МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ   
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1-40 05 01 «Информационные системы и технологии»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Компьютерные языки разметки»

Тема «Web-caйт “Cервис для просмотра фильмов”»

**Исполнитель**

студент(ка) 1 курса 2 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Корень М. А.

подпись, дата

**Руководитель**

ассистент \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Харланович А. В.

должность, ученая степень, ученое звание подпись, дата

Допущен(а) к защите \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Харланович А. В.

подпись дата инициалы и фамилия

**Содержание**

[1 Обзор технических методов и программных средств разработки 5](#_Toc103518352)

[1.1 Обзор аналогичных решений 5](#_Toc103518353)

[1.2 Техническое задание 8](#_Toc103518354)

[1.3 Выбор средств реализации программного продукта 8](#_Toc103518355)

[1.4 Вывод 9](#_Toc103518356)

[2 Проектирование страниц веб-сайта 11](#_Toc103518357)

[2.1 Выбор способа вёрстки 11](#_Toc103518358)

[2.2 Разработка прототипа 11](#_Toc103518359)

[2.3 Выбор шрифтового оформления 11](#_Toc103518360)

[2.4 Разработка логотипа 12](#_Toc103518361)

[2.5 Разработка пользовательских элементов 12](#_Toc103518362)

[2.6 Разработка спецэффектов 13](#_Toc103518363)

[2.7 Выбор стилевого оформления 13](#_Toc103518364)

[2.8 Разработка макета 14](#_Toc103518365)

[2.9 Вывод 14](#_Toc103518366)

[3 Реализация структуры веб-сайта 15](#_Toc103518367)

[3.1 Структура HTML-документа 15](#_Toc103518368)

[3.2 Добавление таблиц стилей Sass и CSS 16](#_Toc103518369)

[3.3 Использование стандартов XML (SVG) 19](#_Toc103518370)

[3.4 Адаптивный дизайн веб-сайта 20](#_Toc103518371)

[3.5 Выводы 22](#_Toc103518372)

[4 Тестирование веб-сайта 23](#_Toc103518373)

[4.1 Кроссбраузерность веб-сайта 23](#_Toc103518374)

[4.2 Руководство пользователя 23](#_Toc103518375)

[4.3 Тестирование ссылок 23](#_Toc103518376)

[4.4 Тестирование форм 24](#_Toc103518378)

[4.5 Проверка HTML/CSS 24](#_Toc103518381)

[4.6 Выводы 25](#_Toc103518383)

[Заключение 26](#_Toc103518384)

[Список использованных литературных источников 27](#_Toc103518385)

[Приложение 28](#_Toc103518386)

**Введение**

В настоящее время Интернет становиться все более развитой средой для осуществления коммуникаций. В связи с глобальным развитием сети Интернет, в программировании все более резко начала выделяться отдельная его отрасль — веб-программирование. Сейчас, чтобы привлечь внимание клиентов, покупателей или партнёров, просто необходимо заявить о себе в интернете, путём создания веб-сайта. Эффективно работающий ресурс положительно влияет на уровень продаж компании, укрепляет ее позиции среди конкурентов. Поэтому роль сайта чрезвычайно велика, и от того, насколько правильно он подобран, во многом зависит успешность бизнеса.

Одно из самых распространенных отвлекающих действий - это просмотр фильмов. Будь то для развлечения или по любой другой причине, никогда не помешает время от времени смотреть хороший фильм, как в компании, так и в одиночестве. Однако не всегда нужно покупать их физически или в цифровом виде или ходить в кино, чтобы насладиться ими; Есть множество сайтов, порталов и веб-страниц, на которых можно бесплатно посмотреть фильмы. Целью данного проекта является создание сайта для просмотра фильмов. Веб-сайт предоставляет возможность пользователю просматривать пользователям фильмы абсолютно бесплатно. Также не мало важной задачей данной курсовой работы состоят в том, чтобы создать сайт с адаптивной версткой, разработать уникальный дизайн, делающий нахождение пользователя на сайте максимально комфортным, и обеспечить удобный просмотр информации. Курсовой проект включает в себя 4 разделов: введение, содержание, обзор технических методов и программных средств разработки веб-сайта, последовательность разработки содержания и структуры веб-сайта, программная реализация проекта, описание хода работы программного средства с копиями экрана, заключение, список использованной литературы, приложение. Основная цель курсового проекта — развитие навыков верстки адаптивного и кроссбраузерного сайта при помощи HTML и CSS, получение опыта работы с базой данных, обучение программному управлению элементами веб-страницы, создание полноценного проекта для прямого использования пользователями.

# 1 Обзор технических методов и программных средств разработки

## **1.1 Обзор аналогичных решений**

Курсовой проект представляет из себя веб-сайт для просмотра фильмов. В сфере кино уже существуют некоторые аналогичные решения. В этом разделе будут приведены веб-сайты существующих онлайн-кинотеатров.

Сайт kinogoo.by. Сайт имеет приятную для глаз цветовую палитру. Шрифт читабельный. Также сайт имеет удобную систему навигации. Дизайн представлен на рисунке 1.1.

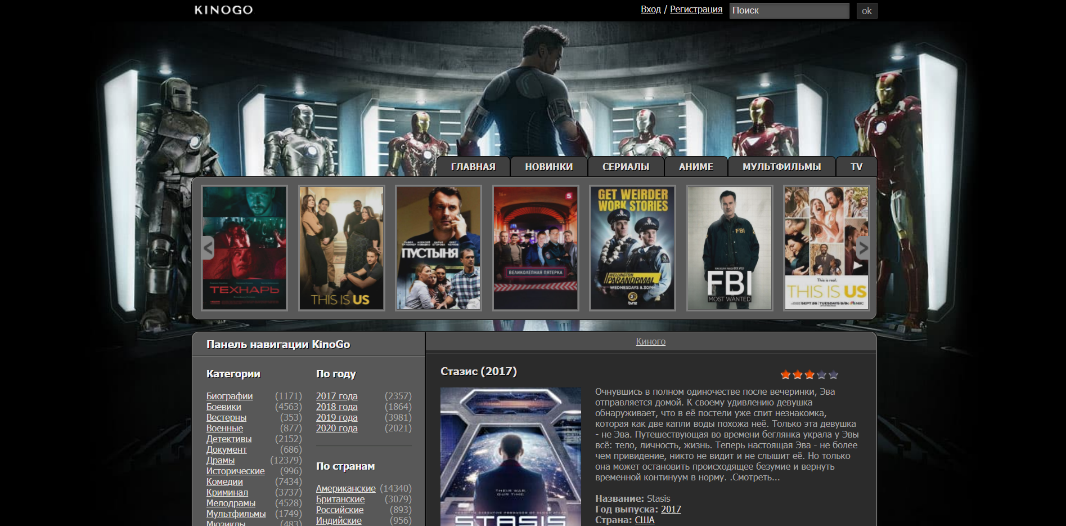


Рисунок 1.1 – Пример дизайна главной страницы

Также хорошо проработана мобильная версия сайта, горизонтальное меню и меню сортировки жанров становятся выпадающими рисунок 1.2

