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-сайт "Информационный сервис о кино"»

**Исполнитель**

студент(ка) 1 курса \_\_ группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ П.С. Синяк

подпись, дата

**Руководитель**

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

должность, ученая степень, ученое звание подпись, дата

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

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

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

дата, подпись

Курсовой проект защищен с оценкой

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

подпись дата инициалы и фамилия

# Содержание

[Содержание 2](#_Toc103643565)

[Введение 3](#_Toc103643566)

[1 Обзор технических методов и программных средств разработки 4](#_Toc103643567)

[1.1 Обзор аналогичных решений 4](#_Toc103643568)

[1.2 Техническое задание 7](#_Toc103643569)

[1.3 Выбор средств реализации программного продукта 8](#_Toc103643570)

[1.4 Вывод 10](#_Toc103643571)

[2. Проектирование страниц веб-сайта 11](#_Toc103643572)

[2.1 Создание прототипа 11](#_Toc103643573)

[2.2 Выбор способа вёрстки 11](#_Toc103643574)

[2.3 Выбор стилевого оформления 11](#_Toc103643575)

[2.4 Выбор шрифтового оформления 12](#_Toc103643576)

[2.5 Разработка логотипа 12](#_Toc103643577)

[2.6 Разработка пользовательских элементов 12](#_Toc103643578)

[2.7 Разработка спецэффектов 14](#_Toc103643579)

[2.8 Вывод 15](#_Toc103643580)

[3. Реализация структуры веб-сайта 16](#_Toc103643581)

[3.1 Структура HTML-документа 16](#_Toc103643582)

[3.2 Добавление таблиц стилей CSS 18](#_Toc103643583)

[3.3 Использование стандартов XML (SVG) 21](#_Toc103643584)

[3.4 Использование jQuery 21](#_Toc103643585)

[3.5 Вывод 22](#_Toc103643586)

[4. Тестирование веб-сайта 23](#_Toc103643587)

[4.1 Адаптивный дизайн веб-сайта 23](#_Toc103643588)

[4.2 Кроссбраузерность веб-сайта 23](#_Toc103643589)

[4.3 Руководство пользователя 23](#_Toc103643590)

[4.4 Выводы 26](#_Toc103643591)

[Заключение 27](#_Toc103643592)

[Список литературы 28](#_Toc103643593)

# Введение

Сегодня уже никого не удивишь широким распространением Интернета во всех сферах жизни. В связи с этим быстро начала распространяться такая отрасль, как веб-программирование.

Веб-сайт стал главным критерием для привлечения внимания клиентов и пользователей. 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. Как только люди найдут сайт в Интернете, они заинтересуются продуктом или услугой, предоставляемыми данной организацией, и смогут связаться с компанией благодаря информации на веб-сайте, что также даёт возможность увеличить продажи. Поскольку веб-сайт работает круглосуточно и без выходных, то компания может легко публиковать обновления и объявления для своих клиентов.

Тема моего курсового проекта Web-сайт «Информационный сервис о кино». Web-сайт будет связан с сериалами. Сегодня сериалы становятся всё более популярными. Люди проводят большое количество своего времени за просмотрами сериалов. И в 21 веке их можно рассматривать как один из многих видов творчества наравне с фильмами. Из-за большого количества различных сериалов разных жанров и тематики, людям всё более проблематично выбрать сериал, который сможет их заинтересовать. Поэтому существует большое количество онлайн-сервисов, которые предоставят подробную информацию о нужном тебе сериале. Мой веб-сайт как раз и будет выполнять данные задачи.

Цель курсового проекта:разработать Web-сайт "Информационный сервис о кино" с использованием HTML5 и XML, а также с применением CSS3 и Java Script.

Задачи:

- проанализировать существующие языки разметки;

- разработать макет и прототип сайта;

- разработать структуру веб-сайта;

- наполнить сайт информацией по теме;

- протестировать веб-сайт.

Целевая аудитория сайта будет рассчитана на людей от 12 до 35 лет. Так как на сайте будет присутствовать популярные и новые сериалы, которые чаще всего нацелены именно на данную группу людей.

