проверки корректности расположения элементов на привычном для пользователей месте и наличия пояснительных сообщений при выявлении ошибок. Было также проведено тестирование кроссбраузерности и графического интерфейса, и в результате не было обнаружено критических ошибок, функционал сайта работает без отказов.

Также были представлены рекомендации по продвижению сайта, которые включают в себя построение семантического ядра, оптимизацию мета-тегов, построение человеко-понятных URL, а также добавление атрибутов alt и title для изображений для онлайн-сервиса «Moviq».

6 Экономический раздел

6.1 Общая характеристика разрабатываемого программного средства

Процесс разработки и внедрения программного обеспечения охватывает множество различных аспектов, начиная от технических деталей и заканчивая экономическими и маркетинговыми вопросами. В этом контексте онлайн-сервис «Moviq» представляет собой не только технический продукт, но и результат стратегического планирования, направленного на привлечение новых пользователей и создание устойчивой аудитории.

«Moviq» был создан с целью увеличения числа пользователей и расширения аудитории. Основная идея онлайн-сервиса «Moviq» заключается в том, чтобы предоставить пользователям инструмент для быстрого и удобного поиска фильмов в соответствии с их предпочтениями. Сервис также позволяет создавать собственные подборки фильмов, что делает его привлекательным для тех, кто хочет составить персональные списки любимых фильмов.

Важным аспектом успешного запуска «Moviq» является эффективная маркетинговая стратегия. Помимо обеспечения технической готовности приложения, необходимо разработать план по продвижению сервиса на рынке. Это включает в себя определение целевой аудитории, выбор каналов коммуникации, создание привлекательного контента и установление партнерских отношений с другими компаниями и платформами.

Ключевым элементом бизнес-модели сервиса является бесплатный доступ к сервису, что привлекает большую аудиторию постоянных пользователей. В отличие от конкурентов, сервис предоставляет возможность создания своих собственных подборок фильмов, а также алгоритмы рекомендации фильмов, что делает процесс выбора фильмов более информативным. Высокое качество контента, интуитивно понятный интерфейс также являются значимыми преимуществами «Moviq» перед конкурентами.

Определение затрат на все стадии разработки и расчет экономической эффективности проекта позволяет лучше понять потенциал и перспективы сервиса на рынке. Предварительное исследование рынка и конкурентной среды позволяет определить преимущества «Moviq» перед аналогичными сервисами и разработать стратегию его позиционирования.

Маркетинговый анализ играет важную роль в разработке стратегии продвижения продукта на рынке. Он представляет собой систематическое исследование рыночной среды с целью понимания потребностей потенциальных клиентов, конкурентной обстановки и возможностей для успешного внедрения продукта. Этот процесс включает в себя оценку текущего состояния рынка, анализ трендов и прогнозирование его развития.

					БГТУ 06.00.ПЗ			
Изм.	Лист	№ документа	Подп.	Дата				
Разраб.		Помоз			Экономический раздел			
Пров.		Осоко						
Консульт.		Евлаш						
Н. контр.		Осоко						
Утв.		Романенко						
						Лит.	Лист	Листов
						y	1	11
						74319008, 2024		

Технологии, используемые при разработке веб-сайта: фреймворк для языка программирования PHP Laravel, фреймворк для языка программирования JavaScript Vue.js, препроцессор SASS, а так же графические редакторы Figma, Adobe Photoshop, CorelDraw, СУБД MySQL.

Стратегия монетизации сервиса предполагает продажу сервиса компаниям, которые заинтересованы в новом продукте, либо хотят масштабировать имеющиеся продукты. Предполагаемые каналы продвижения включают рекламные кампании в социальных сетях, партнерские программы с другими онлайн-сервисами и сарафанное радио.

6.2 Исходные данные для проведения расчётов и маркетинговый анализ

Источниками исходных данных для данных расчетов выступают действующие нормативные правовые акты.

Исходные данные для расчета приведены в таблице 6.1.

Таблица 6.1 – Исходные данные для расчета

Наименование показателя	Единица измерения	Условные обозначения	Норматив
Норматив дополнительной заработной платы	%	$H_{дз}$	15,00
Ставка отчислений в Фонд социальной защиты населения	%	$H_{фсзн}$	34,00
Ставка отчислений по обязательному страхованию в БРУСП «Белгосстрах»	%	$H_{бгс}$	0,60
Норматив прочих затрат	%	$H_{пз}$	20,00
Норматив накладных расходов	%	$H_{нр}$	50,00
Норматив расходов на реализацию	%	$H_{рр}$	15,00
Ставка НДС	%	$H_{ндс}$	20,00
Налог на прибыль	%	$H_{п}$	20,00

Далее приводится информация о стоимости разработки подобного продукта со ссылками на источник информации. Информация о стоимости подобных продуктов представлена в таблице 6.2 и указана в белорусских рублях.

Таблица 6.2 – Исходные данные анализа стоимости подобных продуктов

Продукт-аналог	Источник	Стоимость	Примечание
1	2	3	4
PairMovie	https://pairmovie.ru/	30 000	Уникальный функционал, позволяющий пользователям производить подбор фильмов вдвоем

Продолжение таблицы 6.2

1	2	3	4
Film-Like	https://www.film-like.com/	16 000	Платформа для выбора и просмотра фильмов с поддержкой системы лайков и рекомендаций
filmpro	https://filmpro.ru	32 000	Расширенный функционал для анализа фильмов, включая статистику просмотров, оценок и рецензий
Кинопоиск	https://www.kinopoisk.ru	40 000	Крупнейшая в России платформа для поиска информации о фильмах, сериалах и актерах. Обширная база данных, содержащая сведения о тысячах фильмов, сериалов, актеров и режиссеров. Функционал отзывов и рецензий, а также возможность составления персональных списков и оценивания фильмов

Таким образом, в ходе проведения маркетингового анализа, была определена стоимость разработки аналогичного программного продукта. Средняя цена разработки аналогичного продукта составляет 20 000-30 000 рублей.

Также стоит учитывать административную панель, стоимость разработки которой составляет 3 500 рублей, разработку механизма рекомендаций, стоимость которого составляет 2 500 рублей и механизм создания подборок, стоимость которого равна 2 000 рублей.

Таким образом, общая стоимость разработки данного программного средства, выбранного в качестве базы сравнения составит 28 000 руб.

6.3 Обоснование цены программного средства

В данном разделе будет рассмотрен процесс расчета цены программного продукта с учетом всех факторов, влияющих на его себестоимость и потенциальную прибыльность.