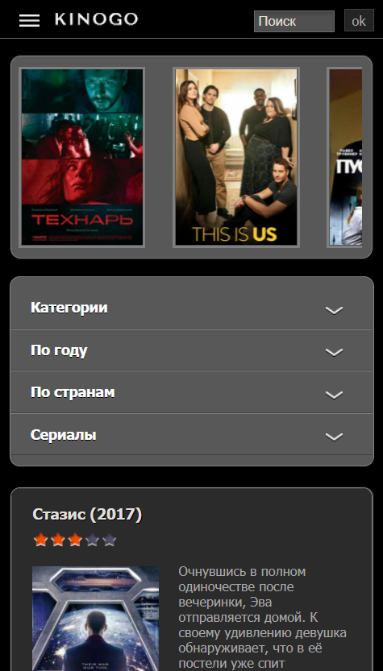


Рисунок 1.2 – Пример дизайна главной страницы мобильная версия

Ссылку на данный сайт можно увидеть в списке литературы [3].

Следующий сайт kinopoisk.ru, а конкретно их онлайн-кинотеатр. При заходе в раздел “магазин”, пользователю предоставляют возможность посмотреть наиболее популярный фильм или сериал на данный момент рисунок 1.3. Хотелось бы выделить интересную и весьма удобную функцию подбора фильмов и сериалов через прокручивающуюся ленту. рисунок 1.4

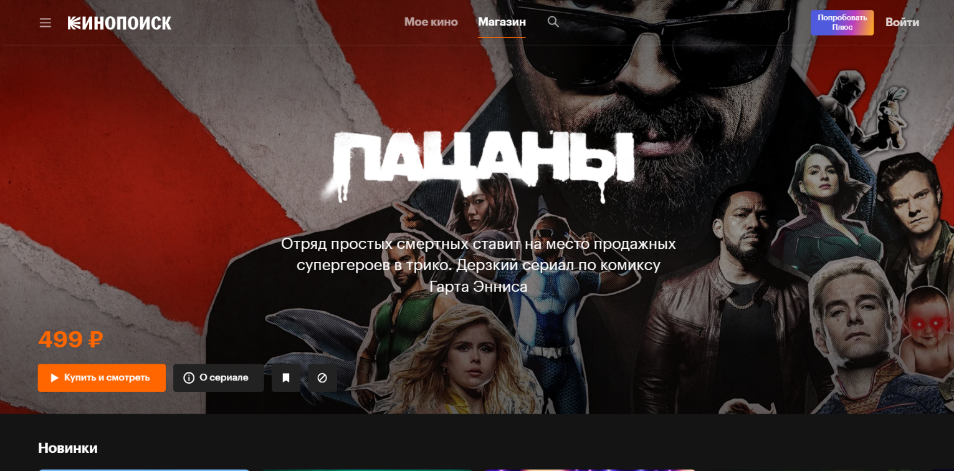


Рисунок 1.3 – Пример дизайна главной страницы

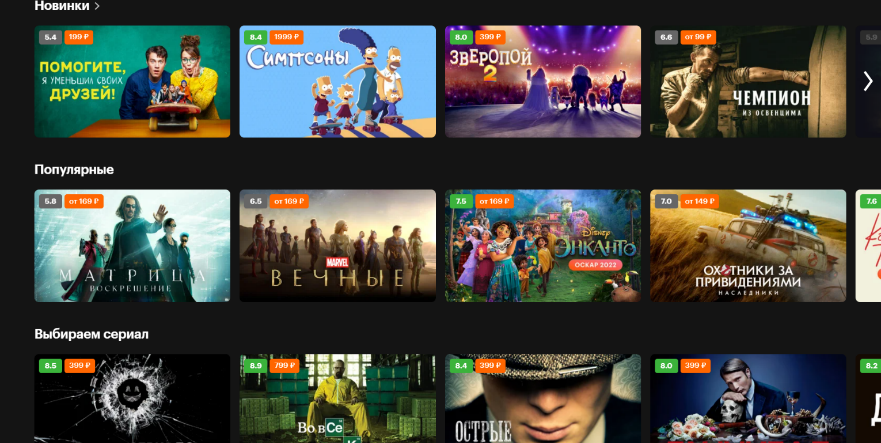


Рисунок – 1.4 – Пример дизайна главной страницы

При просмотре мобильной версии сайта не было найдено весомых отличий. рисунок 1.5

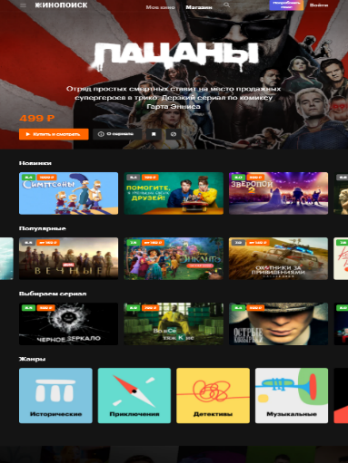


Рисунок 1.5 – Пример дизайна главной страницы мобильная версия

Ссылку на данный сайт можно увидеть в списке литературы [4].

Также в качестве последнего аналога хотелось бы взять сайт megogo.net. По сравнению с останольными сайтами имеет менее приятный дизайн ,а также шрифты рисунок 1.6.

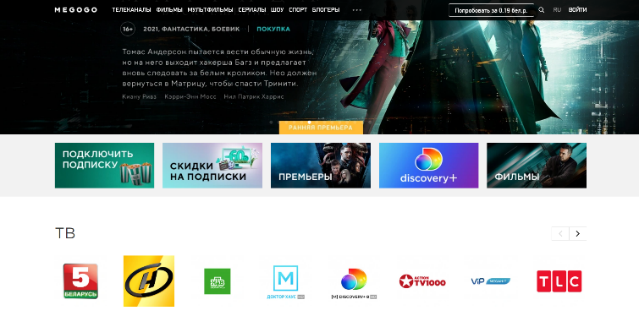


Рисунок 1.6 – Пример дизайна главной страницы

Весьма разочаровывающим моментом стало недоработатка раздела “Вопросы и ответы”, шрифт не имеет дизайна, также он очень мелкий и практически не читабельный. Дизайн представлен на рисунке 1.7.