# 1 Обзор технических методов и программных средств разработки

# 1.1 Обзор аналогичных решений

Курсовой проект представляет из себя онлайн-сервис для предоставления информации о сериалах. В нынешнее время таких онлайн-сервисов полным-полно. Поэтому в данном разделе будут приведены веб-сайты о сериалах.

Сайт kinopoisk. Данный сайт располагает информацией не только о сериалах, но и о фильмах. Дизайн достаточно приятный, он представлен на рисунке 1.1.

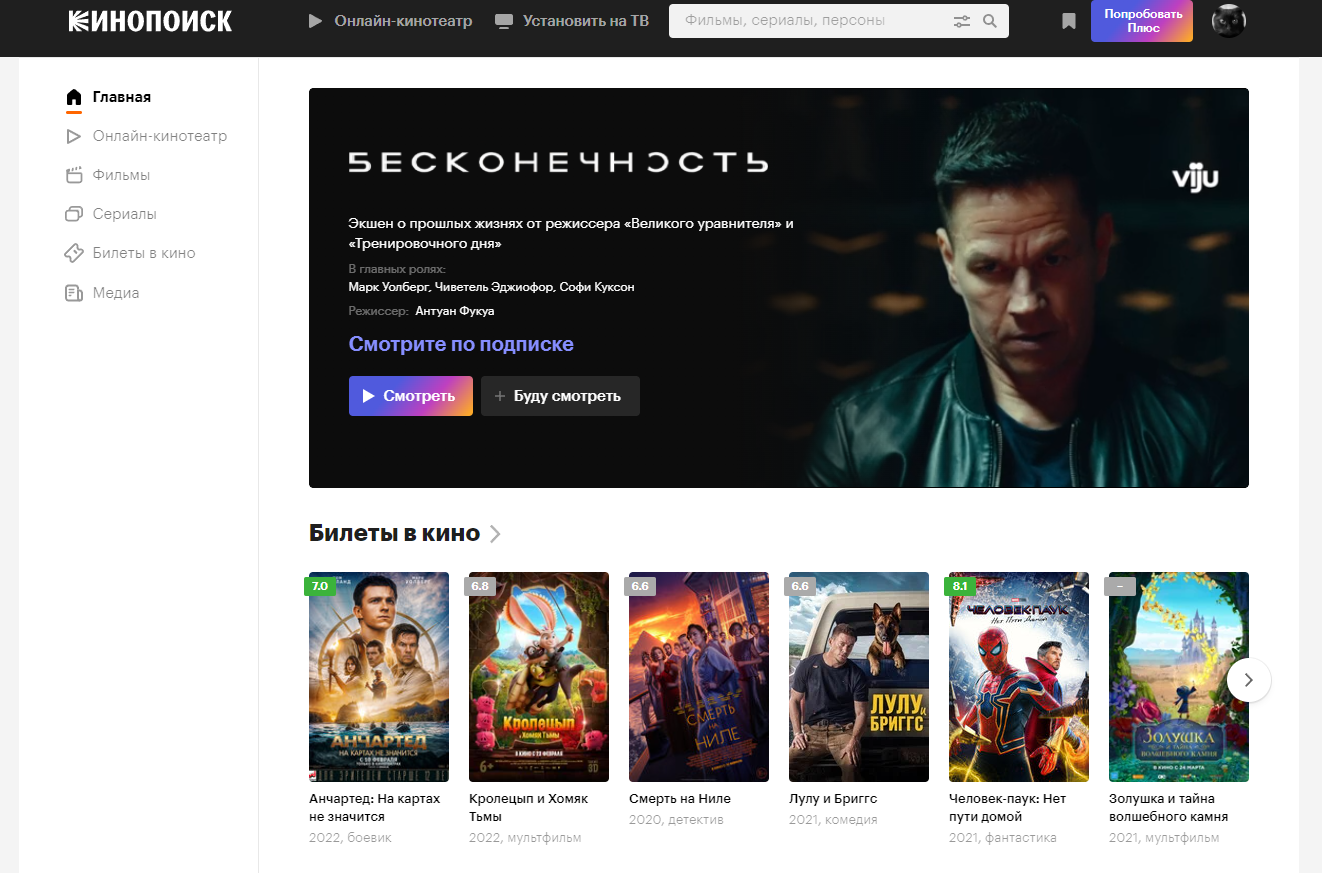


Рисунок 1.1 – Пример дизайна главной страницы

Однако некоторые станицы можно было красивее и интереснее оформить в этом можно убедится на рисунке 1.2. Помимо этого, на Кинопоиске можно не только посмотреть информацию, но и посмотреть нужный тебе фильм или сериал.