6.3.1 Затраты рабочего времени на разработку программного средства

Для определения общей стоимости разработки онлайн-сервиса «Moviq» были учтены все работы, выполненные на различных этапах создания продукта. Затраты рабочего времени рассчитываются в часах и отражают количество времени, затраченное на выполнение различных задач, связанных с разработкой, тестированием и внедрением программного средства.

Затраты рабочего времени на разработку программного средства представлены в таблице 6.3.

Таблица 6.3 – Затраты рабочего времени на разработку онлайн-сервиса

Содержание работ	Специалист	Затраты рабочего времени, часов
1. Постановка целей и задач проекта	Бизнес-аналитик	16
2. Создание и проработка технического задания на разработку сайта	Бизнес-аналитик	16
3. Анализ конкурентов	Менеджер проекта	24
4. Создание прототипов	Дизайнер	48
5. Создание макетов дизайна сайта	Дизайнер	96
6. Верстка макетов дизайна сайта	Front-end разработчик	168
7. Проектирование базы данных	Back-end разработчик	24
8. Разработка функциональной части	Back-end разработчик	248
9. Тестирование	Тестировщик	32
10. Оптимизация сайта для поисковых систем	SEO-специалист	32
Всего		704

Таким образом, исходя из таблицы 6.3, бизнес-аналитик затратил на проект 32 часа, менеджер проекта – 24 часа, дизайнер затратил 144 часа, front-end разработчик – 168 часов, back-end разработчик – 272 часа, тестировщик затратил 32 часа и SEO-специалист – 32 часа.

6.3.2 Расчет основной заработной платы

Для определения основной заработной платы были изучены данные о зарплатах специалистов на сайте rabota.by [19]. Результаты исследования представлены в таблице 6.4.

Таблица 6.4 – Средняя месячная заработная плата специалистов

Специалист	Средняя месячная заработная плата, руб	Часовая ставка, руб/час
1	2	3
Бизнес-аналитик	1 600	9,52
Менеджер проекта	1 600	9,52
Дизайнер	1 900	11,30
Front-end разработчик	2 000	11,90
Back-end разработчик	2 000	11,90

Продолжение таблицы 6.4

1	2	3
Тестировщик	1 400	8,33
SEO-специалист	1 600	9,52

Основная заработная плата, $C_{оз}$, руб., отдельного специалиста будет рассчитываться по формуле:

$$C_{оз} = T_{раз} \cdot C_{зн}, \quad (6.1)$$

где $T_{раз}$ – трудоемкость (чел./час);

$C_{зн}$ – средняя часовая ставка руб./час.

Были взяты следующие значения, а именно $T_{раз} = 32$ (таблица 6.3); $C_{зн} = 9,52$ (таблица 6.4).

$$C_{оз} = 32 \cdot 9,52 = 304,64 \text{ руб.}$$

Для прочих специалистов основная заработная плата вычислялась аналогичным способом.

Полученные значения приведены в таблице 6.5.

Таблица 6.5 – Средняя месячная заработная плата специалистов

Специалист	Затраты рабочего времени, часов	Часовая ставка, руб/час	Основная заработная плата, руб
Бизнес-аналитик	32	9,52	304,64
Менеджер проекта	24	9,52	228,48
Дизайнер	144	11,30	1 627,20
Front-end разработчик	168	11,90	1 999,20
Back-end разработчик	272	11,90	3 236,80
Тестировщик	32	8,33	266,56
SEO-специалист	32	9,52	304,64
Итого	704	71,99	7 967,52

Таким образом, на основании основной заработной платы отдельных специалистов, приведенной в таблице 6.5, основная заработная плата составляет 7 967,52 белорусских рублей.

6.3.3 Расчет дополнительной заработной платы

Дополнительная заработная плата, $C_{дз}$, руб., определяется по нормативу в процентах к основной заработной плате по формуле:

$$C_{дз} = \frac{C_{оз} \cdot H_{дз}}{100}, \quad (6.2)$$

где $H_{дз}$ – норматив дополнительной заработной платы, %.

Были взяты следующие значения: $C_{оз} = 7\,967,52$ рублей (таблица 6.5), $H_{дз} = 15\%$ (таблица 6.1).

Дополнительная заработная плата составит:

$$C_{дз} = 7\,967,52 \cdot 15 / 100 = 1\,195,13 \text{ руб.}$$

Дополнительная заработная плата составила 1 195,13 рублей.

6.3.4 Расчет отчислений в Фонд социальной защиты населения и по обязательному страхованию

Отчисления в Фонд социальной защиты населения (ФСЗН) и по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний в БРУСП «Белгосстрах» определяются в соответствии с действующими законодательными актами по нормативу в процентном отношении к фонду основной и дополнительной зарплаты исполнителей.

Отчисления в Фонд социальной защиты населения вычисляются по формуле:

$$C_{фсзн} = \frac{(C_{оз} + C_{дз}) \cdot H_{фсзн}}{100}, \quad (6.3)$$

где $H_{фсзн}$ – норматив отчислений в Фонд социальной защиты населения, %.

Были взяты следующие значения: $C_{оз} = 7\,967,52$ рублей (формула (6.1)), $C_{дз} = 1\,195,13$ рублей (формула (6.2)), $H_{фсзн} = 34\%$ (таблица 6.1).

Таким образом, по формуле (6.3), отчисления в Фонд социальной защиты населения будут равны:

$$C_{фсзн} = (7\,967,52 + 1\,195,13) \cdot 34 / 100 = 3\,115,3 \text{ руб.}$$

Отчисления в БРУСП «Белгосстрах» вычисляются по формуле 6.4.

$$C_{бгс} = \frac{(C_{оз} + C_{дз}) \cdot H_{бгс}}{100}, \quad (6.4)$$

где $H_{бгс}$ – ставка отчислений по обязательному страхованию в БРУСП «Белгосстрах», %.

Значение $H_{бгс} = 0,6$ по таблице (6.1).

Следовательно, отчисления будут равны:

$$C_{бгс} = (7\,967,52 + 1\,195,13) \cdot 0,6 / 100 = 55 \text{ руб.}$$

Таким образом, общие отчисления в БРУСП «Белгосстрах» составили 55 рублей, а в фонд социальной защиты населения – 3 115,3 руб.