Рисунок 1.7 – Пример дизайна “вопросы и ответы” страницы

В мобильной версии сайта практически отсутствует адаптивность главной страницы рисунок 1.8.

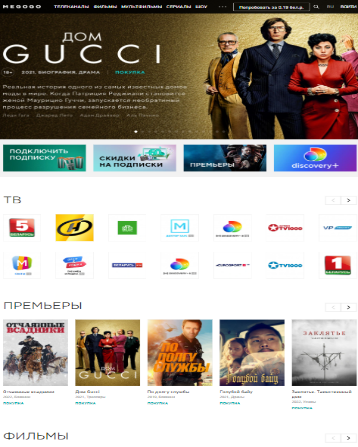


Рисунок 1.8 – Пример дизайна главной страницы

Таким образом при создании сайта я буду руководствоваться в основном первыми двумя аналогами.

Ссылку на данный сайт можно увидеть в списке литературы [5].

## **1.2 Техническое задание**

В данном проекте требуется создать рабочий онлайн-кинотеатр. Сайт должен содержать фильмы. Основным контентом являются фильмы и трейлеры, а также их описания.

Сайт должен быть интерактивным, содержать актуальную информацию. Данный сайт предназначен для желающих провести свой досуг за просмотром фильма.

Основными требованиями к программному продукту являются адаптивность, лёгкость. На создаваемом сайте не может быть больших программных функций или очень сложных анимаций – всё это может сильно замедлить скорость загрузки информации и ухудшить её восприятие. Адаптивность разрабатывалась исходя из существующих актуальных браузеров и устройств. Необходимо реализовать макет для главной страницы, а также странице с фильмом.

Основные требования к исполнителю: создание дизайна, проведение анализа существующих решений, создание прототипа, адаптивность и кроссбраузерность сайта (корректное отображение в браузерах, таких как Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.). В проекте используется гипертекстовый язык разметки HTML, CSS/Sass, графические элементы в формате SVG; данные хранятся в XML-формате.

## **1.3 Выбор средств реализации программного продукта**

Будет приведён основной набор, то есть обязательный к применению в требованиях к проекту: HTML5, XML, Sass/CSS3, SVG, библиотека имён XML, библиотека имён SVG. При необходимости будут использованы другие средства как JavaScript для увеличения функционала сайта.

Сайт разрабатывался в текстовом редакторе Visual Studio Code. Страницы сайта могут просматриваться в таких браузерах как Internet Explorer, Microsoft Edge, Safari, Opera, Mozilla Firefox, Google Chrome.

Visual Studio Code — [редактор исходного кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0), разработанный [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) для [Windows](https://ru.wikipedia.org/wiki/Windows), [Linux](https://ru.wikipedia.org/wiki/Linux) и [macOS](https://ru.wikipedia.org/wiki/MacOS), представлен на рисунке 1.6. Позиционируется как «лёгкий» редактор кода для [кроссплатформенной](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) разработки [веб](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)- и [облачных](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%87%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) приложений. Включает в себя [отладчик](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D1%87%D0%B8%D0%BA), инструменты для работы с [Git](https://ru.wikipedia.org/wiki/Git), [подсветку синтаксиса](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D1%81%D0%B2%D0%B5%D1%82%D0%BA%D0%B0_%D1%81%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81%D0%B0) и средства для [рефакторинга](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3). JavaScript, CSS3, PHP, PHPMyAdmin, MySQL.

Подробнее про каждый из них:

**HTML** (от английского**HyperText Markup Language**) — это код, который используется для структурирования и отображения веб-страницы и её контента. Например, контент может быть структурирован внутри множества параграфов, маркированных списков или с использованием изображений и таблиц данных. **HTML** используется, чтобы сообщать вашему браузеру, как отображать веб-страницы, которые вы посещаете. Браузер интерпретирует HTML-документ, выстраивая его структуру (DOM) и отображая ее в соответствии с инструкциями, включенными в этот файл (таблицы стилей, скрипты). Если разметка правильная, то в окне браузера будет отображена страница, содержащая HTML-элементы — заголовки, таблицы, изображения и т.д. В 2014 году был создан HTML5. Разработчики нового языка ориентировались на устранение проблем предыдущих версий и реализацию современных функций:

Высокая скорость работы сайта за счет удаления всего лишнего из кода и перераспределения функциональных элементов.

Сайты на HTML5 используют совместно ресурсы браузеров и удаленных серверов, чтоб делает серфинг и просмотр сайтов более комфортным.

Сайты стали более легкими, что очень важно для мобильного интернета, который работает медленнее стационарного.

Медиа и интерактивные функции сайтов теперь решаются не путем установки на компьютер специальных приложений, а ресурсами самого кода сайта.

Улучшены возможности по работе сайтов на разных операционных системах и с разными браузерами.

JavaScript – это интерпретируемый язык, то есть исходный код скриптов не нуждается в компиляции. Как и PHP, интерпретаторы этого языка автоматически управляют памятью и очищают неиспользуемые блоки, следовательно, программисту не нужно следить за выделением и очисткой памяти, как, например, в C++. JavaScript отличается от большинства объектно-ориентированных языков программирования тем, что в нём отсутствует понятие класса. Все объекты являются копиями уже существующего экземпляра объекта. Синтаксис JavaScript был вдохновлён языками C и Java. Все идентификаторы (названия переменных, функций и т.д.) являются регистр зависимыми. Операторы разделяются при помощи точки с запятой «;», однако использование этого символа необязательно: оператор автоматически завершается в конце строки.

CSS (Cascading Style Sheets, каскадные таблицы стилей) – это язык описания внешнего вида документа, написанного с использованием языка разметки. CSS позволяет представлять один и тот же документ в различных стилях. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

## **1.4 Вывод**

В данном разделе были рассмотрены аналогичные решения, были проанализированы их недостатки и преимущества, для создания в последующем качественного веб-сайта. Были поставлены конкретные задачи для создания интерактивного информационного ресурса. Кроме того, были рассмотрены средства реализации программного продукта, такие как языки разметки, а также редактор кода VS Code. Язык разметки HTML структурирует и отображает контент сайта с помощью параграфов, блоков, списков и таблиц. CSS оформляет веб-страницу и добавляет стили элементам веб-страницы, создавая уникальный дизайн. JavaScript дает возможность взаимодействовать пользователю со страницей, добавляя функциональность отдельным элементам сайта. Все вышеперечисленные пункты понадобятся для достижения поставленной задачи и для создания в дальнейшем качественного продукта.