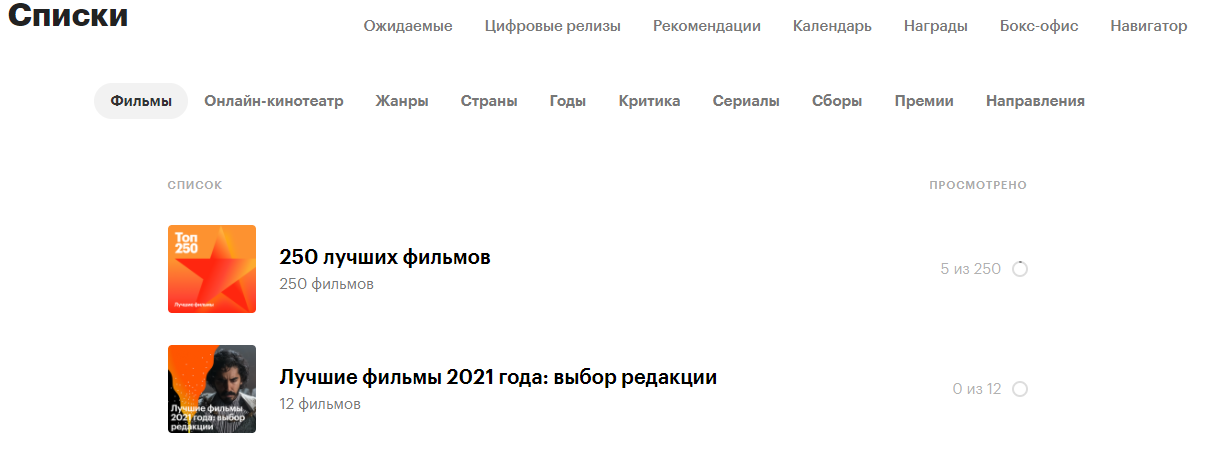


Рисунок 1.2 – Пример оформления страницы

Следующий сайт hdkinoteatr. Дизайн этого сайт, на мой взгляд, отвратителен он представлен на рисунке 1.3. Но с другой стороны, он сразу предоставляет всю нужную пользователю информацию. Но для этого нужно время, чтобы все найти.