6.3.5 Расчет суммы прочих прямых затрат

Сумма прочих затрат $C_{пз}$ определяется как произведение основной заработной платы исполнителей на конкретное программное средство $C_{оз}$ на норматив прочих затрат в целом по организации $H_{пз}$ и находится по формуле:

$$C_{пз} = \frac{C_{оз} \cdot H_{пз}}{100}, \quad (6.5)$$

Прочие прямые затраты для данной разработки рассчитываются по формуле (6.5), где $C_{оз} = 7\,967,52$ рублей (формула (6.1)), $H_{пз} = 20\%$ (таблица 6.1) по исходным данным.

$$C_{пз} = 7\,967,52 \cdot 20 / 100 = 1\,593,5 \text{ руб.}$$

Прочие затраты при разработке проекта необходимы для покрытия различных дополнительных расходов, которые не включены в основные категории, такие как заработная плата или накладные расходы. Эти затраты играют важную роль в обеспечении успешного завершения проекта, поскольку они охватывают широкий спектр потребностей, возникающих в ходе работы. В процессе разработки требуется специализированное оборудование и программное обеспечение, включая лицензии на программные продукты, аренду серверов, облачные сервисы и другие технические ресурсы.

6.3.6 Расчет суммы накладных расходов

Сумма накладных расходов $C_{н.р.}$ – произведение основной заработной платы исполнителей на конкретное программное средство $C_{оз}$ на норматив накладных расходов в целом по организации $H_{н.р.}$, по формуле:

$$C_{н.р.} = \frac{C_{оз} \cdot H_{н.р.}}{100}, \quad (6.6)$$

Сумма накладных расходов составит:

$$C_{н.р.} = 7\,967,52 \cdot 50 / 100 = 3\,983,76 \text{ руб.}$$

Таким образом, сумма общепроизводственных, общехозяйственных расходов составила 3 983,76 рублей.

6.3.7 Сумма расходов на разработку программного средства

Сумма расходов на разработку программного средства, C_p , руб., определяется как сумма основной и дополнительной заработной платы исполнителей на конкретное программное средство, отчислений на социальные нужды, суммы прочих затрат и суммы накладных расходов, по формуле:

$$C_p = C_{оз} + C_{дз} + C_{фсзн} + C_{бгс} + C_{пз} + C_{н.р.}, \quad (6.7)$$

По формуле (6.7) можно определить сумму расходов на разработку программного средства:

$$C_p = 7\,967,52 + 1\,195,13 + 3\,115,3 + 55 + 1\,593,5 + 3\,983,76 = 17\,910,21 \text{ руб.}$$

Таким образом, сумма расходов на разработку программного средства была вычислена на основе данных, рассчитанных в данном разделе, и составила 17 910,21 рублей.

6.3.8 Расходы на реализацию

Сумма расходов на реализацию программного средства, C_{pp} , руб., определяется как произведение суммы расходов на разработку на норматив расходов на реализацию H_{pp} , и находится по формуле:

$$C_{pp} = \frac{C_p \cdot H_{pp}}{100}, \quad (6.8)$$

где H_{pp} – норматив расходов на реализацию, %.

Норматив расходов на реализацию $H_{pp} = 15\%$ по таблице 6.1, тогда, по формуле (6.8), сумма расходов на реализацию составляет:

$$C_{pp} = 17\,910,21 \cdot 15 / 100 = 2\,686,53 \text{ руб.}$$

Все проведенные выше расчеты необходимы для вычисления полной себестоимости проекта.

6.3.9 Расчет полной себестоимости

Полная себестоимость, C_{π} , руб., определяется как сумма двух элементов: суммы расходов на разработку, C_p , руб., и суммы расходов на реализацию программного средства, C_{pp} , руб.

Полная себестоимость C_{π} определяется по формуле:

$$C_{\pi} = C_p + C_{pp}, \quad (6.9)$$

$$C_{\pi} = 17\,910,21 + 2\,686,53 = 20\,596,74 \text{ руб.}$$

Полная себестоимость программного средства была вычислена на основе данных, рассчитанных ранее в данном разделе, и составила $C_{\pi} = 20\,596,74$ рублей.

6.3.10 Определение цены, оценка эффективности

Данное программное средство было разработано с целью его дальнейшей продажи. Ожидается, что после разработки веб-сайта число посетителей будет стабильно расти каждый месяц.

В ходе анализа было установлено, что полная стоимость разработки веб-приложения составляет 28 000 рублей.

$$C_{\text{сНДС}} = 28\,000 \text{ руб.}$$

Сумма налога на добавленную стоимость, НДС, руб., определяется как произведение цены разработчика, $C_{\text{сНДС}}$, руб., на ставку НДС, $H_{\text{НДС}}$, %, и находится по формуле:

$$\text{НДС} = C_{\text{сНДС}} \cdot H_{\text{НДС}} / (100 + H_{\text{НДС}}), \quad (6.10)$$

Ставка НДС $H_{\text{НДС}} = 20\%$ известна из таблицы 6.1. Цена разработчика $C_{\text{сНДС}}$ составляет 28 000 рублей. Подставив значения в формулу, получим:

$$\text{НДС} = 28\,000 \cdot 20 / 120 = 4\,666 \text{ руб.}$$

Цена разработки программного средства, C_p , руб., без налогов определяется как разность двух элементов: отпускной цены $C_{\text{сндс}}$ минус сумма налога на добавленную стоимость НДС и вычисляется по формуле:

$$C_p = C_{\text{сндс}} - \text{НДС}, \quad (6.11)$$

Подставив уже известные значения в формулу (6.11), получим:

$$C_p = 28\,000 - 4\,666 = 23\,334 \text{ руб.}$$

Прибыль от реализации программного средства, $P_{\text{пс}}$, руб., определяется как разность цены разработки программного средства без налога C_p минус полную себестоимость продукта, $C_{\text{п}}$, руб., и находится по формуле:

$$P_{\text{пс}} = C_p - C_{\text{п}}, \quad (6.12)$$

Полная себестоимость продукта $C_{\text{п}}$ была определена ранее по формуле (6.9) и составляет 20 606,74 рублей. Цена разработки определена по формуле (6.11). Подставив значения в формулу, получим:

$$P_{\text{пс}} = 23\,334 - 20\,596,74 = 2\,737,25 \text{ руб.}$$

Уровень рентабельности $Y_{\text{рент}}$ определяется как отношение прибыли от реализации программного средства $P_{\text{пс}}$ и полной себестоимости разработанного продукта $C_{\text{п}}$ и рассчитывается по формуле:

$$Y_{\text{рент}} = P_{\text{пс}} / C_{\text{п}} \cdot 100\%, \quad (6.13)$$

Полная себестоимость продукта $C_{\text{п}}$ была определена ранее по формуле 6.9 и составляет 20596,74 рублей. Прибыль от реализации $P_{\text{пс}} = 2737,25$ рублей. Подставив необходимые значения в формулу, получим:

$$Y_{\text{рент}} = (2737,25 / 20596,74) \cdot 100\% = 13,29\%.$$

В результате были определены цена разработки программного средства без НДС – 23334 руб. и уровень рентабельности сайта – 13,29%.