# 2 Проектирование страниц веб-сайта

## **2.1 Выбор способа вёрстки**

В соответствие с требованиями к проекту, в частности адаптивность и кроссбраузерность сайта, было принято решение об использовании блочная верстка сайта.  концепция блочной вёрстки основана на активном использовании универсальных тегов **<div>,** внутрь которых помещается содержимое, включая другие теги.

Блочная вёрстка лишена недостатков табличной — поисковыми системами она индексируется лучше, её код не такой развесистый, да и блоки **<div>**, которые так любят называть «слоями», изначально задумывались универсальными, то есть «для всего», тогда как **<table>** — это таблица, которую нужно использовать для отображения табличных данных и не более того. Единственный ощутимый минус блочной вёрстки — сделанные на ней сайты могут по-разному отображаться в обозревателях. Чтобы этого избежать, нужно делать вёрстку «кроссбраузерной», то есть одинаково отображаемой любым обозревателем.

Для создания отдельных блоков страницы, такие как горизонтальное меню, будет использована Flexbox-верстка. С помощью этой технологии можно очень просто и гибко расставить элементы в контейнере, распределить доступное пространство между ними, и выровнять их тем или иным способом даже если они не имеют конкретных размеров. CSS Flexbox поддерживается всеми используемые на сегодняшний момент современными браузерами (с использованием префиксов: IE10+, Edge12+, Firefox 2+, Chrome 4+, Safari 3.1+, Opera 12.1+, iOS Safari 3.2, Opera mini, Android 2.1+, Blackberry 7+).

## **2.2 Разработка прототипа**

Прототип — это схема всех или нескольких страниц сайта в виде наброска, эскиза или html-документа, где отображены взаимодействия и структурные элементы будущего сайта: меню, кнопки, формы и другие. На данном этапе разработки веб-сайта, для удобной верстки в дальнейшем, было принято решение о создании прототипа. Полный прототип сайта мы можем увидеть в приложении 1.

## **2.3 Выбор шрифтового оформления**

В данном проекте будут использованы два основных шрифта: Gugi и Playfair Display. Шрифты подключались с помощью библиотеки Google Fonts.

**Gugi** — это шрифт, который отражает визуальные характеристики хангыля, найденные в Hunminjeongeum, и использует органическое движение мяча в качестве метода построения.

Playfair - это переходный дизайн. В эпоху европейского Просвещения в конце 18 века широкие перья были заменены остроконечными стальными ручками в качестве популярного пишущего инструмента того времени. Вместе с развитием технологий печати, чернил и производства бумаги стало печатать высококонтрастные буквы с тонкими линиями, которые все больше отделялись от письменных форм букв.

## **2.4 Разработка логотипа**

Разработка логотипа осуществлялась в графическом редакторе Figma. Дизайн довольно простой. Было принято решение не загромождать его множеством декоративных элементов. Логотип представляет из себя два слова Kino и Online. Получается “Кино онлайн”. Что и отражает саму суть данного сайта. Эти два слова были написаны на черном прямоугольнике (так как прямоугольник черного цвета, то в шапке сайта его не будет видно, сделано это специально из личных соображений), а затем скачаны в pdf формате. После этого, используя сайт image.online-convert.com, конвертированы в svg формат. Дизайн логотипа представлен на рисунке 2.0.



Рисунок 2.1 – Пример дизайна логотипа

## **2.5 Разработка пользовательских элементов**

В проекте будут представлены такой элемент пользовательского интерфейса, как поле и кнопка для поиска по сайту рисунок 2.2.



Рисунок 2.2 – Пример дизайна поиска

Также не мало важным элементом является ссылки на контакты разработчика Web-сайта рисунок 2.3.



Рисунок 2.3 – Пример дизайна “Контакты”

Еще одна доработка будет состоять в ссылке-якоре при нажатии на “Контакты” рисунок 2.4 страничка будет скролиться на footer.

Это является весьма полезной и удобной функцией навигации на сайте.



Рисунок 2.4 – Пример дизайна меню

## **2.6 Разработка спецэффектов**

На сайте будут разработаны некоторые динамические эффекты и анимации, которые позволят придать сайту интерактивность и сделают его более привлекательным для пользователя. На главной странице будет присутствовать анимация изменения цвета выбранного элемента при наведении курсора рисунки 2.5.

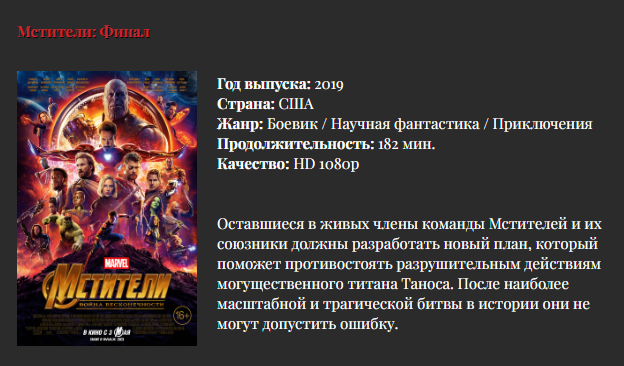


Рисунок 2.5 – Пример дизайна анимации

## **2.7 Выбор стилевого оформления**

Для реализации стилевого оформления сайта было выбрано минималистическое направление с элементами классического веб-дизайна. Такой стиль в наше время весьма популярен, а также привлекает упрощает разработку.

Большинство изображений выдержаны в одном стиле, не перегружены мелкими элементами.

Цветовые решения не яркие, весь сайт выполнен в темных и серых тонах. Основные цвета: #000000, #534F4F, #504949. Цветовая палитра представлена на рисунке 2.6



Рисунок 2.6 - Цветовая палитра

Данная палитра цветов хоть и кажется одинаковой, но в действительности при верстке становиться видно переходы блоков.

## **2.8 Разработка макета**

Макет сайта — нечто среднее между блочной схемой и прототипом. Это оптимальный вариант для клиента, которому не придется переплачивать за интерактивные эффекты прототипа. С другой стороны, макет не выглядит слишком простым как блочная схема. С его помощью верстальщику проще понять, каким именно должен быть сайт. На данном этапе разработке было принято решение о компоновке разработанных элементов и прототипов, и на основе созданного ранее прототипа сайта создать полноценный макет сайта, на который можно будет ориентироваться при создании веб-странички, сам макет можно увидеть в приложении 2.