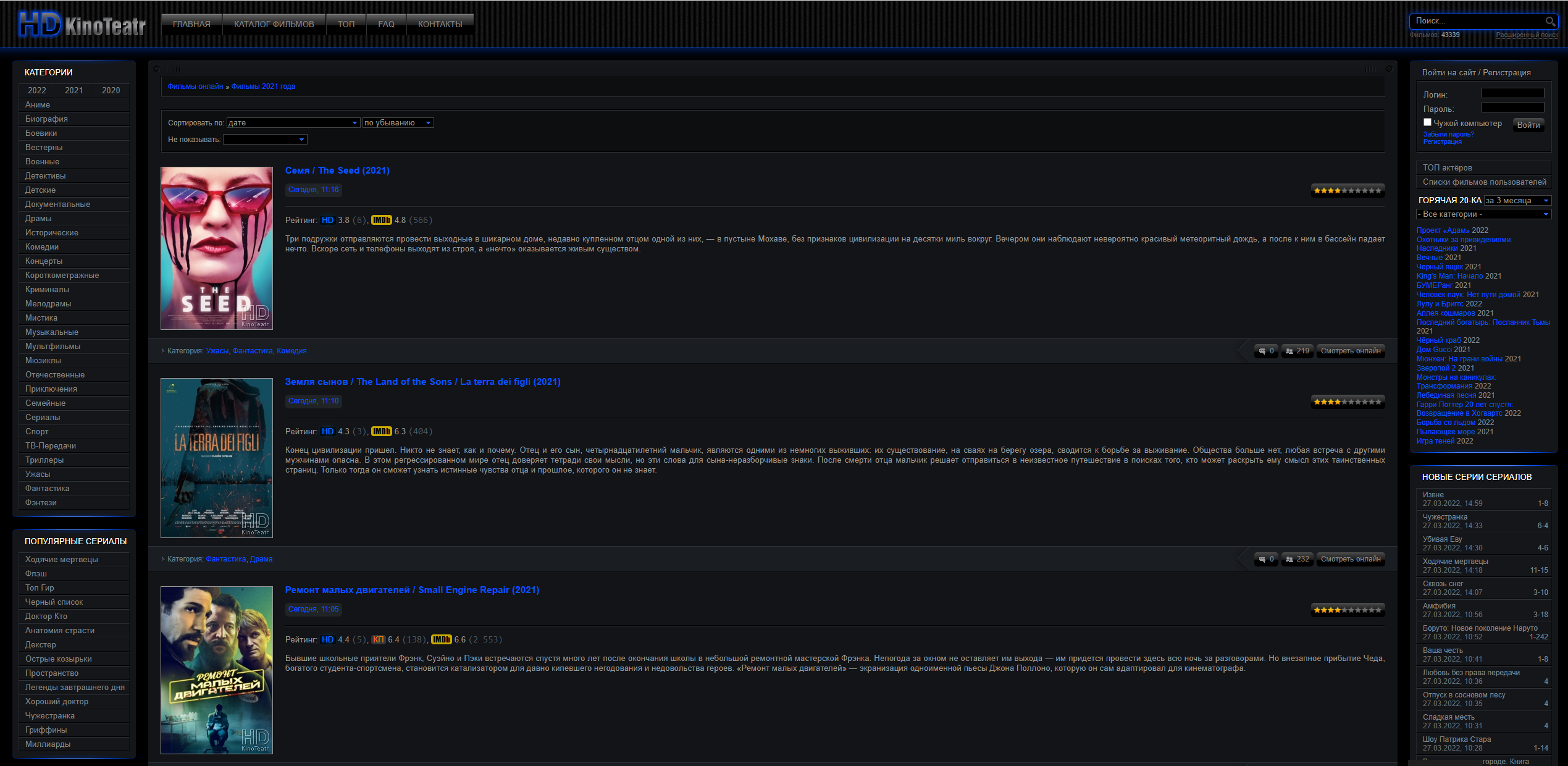


Рисунок 1.3 – Пример оформления главной страницы

Данный сайт, как и предыдущий, предоставляет возможность посмотреть фильм или сериал как показано на рисунке 1.4. В плане информации здесь куда хуже, чем на предыдущем сайте.