Налог на прибыль рассчитывается по формуле:

$$\text{НП} = P_{\text{пс}} \cdot (H_{\text{п}} / 100), \quad (6.14)$$

Ставка налога на прибыль ($H_{\text{п}}$) равна 20% по таблице 6.1. Прибыль от реализации $P_{\text{пс}} = 2\,737,25$ рублей.

Подставив необходимые значения в формулу (6.14), получим:

$$\text{НП} = 2\,737,25 \cdot (20 / 100) = 547,45 \text{ руб.}$$

Проведенные расчеты необходимы для вычисления чистой прибыли проекта. Чистая прибыль рассчитывается по формуле:

$$P_{\text{ч}} = P_{\text{пс}} - \text{НП}. \quad (6.15)$$

Прибыль от реализации $P_{\text{пс}} = 2\,737,25$ рублей по формуле (6.12). Налог на прибыль (НП) составляет 547,45 рублей по формуле (6.14). Подставив необходимые значения в формулу, получим:

$$\Pi_4 = 27\,37,25 - 547,45 = 2\,189,8 \text{ руб.}$$

Таким образом, чистая прибыль от реализации программного средства составляет 2 189,8 рублей.

6.4 Расчет стоимости хостинга и доменного имени

Для публикации веб-приложения, помимо кода и базы данных, необходимо иметь доменное имя и хост провайдера. Доменное имя дает сайту адрес в сети. В зависимости от уровня домена и его локации стоимость может изменяться от 15 до 33 рублей в год. Для разработанного веб-сайта был выбран домен «.by». Стоимость домена «.by» составила 33 рубля в год [20].

Хостинг провайдер предоставляет место для размещения данных в сети Интернет и определенное количество памяти. Большинство провайдеров предоставляют разные объемы памяти, оперативной памяти, способы шифрования и прочее под различные размеры сайтов. Стоимость будет изменяться в зависимости от выбранного тарифа. Для разработанного веб-сайта был подобран хостинг со стоимостью тарифного плана 140 рублей в год [21]. Таким образом можно выяснить, что кроме покупки сайта, заказчик будет тратить на содержание сайта 173 рубля в год.

6.5 Выводы по разделу

В таблице 6.6 представлены результаты расчетов для основных показателей:

Таблица 6.6 – Результаты расчетов

Наименование показателя	Значение
Время разработки, ч.	704,00
Основная заработная плата, руб.	7967,52
Дополнительная заработная плата, руб.	1195,13
Отчисления в Фонд социальной защиты населения, руб.	3115,30
Отчисления в БРУСП «Белгосстрах», руб.	55,00
Прочие прямые затраты, руб.	1593,50
Накладные расходы, руб.	3983,76
Себестоимость разработки программного средства, руб.	17910,21
Расходы на реализацию, руб.	2686,53
Полная себестоимость, руб.	20596,74
Цена реализации, сформированная на основе проведенного маркетингового анализа рынка, руб.	28000,00
Прибыль от реализации, руб.	2737,25
Рентабельность разработки программного средства, %.	13,29
Чистая прибыль, руб.	2189,80

Проведенные расчеты показали, что реализация веб-сайта «Moviq» является экономически целесообразной и выгодной. Полная себестоимость разработки программного средства составляет 20 596,74 рубля, в то время как цена реализации

на основе маркетингового анализа установлена на уровне 28 000 рублей. Это позволяет получить прибыль от реализации в размере 2 737,25 рубля и чистую прибыль в размере 2 189,80 рубля.

Рентабельность разработки программного средства составила 13,29%, что является хорошим показателем и демонстрирует, что проект обладает устойчивостью и потенциалом для успешного выхода на рынок.

Проведенный анализ рынка и конкурентной среды позволяет утверждать, что «Moviq» имеет ряд преимуществ перед аналогичными сервисами, что дополнительно способствует его успешному внедрению и продвижению. Ключевыми преимуществами «Moviq» являются индивидуальные рекомендации, возможность создания пользовательских подборок и интеграция интерактивного контента.

Основная планируемая модель монетизации «Moviq» – это одноразовая продажа. Во-первых, одноразовая продажа программного средства позволяет получить значительную прибыль за короткий период. Во-вторых, наличие готового продукта на рынке увеличивает шансы на привлечение клиентов и установление партнерских отношений. В-третьих, успешная положительная отзывы пользователей способствуют укреплению репутации компании-разработчика на рынке.

Заключение

В процессе разработки онлайн-сервиса «Moviq» были проанализированы аналогичные и конкурентные платформы в данной области, выявлены их достоинства и недостатки. Определены основные цели и задачи проекта, а также выбрана целевая аудитория сервиса. На основе полученных данных была разработана структура сайта, созданы прототипы страниц и определены стили дизайна.

Для сервиса был тщательно подобран и обработан контент, который будет использоваться на его страницах. Все страницы кроссбраузерны, адаптивны и были созданы с использованием фреймворка Laravel. Стили были описаны с помощью динамического языка стилей SASS, который затем компилировался в CSS. Интерактивные элементы сайта были реализованы с использованием фреймворка языка программирования JavaScript Vue.js. База данных для проекта создавалась в СУБД MySQL.

Готовый сервис был протестирован в браузерах Microsoft Edge, Google Chrome и Яндекс. По результатам тестирования юзабилити, сервис получил позитивные оценки пользователей, которые были удовлетворены его работой.

После реализации проекта пользователи смогут легко находить подходящие фильмы благодаря продуманной системе поиска и фильтрации по различным параметрам. Рекомендательные алгоритмы помогут пользователям открывать для себя новые фильмы, соответствующие их вкусам. Регистрация и авторизация позволят пользователям пользоваться сервисом, осуществлять поиск требуемых фильмов, а также создавать подборки фильмов. Форма связи с модератором поможет улучшить взаимодействие между пользователями и администрацией сервиса.

Адаптивность позволит пользователям пользоваться сервисом на различных устройствах, включая мобильные. Для администраторов будет доступна панель управления, где они смогут добавлять и редактировать контент.

В целом, данный функционал сделает сервис более удобным и функциональным для пользователей и администраторов.

Таким образом, цели и задачи, поставленные в проекте, полностью выполнены. Результатом работы является полноценный и законченный онлайн-сервис для подбора фильмов «Moviq».