## **2.9 Вывод**

На данном этапе были созданы прототипы, а затем и макеты сайта для облегчения последующей верстки. Было упорядочено расположение всех элементов на страницах (логотипа, основного контента страницы, расположение картинок, подвала и шапки сайта). Было определено стилевое оформление сайта, то есть основная цветовая гамма, шрифтовое оформление, элементы пользовательского интерфейса, а также эффекты, придающие динамики страницам.

# 3 Реализация структуры веб-сайта

## **3.1 Структура HTML-документа**

Структура сайта была написана на языке HTML5. HTML5 является последним стандартом для браузеров для отображения и взаимодействия с веб-страницами. Утвержденный в 2014 году, это первое обновление для HTML за 14 лет.

Структура моего сайта состоит из семантических тегов таких как: <header>, <footer>, <nav>. А также из большого числа блочных тегов таких как <div>. Ещё использовались теги для создания таблицы <table>. Структура тега <body> представлена в листинге 3.1.

|  |
| --- |
| <body>  <div class="wrapper">  <header>…  <div class="header">  </header>  <div class="city">…  </div>  <footer>…  </footer>  </div>  </body> |

Листинг 3.1 – Содержимое тега <body>

В теге <header> располагается шапка сайта. В ней находится логотип в формате SVG, поисковое поле, также в формате SVG и тег <nav> для создания блоков навигации. Для работы с SVG иконками использовался xml и в дальнейшем тег <svg>, также для создания этих самых иконок тег <path>. И, как я уже сказал, в шапке сайта есть теги навигации <nav> для их успешного использования применялись теги <a>, чтобы создать гиперссылку на другой элемент сайта или же на совершенно другую страницу. Логотип SVG был размещен внутри CSS файла на класс SVG-logo, структура представлена в листинге 3.2

|  |
| --- |
| Html структура:  <div class="logo"><a href="../kyrsach/cur.html"><span class="SVG-logo"></span></a></div>  CSS параметр:      background-image: url(../img/SVG.svg); |

Листинг 3.2 – Пример вставки логотипа

Такое расположение было выбрано для удобной настройки flex-элементов.

## **3.2 Добавление таблиц стилей Sass и CSS**

В процессе разработки веб-сайта использовалась такая таблица стилей как CSS. CSS – это язык, с помощью которого описывается внешний вид документа HTML, XML, XHTML. Название означает «каскадная таблица стилей», или Cascading Style Sheets. CSS-стили незаменимы при оформлении страниц сайтов: в одном файле содержатся сведения об отображении всех элементов документа. Пример подключения CSS к HTML коду.

Использование CSS направлено на то, чтобы задавать визуальное оформление web-страницы. При выполнении разметки элементов языком HTML документ получит основную структуру страницы. Это заголовки различного уровня, параграфы, маркированные и нумерованные списки, отступы, картинки и прочее. Но, если все оставить как есть, то пользователю на странице будет выведен безликий текст, который будет плохо восприниматься.

CSS, в отличие от HTML, может менять настройки текстовых блоков. Пользователь задает шрифт, размер этого шрифта, цвет текста и фона. При создании страницы на сайте пользователь сначала прописывает HTML-кодом, что именно там будет отображаться. А при помощи кода CSS можно задать визуальные настройки текстовой основы. В данном проекте работе несколько CSS файлов каждый из которых отвечает за разные участки сайта всего таких файла получилось 4 их можно посмотреть в листинге 3.3.

|  |
| --- |
| <link rel="stylesheet" href="css/style.css">  <link rel="stylesheet" href="css/dropMenu.css">  <link rel="stylesheet" href="css/kas.css">  <link rel="stylesheet" href="css/gl.css"> |

Листинг 3.3 – ссылки на CSS файлы проекта

В CSS файле style.css я разместил все параметры нужные для правильной структуры блоков <div>, а также его содержимого. В файле dropMenu.css я прописал правильную структуру для <header> а также адаптивность для данного участка кода. Файл kas.css стал своего рода костылём для некоторых недоработок, а также там расположились параметры для главной страницы. Для файла gl.css выпала роль быть специальным обнуляющим селектором, пример которого мы можем увидеть в листинге 3.4.

|  |
| --- |
| \*{  padding: 0px 0px 0px 0px;  margin:  0px 0px 0px 0px;  border: 0px 0px 0px 0px;  } |

Листинг 3.4 – Пример применения обнуляющего селектора

Данные параметры помогают в работе с блочными структурами.

Добавление таблиц стилей CSS позволило упростить создание стилистического оформления страницы. Стили являются удобным, практичным и эффективным инструментом при вёрстке веб-страниц и оформления текста, ссылок, изображений и других элементов. Несмотря на явные плюсы применения стилей, рассмотрим все преимущества CSS, в том числе и незаметные на первый взгляд:

1. Разграничение кода и оформления. Идея о том, чтобы код HTML был свободен от элементов оформления вроде установки цвета, размера шрифта и других параметров, стара как мир. В идеале, веб-страница должна содержать только теги логического форматирования, а вид элементов задаётся через стили. При подобном разделении работа над дизайном и версткой сайта может вестись параллельно.
2. Разное оформление для разных устройств. С помощью стилей можно определить вид веб-страницы для разных устройств вывода: монитора, принтера, смартфона, планшета. Например, на экране монитора отображать страницу в одном оформлении, а при её печати — в другом. Эта возможность также позволяет скрывать или показывать некоторые элементы документа при отображении на разных устройствах.
3. Расширенные по сравнению с HTML способы оформления элементов. В отличие от HTML стили имеют гораздо больше возможностей по оформлению элементов веб-страниц. Простыми средствами можно изменить цвет фона элемента, добавить рамку, установить шрифт, определить размеры, положение и многое другое.
4. Ускорение загрузки сайта. При хранении стилей в отдельном файле, он кэшируется и при повторном обращении к нему извлекается из кэша браузера. За счёт кэширования и того, что стили хранятся в отдельном файле, уменьшается код веб-страниц и снижается время загрузки документов.