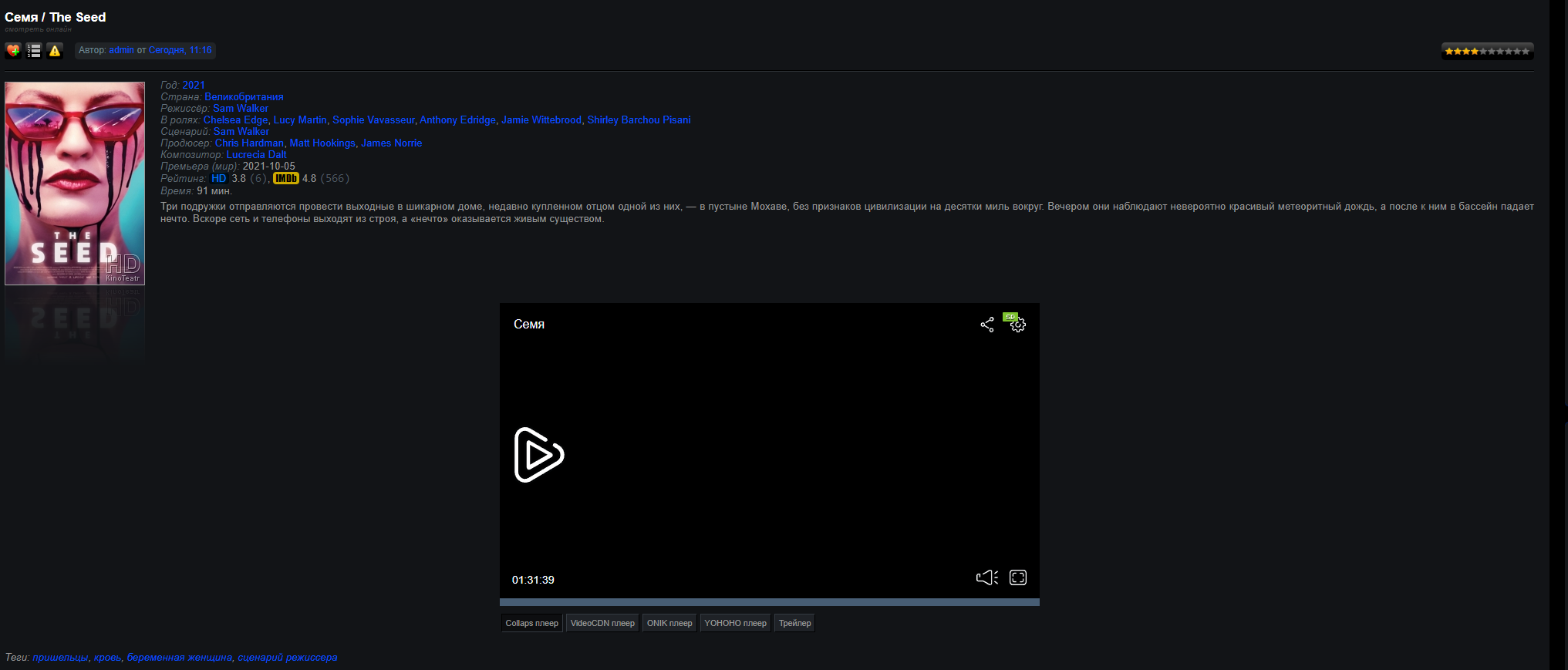


Рисунок 1.4 – Пример оформления страницы

Ну и наконец последний аналог myshows. Этот сайт создан уже исключительно для просмотра информации о сериалах. Данный сайт уже не предоставляет возможности просмотра сериалов. В этом можно убедиться на рисунке 1.5.