					БГТУ 00.00.ПЗ			
Изм	Лист	№ документа	Подп.	Дата	Заключение	Лит.	Лист	Листов
Разраб.	Помоз					у	1	1
Пров.	Осоко					74319008, 2024		
Консульт.								
Н. контр.	Осоко							
Утв.	Романенко							

Список использованных источников

1. Что такое веб-дизайн? [Электронный ресурс] / Сайт «Евробайт» – 2021. – Режим доступа: <https://eurobyte.ru/articles/chto-takoe-veb-dizajn>. – Дата доступа: 25.02.2024.
2. Большинство времени проводим в Интернете [Электронный ресурс] / Логистика – 2023. – Режим доступа: <https://logistics.by/blog/bolshinstvo-vremeni-provodim-ozhidaya-sobytij-kotorye-mogut-nikogda-ne-proizojti>. – Дата доступа: 25.02.2024.
3. Главная страница сайта PairMovie [Электронный ресурс] / PairMovie. – 2014–2023. – Режим доступа: <https://pairmovie.ru/>. – Дата доступа: 27.02.2024.
4. Главная страница сайта FilmLike [Электронный ресурс] / FilmLike. – 2022. – Режим доступа: <https://film-like.com/>. – Дата доступа: 27.02.2024.
5. Главная страница сайта filmpro [Электронный ресурс] / Сайт «filmpro». – 2023. – Режим доступа: <https://www.filmpro.ru/>. – Дата доступа: 27.02.2024.
6. Главная страница сайта Кинопоиск [Электронный ресурс] / Сайт «Кинопоиск». – 2023. – Режим доступа: <https://www.kinopoisk.ru/> – Дата доступа: 27.02.2024.
7. Пользовательские сценарии в UX-дизайне [Электронный ресурс] / Experrto. 2020. – Режим доступа: <https://ru.experrto.io/blog/2019/06/19/polzovatelskie-scenarii-v-ux-dizajne/> – Дата доступа: 15.04.2024.
8. Основы применения UML. Кто и как его использует [Электронный ресурс] / Systems Education. 2023. – Режим доступа: <https://systems.education/who-uses-uml/> – Дата доступа: 16.04.2024.
9. Минимализм в веб-дизайне: особенности стиля [Электронный ресурс] / Сайт «idbi». – 2024. – Режим доступа: <https://idbi.ru/blogs/blog/minimalizm-v-veb-dizayne> – Дата доступа: 19.04.2024.
10. Основы типографики в дизайне [Электронный ресурс] / Bang Bang Education. – 2024. – Режим доступа: <https://bangbangeducation.ru/point/grafichieskii-dizain/chto-takoe-tipografika> – Дата доступа: 19.04.2024.
11. Что такое логотип и зачем он нужен [Электронный ресурс] / Turbologo. – 2024. – Режим доступа: <https://turbologo.ru/blog/logotype> – Дата доступа: 22.04.2024.
12. Разработка дизайн-макета и создание структуры сайта [Электронный ресурс] / OKsoft. – 2023. – Режим доступа: https://oksoft.ru/create_site_design – Дата доступа: 25.04.2024.
13. Стоковые фотографии, видео, векторные изображения [Электронный ресурс] / Сайт «freepik». – 2024. – Режим доступа: <https://www.freepik.com>. Дата доступа: 02.05.2024.
14. Sass [Электронный ресурс] / Skillfactory. – 2023. – Режим доступа: <https://blog.skillfactory.ru/glossary/sass/> – Дата доступа: 05.05.2024.
15. Как работает CSS Flexbox [Электронный ресурс] / Tproger. – 2020. – Режим

					БГТУ 00.00.ПЗ			
Изм.	Лист	№ документа	Подп.	Дата				
Разраб.	Помоз				Список использованных источников	Лит.	Лист	Листов
Пров.	Осоко					у	1	2
Консульт.						74319008, 2024		
Н. контр.	Осоко							
Утв.	Романенко							

доступа: <https://tproger.ru/translations/how-css-flexbox-works> – Дата доступа: 05.05.2024.

16. CSS медиа-запросы [Электронный ресурс] / Сайт «Итшеф». – 2023. – Режим доступа: <https://itchief.ru/html-and-css/media-queries> – Дата доступа: 05.05.2024.

17. Фильмы, сериалы и т.д. [Электронный ресурс] / Neiros. – 2024. – Режим доступа: <https://itchief.ru/html-and-css/media-queries> – Дата доступа: 12.02.2024.

18. Тестирование и 7 основных этапов его проведения [Электронный ресурс] / neiros. – 2023. – Режим доступа: <https://neiros.ru/blog/code/testirovanie-i-7-osnovnykh-etapov-ego-provedeniya> – Дата доступа: 20.05.2024.

19. IT-вакансии в Беларуси [Электронный ресурс] / Сайт rabota.by. – 2024. – Режим доступа: <https://rabota.by/> – Дата доступа: 20.05.2024.

20. Регистрация доменов [Электронный ресурс] / Сайт [домен.by](https://domain.by/). – 2024. – Режим доступа: <https://domain.by/domain-register/> – Дата доступа: 20.05.2024.

21. Хостинг для размещения сайтов с поддержкой PHP [Электронный ресурс] / Сайт [домен.by](https://domain.by/). – 2024. – Режим доступа: <https://domain.by/hosting-order/> – Дата доступа: 20.05.2024.

Перечень графического иллюстративного материала

- БГТУ 01.00 С1 – Логическая схема базы данных;
- БГТУ 02.00 С1 – Структурная схема онлайн-сервиса;
- БГТУ 03.00 РР – Диаграмма вариантов использования;
- БГТУ 04.00 РР – Прототипы страниц сервиса;
- БГТУ 05.00 РР – Дизайн-макеты основных страниц;
- БГТУ 06.00 РР – Элементы дизайна.