Пройдемся по CSS коду, который я использовал при создании сайта. Правило @font-face позволяет определить настройки шрифтов, а также загрузить специфичный шрифт на компьютер пользователя. CSS-свойство font-family определяет приоритетный список из одного или нескольких названий семейства шрифтов и/или общее имя шрифта для выбранного элемента. Атрибут src указывает адрес файла (URL), который будет загружаться во фрейм. Таким образом можно подключить любой понравившийся вам шрифт. Пример подключения google шрифтов в листинге 3.5.

|  |
| --- |
| <link rel="preconnect" href="https://fonts.googleapis.com">      <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin> |

Листинг 3.5 – Пример подключения google шрифтов

Идея о том, чтобы код HTML был свободен от элементов оформления вроде установки цвета, размера шрифта и других параметров, стара как мир. В идеале, веб-страница должна содержать только теги логического форматирования, а вид элементов задаётся через стили. При подобном разделении работа над дизайном и версткой сайта может вестись параллельно.

С помощью стилей можно определить вид веб-страницы для разных устройств вывода: монитора, принтера, смартфона, планшета и др., например, на экране монитора отображать страницу в одном оформлении, а при её печати — в другом. Эта возможность также позволяет скрывать или показывать некоторые элементы документа при отображении на разных устройствах. Легче всего адаптив для страницы сделать, используя медиа-запросы. Пример их использования в моем коде можно увидеть в листинге 3.6.

|  |
| --- |
| @media (max-width: 700px) {    .city{      margin: 260px 1px 0;      margin-top: 0px;    }    .main-content{      float: none;      width: auto;      height: auto !important;    }    .extra-container{      display: none;      float: none;    }    footer{      display: block;    }    li{      font-size: large;    }  } |

Листинг 3.6 – Пример использования медиа-запросов

Расширенные по сравнению с HTML способы оформления элементов. В отличие от HTML стили имеют гораздо больше возможностей по оформлению элементов веб-страниц. Простыми средствами можно изменить цвет фона элемента, добавить рамку, установить шрифт, определить размеры, положение и многое другое.

Ускорение загрузки сайта. При хранении стилей в отдельном файле, он кэшируется и при повторном обращении к нему извлекается из кэша браузера. За счёт кэширования и того, что стили хранятся в отдельном файле, уменьшается код веб-страниц и снижается время загрузки документов.

## **3.3 Использование стандартов XML (SVG)**

В процессе разработки веб-сайта, XML использовался для подключения SVG к веб-сайту.

SVG — XML язык разметки на основе векторной графики. По сути это текстовый файл, который является открытым веб-стандартом для описания двухмерных векторных изображений без потери качества при масштабировании. Также поддерживает интерактивность и анимацию. Разрабатывается с 1999 года и используется для того, чтобы описать с помощью языка программирования графические элементы сайта.

В моем сайте SVG использовалось для логотипа. Логотип был создан через графический онлайн-редактор Figma. Он был первоначально сохранен в другом формате, но посредствам онлайн-ресурсов удалось формат перевести в SVG. Помимо этого, использовались и различные иконки для социальных сетей такие как: VK, Facebook, Twitter и Instagram, которые также находятся в SVG формате. Структура логотипа представлена на листинге 3.7.

|  |  |
| --- | --- |
| <?xml version="1.0" standalone="no"?>  <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20010904//EN"  "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">  <svg version="1.0" xmlns="http://www.w3.org/2000/svg"  width="137.000000pt" height="41.000000pt" viewBox="0 0 137.000000 41.000000"  preserveAspectRatio="xMidYMid meet">  <g transform="translate (0.000000,41.000000) scale (0.100000,-0.100000)"  fill="#000000" stroke="none">  <path d="M0 205 l0 -205 685 0 685 0 0 205 0 205 -685 0 -685 0 0 -205z m270  75 c0 -5 -7 -10 -15 -10 -8 0 -15 5 -15 10 0 6 7 10 15 10 8 0 15 -4 15 -10z  m680 -80 c0 -53 -4 -90 -10 -90 -6 0 -10 37 -10 90 0 53 4 90 10 90 6 0 10  -37 10 -90z m60 80 c0 -5 -7 -10 -15 -10 -8 0 -15 5 -15 10 0 6 7 10 15 10 8  0 15 -4 15 -10z m-910 -35 c0 -19 4 -35 9 -35 6 0 22 16 37 35 14 19 33 35 41  35 10 0 2 -14 -22 -41 l-37 -41 43 -44 c55 -56 29 -58 -28 -2 l-43 42 0 -42  c0 -23 -4 -42 -10 -42 -6 0 -10 35 -10 85 0 50 4 85 10 85 6 0 10 -16 10 -35z2z  m-465 -80 c0 -37 -4 -65 -10 -65 -6 0 -10 28 -10 65 0 37 4 65 10 65 6 0 10  -28 10 -65z m60 53 c0 -9 3 -9 12 0 15 15 51 15 66 0 14 -14 16 -118 2 -118  -5 0 -10 18 -10 39 0 51 -8 71 -29 71 -28 0 -41 -21 -41 -67 0 -24 -4 -43 -10  -43 -6 0 -10 28 -10 65 0 37 4 65 10 65 6 0 10 -5 10 -12z  -53 c0 -37 -4 -65 -10  -65 -6 0 -10 28 -10 65 0 37 4 65 10 65 6 0 10 -28 10 -65z m138 -2 c2 -37 -1  -63 -7 -63 -5 0 -11 24 -13 52 -2 43 -7 54 -22 56 -27 6 -46 -24 -46 -70 0  -21 -4 -38 -10 -38 -6 0 -10 28 -10 63 l0 64 53 -1 52 -1 3 -62z m136 51 c9  -7 44 -16z"/>  <path d="M627 242 c-21 -23 -22 -76 -1 -96 37 -37 104 -6 104 49 0 56 -67 86  -103 47z"/>  <path d="M470 206 c-28 -35 11 -93 48 -70 35 22 22 84 -18 84 -10 0 -23 -7  -30 -14z"/>  <path d="M1217 213 c-15 -15 -5 -23 29 -23 31 0 35 2 24 15 -13 16 -42 20 -53  8z"/>  </g>  </svg> |  |

Листинг 3.7 – Пример использования SVG

Также при создании логотипа было принято решение о применении нестандартного шрифта для создания более подходящей атмосферы на сайте.

## **3.4 Адаптивный дизайн веб-сайта**

Сегодня количество пользователей, которые заходят в Интернет с мобильных устройств очень велико, и с каждым днем их становится все больше. А потому адаптация веб-сайта под маленькие экраны – актуальная задача и для курсового проекта.