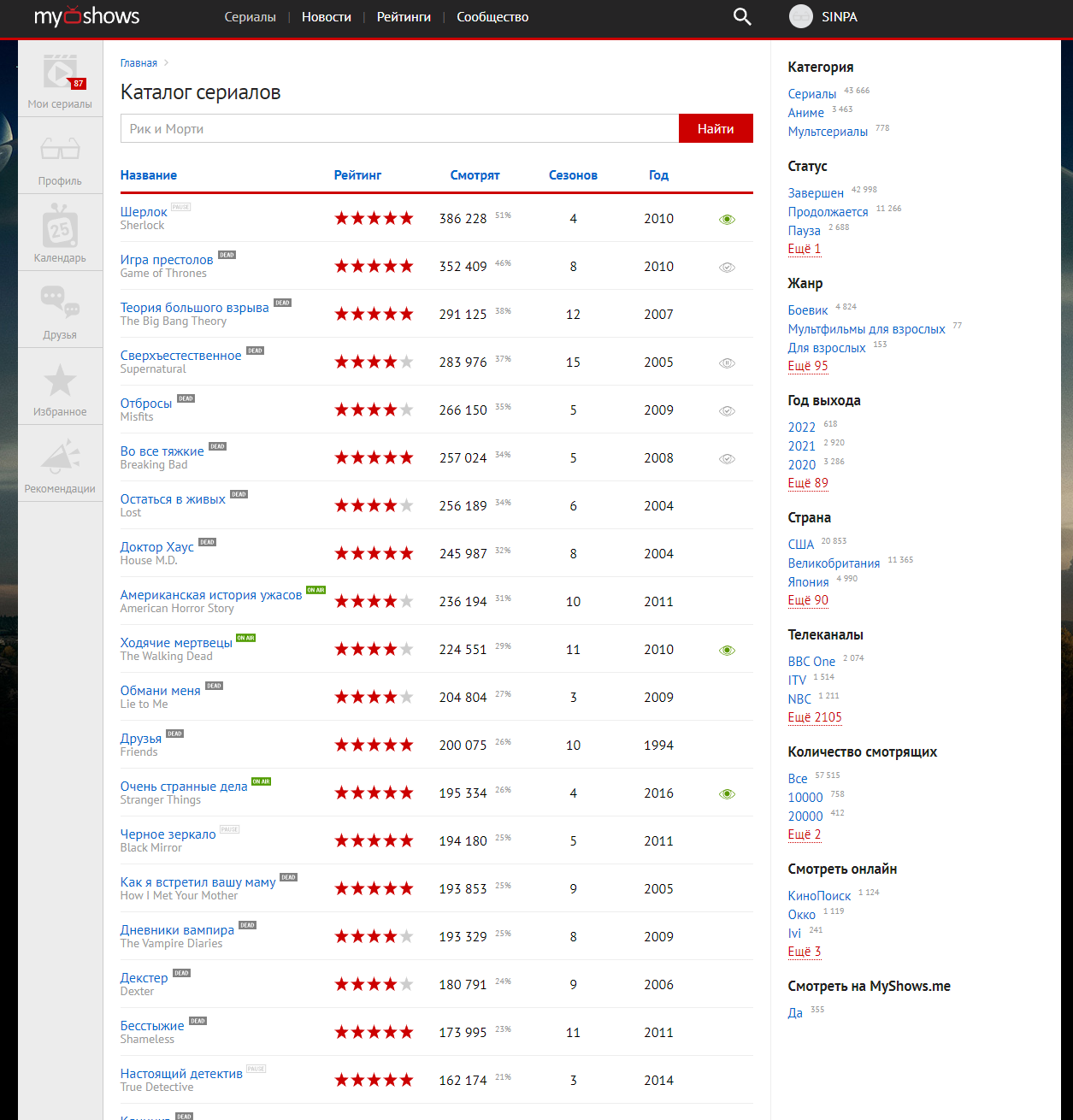


Рисунок 1.5 – Каталог сериалов

Но основная цель данного сайта – это отметить просмотренные тобой сериалы, серии, поставить им оценку и поделится мнением о них как показано на рисунках 1.6 и 1.7.