Полная структура разработанного проекта

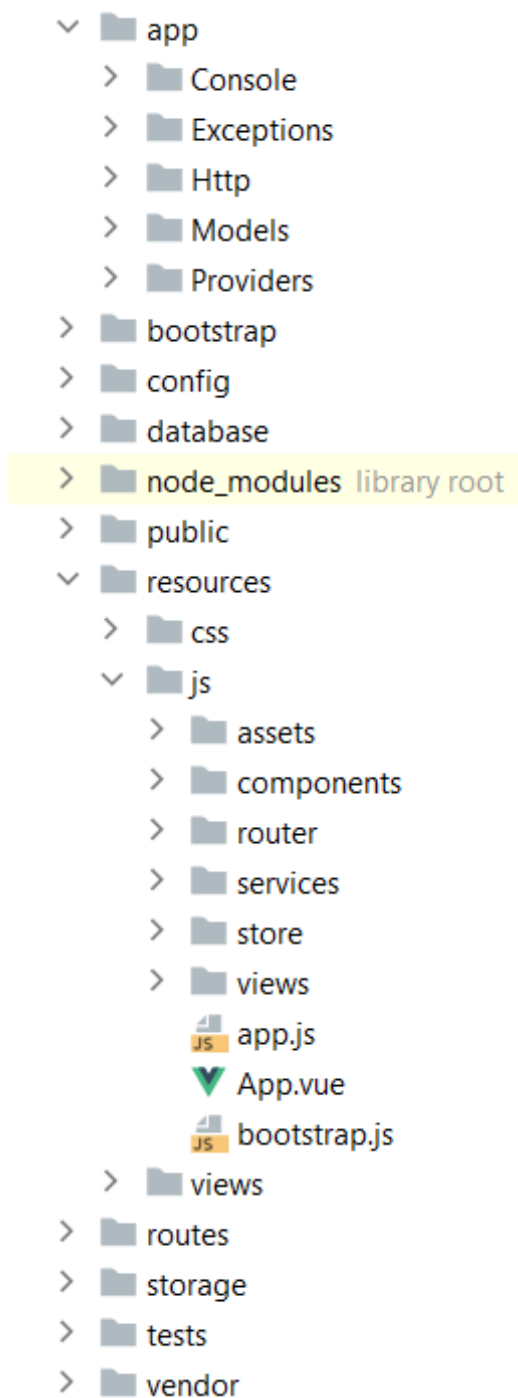


Рисунок А.1 – Структура разработанного проекта

Основной программный код сервиса

```
public static function callbackGoogle(Request $request) {
    try {
        $code = $request['code'];

        // Получаем ответ с токеном доступа
        $tokenResponse = Socialite::driver('google')->getAccessTokenResponse($code);

        // Передаем токен в метод для получения объекта пользователя
        $user = Socialite::driver('google')->userFromToken($tokenResponse['access_token']);

        $existingUser = User::where('email', $user->getEmail())->first();

        $res = [];

        if ($existingUser) {
            if ($existingUser->password == '') {
                $res[] = [
                    "status" => true,
                    "type" => 0,
                    "message" => "Вход успешен",
                    "user" => $existingUser,
                ];
            } else {
                $res[] = [
                    "status" => false,
                    "type" => 1,
                    "message" => "Войдите стандартным способом",
                ];
            }
        } else {
            // Создаем нового пользователя
            $newUser = User::create([
                'nickname' => $user->getName(),
                'email' => $user->getEmail(),
                'password' => '',
                'role_id' => 2,
            ]);

            $res[] = [
                "status" => true,
                "type" => 0,
                "message" => "Вход успешен!",
                "user"=>$newUser['data']
            ];
        }
    }
}
```



```

        return $res;
    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}

```

Листинг Б.1 – Метод callbackGoogle

```

public static function editProfile(Request $request) {
    $id = $request['userID'];
    $name = $request['name'];
    $email = $request['email'];
    $old_email = $request['old_email'];
    $password = $request['password'];

    $existingUser = User::where('email', $old_email)
        ->where('id', '!=', $id)
        ->exists();

    if ($existingUser) {
        return response()->json([
            "status" => false,
            "type" => 1,
            "message" => "Пользователь с таким Email уже зарегистри-
рован"
        ]);
    }

    // Обновление данных пользователя
    $user = User::find($id);

    if ($user) {
        if (!empty($name)) {
            $user->nickname = $name;
        }

        if (!empty($email)) {
            $user->email = $email;
        }

        if (!empty($password)) {
            $hashedPassword = Hash::make($password);
            $user->password = $hashedPassword;
        }

        $user->save();

        return response()->json([
            "status" => true,
            "type" => 0,

```

```

        "message" => "Данные успешно изменились",
    });
} else {
    return response()->json([
        "status" => false,
        "type" => 1,
        "message" => "Произошла ошибка, данные не были изме-
нены",
    ]);
}
}

```

Листинг Б.2 – Код для редактирования данных пользователя

```

public static function getMoviesTypes() {
    $types = Type::select('name')->get();

    if ($types->isNotEmpty()) {
        $res = [
            "status" => true,
            "type" => 0,
            "message" => "200 ok",
            "items" => $types,
        ];
    } else {
        $res = [
            "status" => false,
            "type" => 1,
            "message" => "Произошла ошибка"
        ];
    }

    return $res;
}

public static function getMoviesRestricts() {
    $restricts = Restrict::select('restriction')->get();

    if ($restricts->isNotEmpty()) {
        $res = [
            "status" => true,
            "type" => 0,
            "message" => "200 ok",
            "items" => $restricts,
        ];
    } else {
        $res = [
            "status" => false,
            "type" => 1,
            "message" => "Произошла ошибка"
        ];
    }
}

```

```

        return $res;
    }

    public static function getMoviesGenres() {
        $genres = Genre::select('name')->get();

        if ($genres->isNotEmpty()) {
            $res = [
                "status" => true,
                "type" => 0,
                "message" => "200 ok",
                "items" => $genres,
            ];
        } else {
            $res = [
                "status" => false,
                "type" => 1,
                "message" => "Произошла ошибка"
            ];
        }

        return $res;
    }

    public static function addMovie(Request $request) {
        try {
            $countryMovie = $request['country'];
            $kpID = $request['kpID'];
            $nameMovie = $request['nameMovie'];
            $alternativeName = $request['alternativeName'];
            $description = $request['description'];
            $shortDescription = $request['shortDescription'];
            $slogan = $request['slogan'];
            $year = $request['year'];
            $isSeries = $request['isSeries'];
            $kpRating = $request['kpRating'];
            $ratingMpaa = $request['ratingMpaa'];
            $votesKp = $request['votesKp'];
            $budget = $request['budget'];
            $movieLength = $request['movieLength'];
            $seriesLength = $request['seriesLength'];
            $backdrop = $request['backdrop'];
            $poster = $request['poster'];
            $yearRus = $request['yearRus'];
            $yearWorld = $request['yearWorld'];
            $yearSerialStart = $request['yearSerialStart'];
            $yearSerialEnd = $request['yearSerialEnd'];
            $trailer = $request['trailer'];
            $movieSelectedTypeOption = $request['movieSelectedTypeOption'];
            $movieSelectedRestrictOption = $request['movieSelectedRestrictOption'];
            $movieSelectedGenreOption = $request['movieSelectedGenreOption'];