Для тонкой настройки элементов во время уменьшения экрана были использованы медиа запросы. CSS3-медиазапросы используются для создания адаптивного дизайна. Он имеет множество параметров, такие как max-width (отвечающий за максимальную ширину) или min-width (минимальную ширину). В основном в своем проекте я использовал параметр max-width пример можно увидеть в листинге 3.8.

|  |
| --- |
| @media(max-width: 1000px) {    body{      background: #000 !important;    }    .wrapper{      padding-top: 10px;      width: auto;      padding-bottom: 7px;      font-size: 13px;    }    .list{      width: auto;    }    .main-content{      float: none;      width: auto;    } |

Листинг 3.8 – Пример использования параметров max-width в 900px

Я использовал 3 разрешения после которых применялись новые свойства к элементам: 1000 px, 700 px и 900 px. 1000 px и более — это разрешение для персональных компьютеров. 900 px — разрешение для планшетов. 700 px — разрешение для мобильных устройств. Хоть в предыдущем пункте уже и было представлено использование медиа-запросов для адаптива, но хотелось бы выделить использование также и расширений экрана на 900px листинг 3.9

|  |
| --- |
| @media (max-width: 900px) {    .li\_block{      display: block;      float: none;      padding: 0 20px 10px 0;      position: relative;      display: none;    }    .li\_block img{      border: 0!important;      width: 50%;      margin: 0 15px 5px 0;      display: block;    }  } |

Листинг 3.9 – Пример использования медиа-запросов для разрешения в 900px

Медиа-запросы являются одним из самых легких вариантов при написании адаптива. Также в данном проекте было принято решение о добавлении такой структуры как бургер-меню. Такое меню имеет такие плюсы как:

1. Минимизирует количество визуального пространства, которое требуется меню
2. Довольно обычное явление сейчас, когда значительная часть пользователей понимает его функциональность.
3. Облегчает жизнь дизайнеров, убирая все лишнее в мобильном представлении

Но каким бы хорошим решением это не было у всего есть и минусы как раз о них:

1. Требуется дополнительное тестирование, чтобы убедиться, что меню будет хорошо работать с программами чтения с экрана.
2. Может быть слишком сложным с наличием дочерних выпадающих элементов, текста справки и т.д.
3. Добавляет раздувание к проектам, которым может потребоваться дополнительный JavaScript для отображения этих меню (меню CSS только для гамбургеров позволяет избежать этой проблемы)
4. Добавляет дополнительную точку взаимодействия с пользователем (нажмите, чтобы открыть, затем перейдите к просмотру доступных параметров)

Саму реализацию, а также фрагменты листинга можно увидеть в приложении 3.

## **3.5 Выводы**

На данном этапе была реализована структура на HTML, а также созданы таблицы стилей CSS. Было продемонстрировано использование стандартов JSON (SVG). Было определено стилевое оформление веб-сайта, созданы анимации, а также веб-сайт был подготовлен к следующему этапу – к тестированию.

# 4 Тестирование веб-сайта

## **4.1 Кроссбраузерность веб-сайта**

Существует множество различных браузеров с миллионами пользователей у каждого из них. Поэтому при создании веб-сайта или веб-приложения стоит проверить его совместимость со всеми имеющимися браузерами. При этом лучший и самый удобный способ сделать это — использовать кроссбраузерные инструменты проверки.

Кроссбраузерность – это способность веб-ресурса отображаться одинаково и работать во всех популярных браузерах, без перебоев в функционировании и ошибок в верстке, а также с одинаково корректной читабельностью контента.

У каждого браузера есть свои встроенные, экспериментальные или нестандартные свойства и для того, чтобы они корректно работали было принято решения о внедрении вендорных префиксов. Они были автоматически прописаны после компиляции файла с расширением Sass.

После написания основной структуры страницы на HTML и внешнего стилевого оформления появился вопрос о тестировании. Веб-сайт был открыт при помощи различных браузеров. После этого было выявлено, что во всех браузерах сайт выглядит одинаково.

При тестировании данного проекта на кроссбраузерность были задействованы множество сайтов, а также браузеров одними из таких были Firefox, Internet Explorer, Safari, Chrome, также Opera.

Как результат всех тестов является тот факт, что этот проект является кроссбраузерным.

## **4.2 Руководство пользователя**

Данный веб-сайт предоставляет возможность просматривать фильмы пользователю.

Для обеспечения надежности данного веб-сайта реализовано:

1. Разработан удобный пользовательский интерфейс, чтобы даже человек, не работавший с компьютером достаточно, быстро мог разобраться в этом веб-сайте.
2. Вверху главной страницы, вы можете увидеть навигационную панель. Она содержит основные тематические разделы веб-сайта: “Фильмы”, “Сериалы”.
3. Во вкладке “Фильмы” и” Сериалы” пользователь может изучить всё содержимое данного сайта.

## **4.3 Тестирование ссылок**

Для начала тестирования мне кажется будет хорошей практикой проверить функционал всех ссылок. По моему мнению сайт довольно хорошо подготовлен к такому роду тестов, так как почти все элементы являются ссылками на страницы их прямого назначения. Единственная неприятная новость — это то что жанры хоть по своей структуре и являются ссылками, но не работают.

## **4.4 Тестирование форм**

На всех страничка присутствует единственная форма и это форма поиска по сайту рисунок 4.1. После непродолжительных тестов и попыток, что-то найти оказалось, что форма не является рабочей и служит исключительно декоративной цели.



Рисунок 4.1 – Поисковая форма

Данный факт весьма снижает ценность ,качество ,а также желание пользователей посещать, а также пользоваться сайтом где не доработана функция поиска фильмов.

## **4.5 Проверка HTML/CSS**

Последняя проверка, но не менее важной является проверка синтаксиса как HTML структуры, так и CSS формы на наличие устаревших или бесполезный тегов, или CSS параметров. По итогу исследования данных структур не было выявлено никаких устаревших тегов, ошибочных параметров, бесполезных свойств и т.д.

Также для проверки кода на какие-либо ошибки было принято решение провести внешнюю проверку, а конкретнее на специализированных на данной тематике сайтах по типу csslint.net данный ресурс можно также увидеть в списке литературы [6]. Результат такой проверки стал весьма приятным, его можно увидеть на рисунке 4.2.

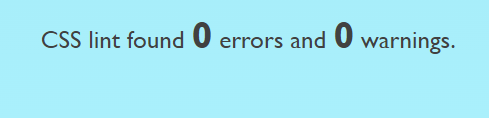


Рисунок 4.2 – Результат про верки на csslint.net

Таким образом можно сказать, что код является дееспособным.