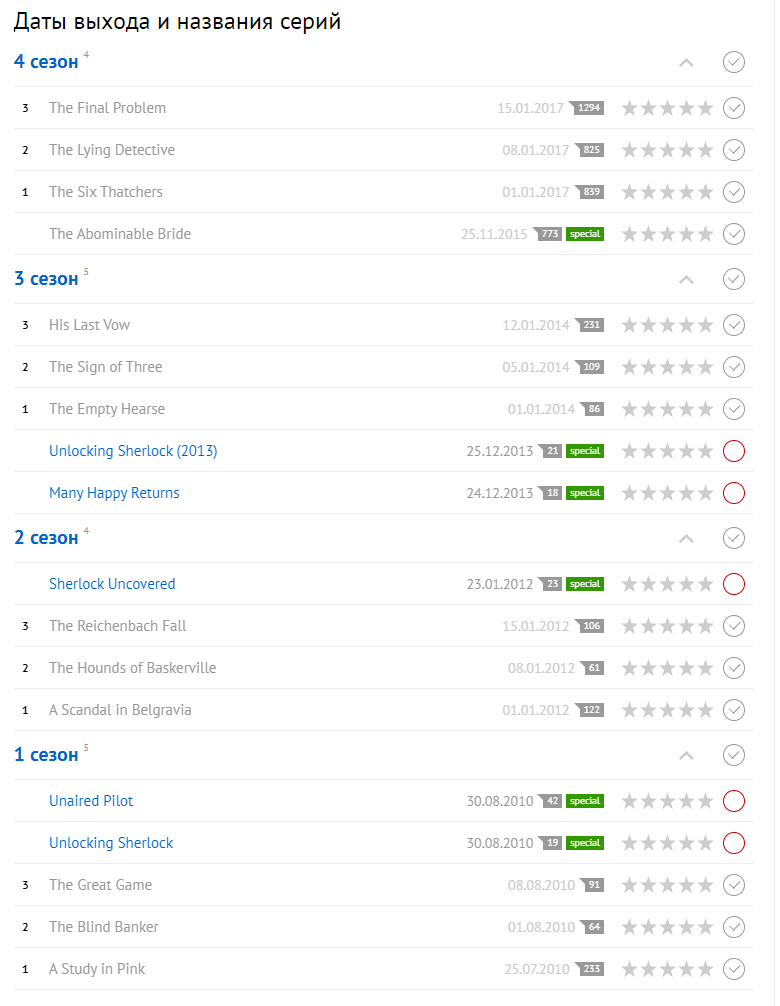


Рисунок 1.6 – Пример оформления страницы

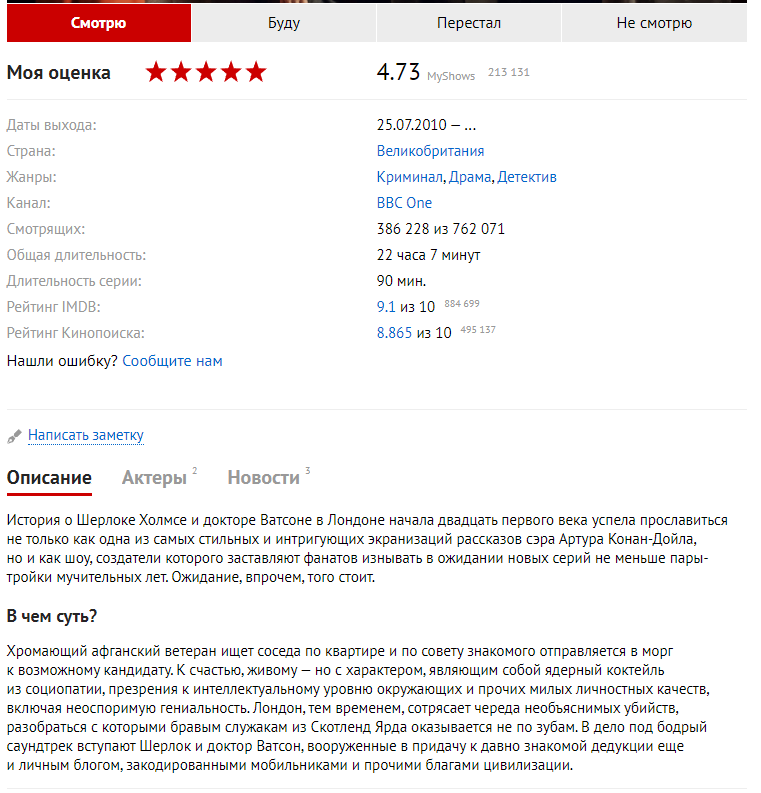


Рисунок 1.7 – Пример страницы с информацией о сериале

Собственно говоря, данный аналог мне очень понравился, более того я им уже пользуюсь большое количество времени и меня всё устраивает. Особенно ненавязчивый простенький дизайн, который для задачи (отметить просмотренный сериал или серию) отлично подходит.

# 1.2 Техническое задание

В данном проекте требуется создать многостраничный сайт, который будет предоставлять информацию о сериалах. Сайт должен содержать контактные данные, каталог сериалов, а также актуальные новости в мире кино. Основным контентом является информация о сериале: дата выхода, страна, жанры, канал, по которому транслируется, общая длительность, рейтинг IMBD и Кинопоиска, а также описание сериала, актеры и трейлер.

Соответственно главной задачей сайта будет снабжение пользователя всей необходимой и интересующей его информацией. Он будет предназначен как раз для людей, которые не знают, что посмотреть.

Сайт должен адаптироваться под мобильную версию.

Требуется реализовать главную страницу, на которой будет высвечен каталог сериалов (их постер и название), наиболее популярные сериалы, а также любопытная информация. Также будет возможность связаться с владельцем сайта через социальные сети.

Будут реализованы страницы, на которых можно подробно прочитать актуальные новости, связанные с тематикой сайта, с удобной возможностью выбрать следующую новость, прочитав её заглавие.

Остальные страницы уже будут содержать информацию о выбранном сериале. Здесь будет отображаться всё самое важное, ради чего пользователь и посетил наш сайт.

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, графические элементы в формате SVG.

# 1.3 Выбор средств реализации программного продукта

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 1.8. 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 средствадля [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).