```

```

        $type = Type::where('name', $movieSelectedTypeOption)-
>first();

        if($type) {
            $typeID = $type->id;
        } else {
            throw new \InvalidArgumentException("Такого типа не су-
ществует");
        }

        $restrict = Restrict::where('restriction', $movieSelecte-
dRestrictOption)->first();

        if($restrict) {
            $restrictID = $restrict->id;
        } else {
            throw new \InvalidArgumentException("Такого ограничения
не существует");
        }

        $genre = Genre::where('name', $movieSelectedGenreOption)-
>first();

        if($genre) {
            $genreID = $genre->id;
        } else {
            throw new \InvalidArgumentException("Такого жанра не су-
ществует");
        }

        $countryID = null;

        $addCountry = Countrie::where('country', $countryMovie)-
>first();

        if (!$addCountry) {
            $country = Countrie::create([
                'country' => $countryMovie,
            ]);

            if($country) {
                $countryID = $country->id;
            } else {
                throw new \InvalidArgumentException("Страна не до-
бавлена");
            }
        } else {
            $countryID = $addCountry->id;
        }

        $movie = Movie::create([
            'externalID' => $kpID,

```

```

        'name' => $nameMovie,
        'alternativeName' => $alternativeName,
        'description' => $description,
        'shortDescription' => $shortDescription,
        'slogan' => $slogan,
        'year' => $year,
        'isSeries' => $isSeries,
        'ratingKp' => $kpRating,
        'ratingMpaa' => $ratingMpaa,
        'votesKp' => $votesKp,
        'budget' => $budget,
        'movieLength' => $movieLength,
        'seriesLength' => $seriesLength,

        'type_id' => $typeID,
        'restrict_id' => $restrictID,
    ]);

    if (!$movie) {
        throw new \Exception("Запись не была создана");
    }

    $movie->genres()->attach($genreID);
    $movie->countries()->attach($countryID);

    if (isset($backdrop)) {
        $addBackdrop = Backdrop::create([
            'movie_id' => $movie->id,
            'url' => $backdrop,
        ]);
        if (!$addBackdrop) {
            throw new \Exception("Данные по постеру не добав-
лены");
        }
    }

    if (isset($poster)) {
        $addPoster = Poster::create([
            'movie_id' => $movie->id,
            'url' => $poster,
        ]);
        if (!$addPoster) {
            throw new \Exception("Данные по постеру не добав-
лены");
        }
    }

    $premiere = Premiere::create([
        'movie_id' => $movie->id,
        'russia' => $yearRus ?? null,
        'world' => $yearWorld ?? null,
    ]);
    if (!$premiere) {
        throw new \Exception("Данные по премьере не добавлены");
    }

```

```

    }

    $releaseYears = ReleaseYear::create([
        'movie_id' => $movie->id,
        'start' => $yearSerialStart ?? null,
        'end' => $yearSerialEnd ?? null,
    ]);
    if (!$releaseYears) {
        throw new \Exception("Данные по дате релиза не добав-
лены");
    }

    if (isset($trailer)) {
        $addTrailer = Trailer::create([
            'movie_id' => $movie->id,
            'url' => $trailer,
        ]);
        if (!$addTrailer) {
            throw new \Exception("Данные по трейлеру не добав-
лены");
        }
    }

    return [
        "status" => true,
        "type" => 0,
        "message" => 'Добавлен новый фильм',
    ];

} catch (\Exception $e) {
    return [
        "status" => false,
        "type" => 1,
        "message" => $e->getMessage(),
    ];
}
}

public static function addCountry(Request $request) {
    try {
        $newCountry = $request['newCountry'];
        $addCountry = Countrie::where('country', $newCountry)-
>first();
        if (!$addCountry) {
            $country = Countrie::create([
                'country' => $newCountry,
            ]);
            if (!$country) {
                throw new \InvalidArgumentException("Произошла
ошибка при добавлении страны");
            }
        } else {
            throw new \InvalidArgumentException("Страна уже суще-
ствует");
        }
    }
}

```

```

    }
    return [
        "status" => true,
        "type" => 0,
        "message" => 'Добавлена новая страна',
    ];
} catch (\Exception $e) {
    return [
        "status" => false,
        "type" => 1,
        "message" => $e->getMessage(),
    ];
}
}

public static function addGenre(Request $request) {
    try {
        $newGenre = $request['newGenre'];

        $addGenre = Genre::where('name', $newGenre)->first();

        if (!$addGenre) {
            $genre = Genre::create([
                'name' => $newGenre,
            ]);

            if (!$genre) {
                throw new \InvalidArgumentException("Произошла
ошибка при добавлении жанра");
            }
        } else {
            throw new \InvalidArgumentException("Жанр уже суще-
ствует");
        }

        return [
            "status" => true,
            "type" => 0,
            "message" => 'Добавлен новый жанр',
        ];

    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}

public static function addRestrict(Request $request) {
    try {
        $newRestrict = $request['newRestrict'];
    }
}

```

```

        $addRestrict = Restrict::where('restriction', $newRestrict)-
>first();

        if (!$addRestrict) {
            $restrict = Restrict::create([
                'restriction' => $newRestrict,
            ]);

            if (!$restrict) {
                throw new \InvalidArgumentException("Произошла
ошибка при добавлении ограничения");
            }
        } else {
            throw new \InvalidArgumentException("Ограничение уже су-
ществует");
        }

        return [
            "status" => true,
            "type" => 0,
            "message" => 'Добавлено новое ограничение',
        ];

    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}

public static function getSearchMovie(Request $request)
{
    try {
        $kpID = $request['kpID'];
        $nameMovie = $request['nameMovie'];
        $alternativeName = $request['alternativeName'];
        $type = $request['type'];
        $year = $request['year'];
        $isSeries = $request['isSeries'];
        $movieLength = $request['movieLength'];
        $seriesLength = $request['seriesLength'];
        $genre = $request['genre'];
        $country = $request['country'];

        $currentPage = $request->input('currentPage', 1);

        $TypeID = null;
        $genreID = null;
        $countryID = null;

        if($type) {
            $typeQuery = Type::where('name', $type)->first();

```



```

        if($typeQuery) {
            $typeID = $typeQuery->id;
        } else {
            throw new \InvalidArgumentException("Такого типа не
существует");
        }
    }

    if($genre) {
        $genreQuery = Genre::where('name', $genre)->first();

        if($genreQuery) {
            $genreID = $genreQuery->id;
        } else {
            throw new \InvalidArgumentException("Такого жанра не
существует");
        }
    }

    if($country) {
        $countryQuery = Countrie::where('country', $country)-
>first();

        if($countryQuery) {
            $countryID = $countryQuery->id;
        } else {
            throw new \InvalidArgumentException("Такой страны не
существует");
        }
    }

    $query = Movie::query();

    if ($nameMovie) {
        $query->where('name', $nameMovie);
    }

    if ($kpID) {
        $query->where('externalID', $kpID);
    }

    if ($alternativeName) {
        $query->where('alternativeName', $alternativeName);
    }

    if ($typeID) {
        $query->where('type_id', $typeID);
    }

    if ($year) {
        $query->where('year', $year);
    }