## **4.6 Выводы**

На данном этапе было проанализировано поведение веб-сайта при использовании с разных устройств, а также веб-ресурс был протестирован в большом количестве браузеров. Также было добавлено руководство пользователя с прикрепленной план-схемой. По итогу выполнения данного этапа для веб-сайта был разработан адаптивный дизайн и кроссбраузерность. Работа, проделанная с сайтом, позволяет пользователям ориентироваться по сайту. Лишних элементов нет, что упрощает пользование сайтом. При пользовании сайтом при помощи мобильного устройства некоторые элементы меняют свое положение, для более удобного пользования.

# Заключение

Целью курсовой работы была разработать веб-сайт, который бы предоставлял информацию о сериалах. Сайт предоставляет возможность ознакомиться с интересующим тебя сериалом, а также почитать актуальные новости из мира кино.

Для создания макетов и прототипов сайта использовалась программа «Figma». Figma — онлайн-сервис для разработки интерфейсов и прототипирования с возможностью организации совместной работы в режиме реального времени.

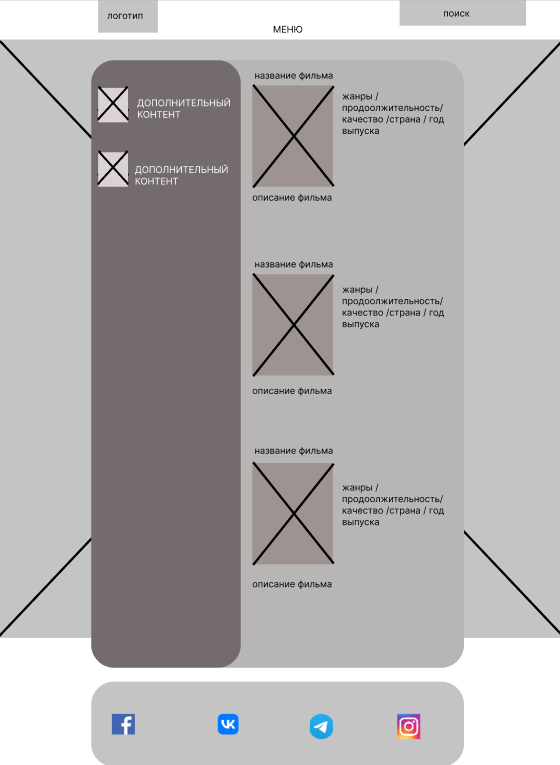
Результатом курсового проекта стал готовый к работе веб-сайт, в котором размещена информация о некоторых сериалах. Конечно, в дальнейшем возможно расширение списка сериалов, а также объём информации который пользователь сможет получить после посещение данного сайта.

**Список использованных литературных источников**

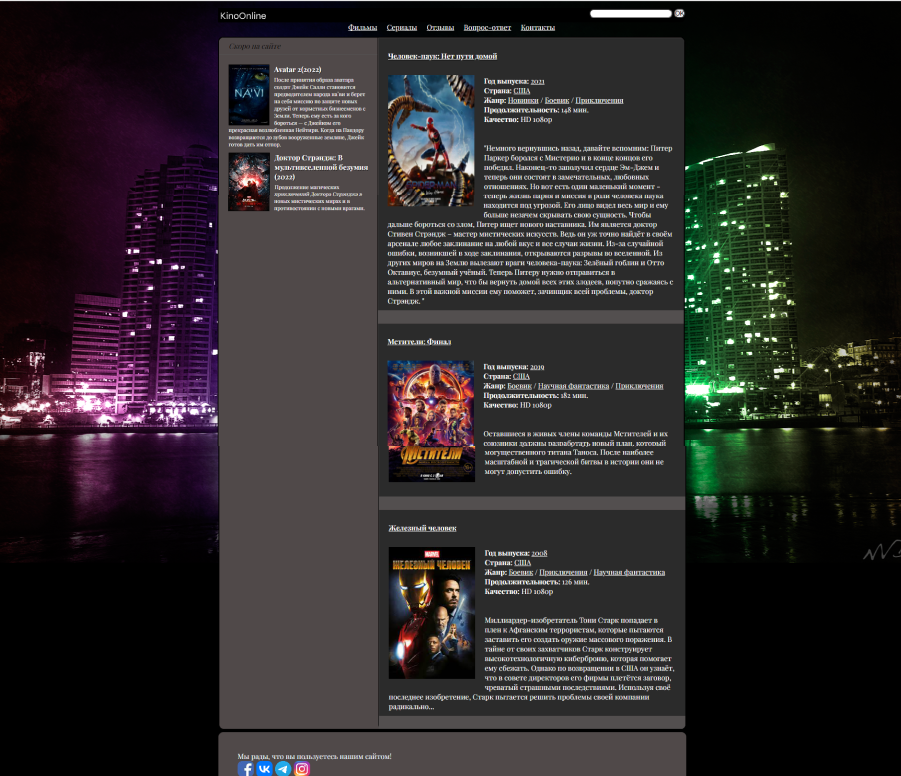
1. Документация по JavaScript/HTML [Электронный ресурс] / Справочный Режим доступа : <https://developer.mozilla.org> – Дата доступа 02.05.2022.
2. Документация по Figma [Электронный ресурс] / Справочный Режим доступа : <https://help.figma.com> – Дата доступа 02.05.2022.
3. Аналоговый ресурс использованный при разработке [Электронный ресурс] / Режим доступа: <https://kinogo.biz/> – Дата доступа 15.05.2022.
4. Аналоговый ресурс использованный при разработке [Электронный ресурс] / Режим доступа: <https://hd.kinopoisk.ru>– Дата доступа 15.05.2022.
5. Аналоговый ресурс использованный при разработке [Электронный ресурс] / Режим доступа: <https://megogo.net>– Дата доступа 15.05.2022.

**Приложение**

**Приложение 1 Прототипы веб-страниц**

****

**Приложение 2 Макет структуры веб-сайта**



**Приложение 3 Создание бургер меню**

|  |  |
| --- | --- |
| До активации меню | После активации |

Элементы листинга:

|  |
| --- |
| .nav {      font-size: 30px;      position: relative;    }    [for=burger]{      display: block;      color: aliceblue;      position: relative;      text-align: center;      width: 30px;      font-size: 80px;    }    nav{      display: none;    }    #burger:checked + label + nav{      display: block;    } |