```

```

if ($isSeries) {
    $query->where('isSeries', $isSeries);
}

if ($movieLength) {
    $query->where('movieLength', $movieLength);
}

if ($seriesLength) {
    $query->where('seriesLength', $seriesLength);
}

if ($genreID) {
    $query->whereHas('genres', function ($query) use ($genreID) {
        $query->where('genre_id', $genreID);
    });
}

if ($countryID) {
    $query->whereHas('countries', function ($query) use ($countryID) {
        $query->where('country_id', $countryID);
    });
}

if ($query->getQuery()->wheres) {
    $movies = $query->paginate(10, ['*'], 'page', $currentPage);
} else {
    $movies = collect(); // Пустая коллекция
}

if($query){
    return [
        "status" => true,
        "type" => 0,
        "message" => '200 ok',
        "data" => $movies,
    ];
} else {
    return [
        "status" => true,
        "type" => 0,
        "message" => '200 ok',
        "data" => null,
    ];
}

} catch (\Exception $e) {
    return [
        "status" => false,
        "type" => 1,
        "message" => $e->getMessage(),
    ];
}

```

```

        ];
    }
}

public static function getRoles() {
    $roles = Role::select('name')->get();

    if ($roles->isNotEmpty()) {
        $res = [
            "status" => true,
            "type" => 0,
            "message" => "200 ok",
            "items" => $roles,
        ];
    } else {
        $res = [
            "status" => false,
            "type" => 1,
            "message" => "Произошла ошибка"
        ];
    }

    return $res;
}

public static function takeRole(Request $request) {
    try {
        $email = $request['email'];
        $role = $request['role'];

        $user = User::where('email', $email)->first();

        if(!$user) {
            throw new \InvalidArgumentException("Пользователя с та-
ким email не существует");
        }

        $existedRole = Role::where('name', $role)->first();

        $roleID = null;

        if($existedRole) {
            $roleID = $existedRole->id;
        } else {
            throw new \InvalidArgumentException("Такой роли не суще-
ствует");
        }

        $user->role_id = $roleID;
        $user->save();

        return [
            "status" => true,
            "type" => 0,

```

```

        "message" => 'Роль пользователя обновлена',
    ];

    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}

public static function getSuggestions() {
    try {
        $suggestions = AddMovie::all();

        if ($suggestions->isNotEmpty()) {
            $formattedSuggestions = $suggestions->map(function
($suggestion) {
                $createdAt = Carbon::parse($suggestion->created_at);
                $suggestion->formatted_created_at = $createdAt->for-
mat('d/m/y');
                return $suggestion;
            });

            $res = [
                "status" => true,
                "type" => 0,
                "message" => "200 ok",
                "items" => $formattedSuggestions,
            ];
        } else {
            $res = [
                "status" => false,
                "type" => 1,
                "message" => "Произошла ошибка"
            ];
        }

        return $res;
    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}

public static function postAnswerForAddMovie(Request $request) {
    try {
        $userID = $request['userID'];
        $ticketID = $request['ticketID'];
        $status = $request['status'];
    }
}

```

```

        $suggestions = AddMovie::where('user_id', $userID)-
>where('id', $ticketID)->first();

        if ($suggestions) {
            $suggestions->status = $status;
            $suggestions->save();

            $res = [
                "status" => true,
                "type" => 0,
                "message" => "200 ok",
            ];
        } else {
            $res = [
                "status" => false,
                "type" => 1,
                "message" => "Произошла ошибка"
            ];
        }

        return $res;
    } catch (\Exception $e) {
        return [
            "status" => false,
            "type" => 1,
            "message" => $e->getMessage(),
        ];
    }
}

```

Листинг Б.3 – Методы контроллера AdminController

```

public static function suggestMovie(Request $request) {
    try {
        $userMovieSelectedTypeOption = $request['userMovieSelectedTypeOption'] ?? null;
        $userMovieSelectedRestrictOption = $request['userMovieSelectedRestrictOption'] ?? null;
        $userIsSeries = $request['userIsSeries'] ?? null;
        $userYear = $request['userYear'] ?? null;
        $userAlternativeName = $request['userAlternativeName'] ?? null;

        $userBudget = $request['userBudget'] ?? null;
        $userCountry = $request['userCountry'] ?? null;
        $userDescription = $request['userDescription'] ?? null;
        $userMovieLength = $request['userMovieLength'] ?? null;
        $userNameMovie = $request['userNameMovie'] ?? null;
        $userRatingMpaa = $request['userRatingMpaa'] ?? null;
        $userShortDescription = $request['userShortDescription'] ?? null;

        $userSeriesLength = $request['userSeriesLength'] ?? null;
        $userSlogan = $request['userSlogan'] ?? null;
        $userID = $request['userID'] ?? null;
    }
}

```

```

$typeID = null;
$genreID = null;
$countryID = null;
$restrictID = null;

if($userMovieSelectedTypeOption) {
    $type = Type::where('name', $userMovieSelectedTypeOption)->first();

    if($type) {
        $typeID = $type->id;
    } else {
        throw new \InvalidArgumentException("Такого типа не существует");
    }
}

if($userMovieSelectedRestrictOption) {
    $restrict = Restrict::where('restriction', $userMovieSelectedRestrictOption)->first();

    if($restrict) {
        $restrictID = $restrict->id;
    } else {
        throw new \InvalidArgumentException("Такого ограничения не существует");
    }
}

$addMovie = AddMovie::create([
    'name' => $userNameMovie,
    'alternativeName' => $userAlternativeName,
    'description' => $userDescription,
    'shortDescription' => $userShortDescription,
    'slogan' => $userSlogan,
    'year' => $userYear,
    'isSeries' => $userIsSeries,
    'ratingMpaa' => $userRatingMpaa,
    'budget' => $userBudget,
    'movieLength' => $userMovieLength,
    'seriesLength' => $userSeriesLength,

    'type_id' => $typeID,
    'restrict_id' => $restrictID,
    'user_id' => $userID,

    'status' => 'review',
]);

if(!$addMovie) {
    throw new \Exception("Запись не была создана");
}

```

```

$addMovie->genres()->attach($genreID);
$addMovie->countries()->attach($countryID);

return [
    "status" => true,
    "type" => 0,
    "message" => 'Фильм предложен',
];
} catch (\Exception $e) {
    return [
        "status" => false,
        "type" => 1,
        "message" => $e->getMessage(),
    ];
}
}

```

Листинг Б.4 – Метод suggestMovie