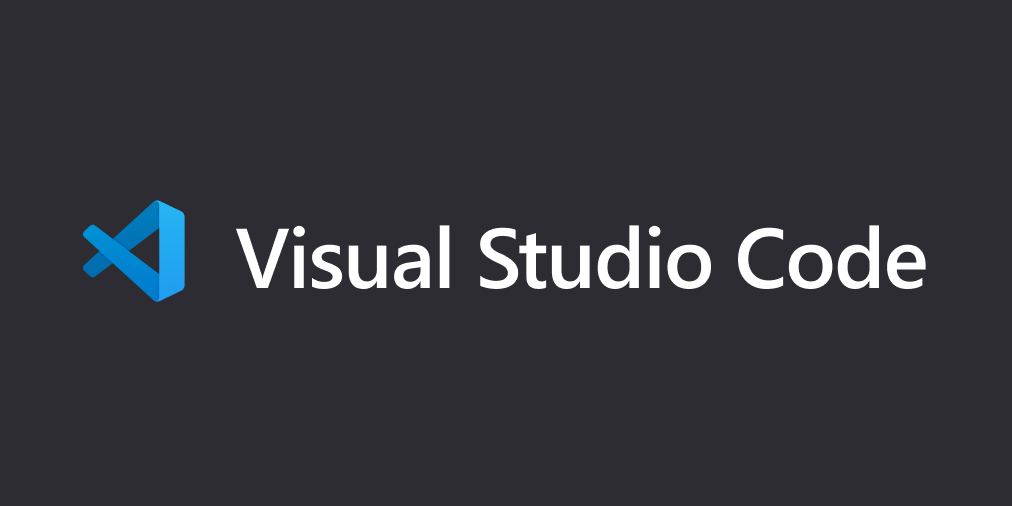


Рисунок 1.8 – Логотип Visual Studio Code

Структура сайта создана с помощью языка разметки HTML, дизайн сайта оформлен с помощью CSS с добавлением JavaScript, в частности с использованием библиотеки jQuery.

**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 и**спользуется, 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.

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.

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 регистрозависимыми. Операторы разделяются при помощи точки с запятой «;», однако использование этого символа необязательно: оператор автоматически завершается в конце строки.

jQuery— набор функций JavaScript, фокусирующийся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX. Разработка jQuery ведётся командой добровольцев на пожертвования.

# 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 Создание прототипа

Для того чтобы начать писать сайт, нужно примерно понять, как он будет выглядеть, для этого создается прототип сайта. По факту прототип – это начальные наброски того, как вы видите свой сайт. В макете же уже отображается конечный вид вашего сайта. Первоначально кончено создавалась главная страница, на которую пользователь перейдет после того как попадет на наш сайт. Поэтому хотелось ее наполнить самым интересным и актуальным контентом и заинтересовать пользователя остаться на нашем сайте. Но для того, чтобы его не отпугнуть было принято решения не загромождать главную страницу большим количеством текста, а сделать упор на постеры и картинки. Ознакомится с прототипом можно в приложении А.

# 2.2 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. 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+).

# 2.3 Выбор стилевого оформления

Сайт не загромождается лишними деталями (боковыми панелями и т.п.), большим количеством спецэффектов. Акцент на сайте ориентируется на основной контент и постеры сериалов.

Изображения можно разделить на две группы: маленькие изображения постеров (на главной странице) и большие изображения на страницах с информацией.

Цветовые решения выбраны по своему вкусу, для самого контента были выбраны фиолетовые и розовые цвета: #6E3667 и # C776BD. Шапка и подвал оформлены традиционно на черном цвете с белыми элементами.

# 2.4 Выбор шрифтового оформления

В макете использовался шрифт Work Sans, однако от него пришлось отказаться при версте сайта, так как он не поддерживает кириллицу. В данном проекте будут использованы один шрифт Montserrat и разные его начертания (он почти не отличается от Work Sans). Данный шрифт очень приятный на вид и красиво меняется при изменении начертания, что позволяет использовать его как для простого текста, так и для заглавных слов.

Текст же почти везде написан белым цветом, изредка меняясь на золотой и черный.

# 2.5 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 About (дословно переводится как “о”) и Series (“Сериалы”). Получается “О сериалах”. Что и отражает саму суть данного сайта. Эти два слова были написаны на эллипсе (так как эллипс черного цвета, то в шапке и подвале 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. Логотип представлен на рисунке 2.1.

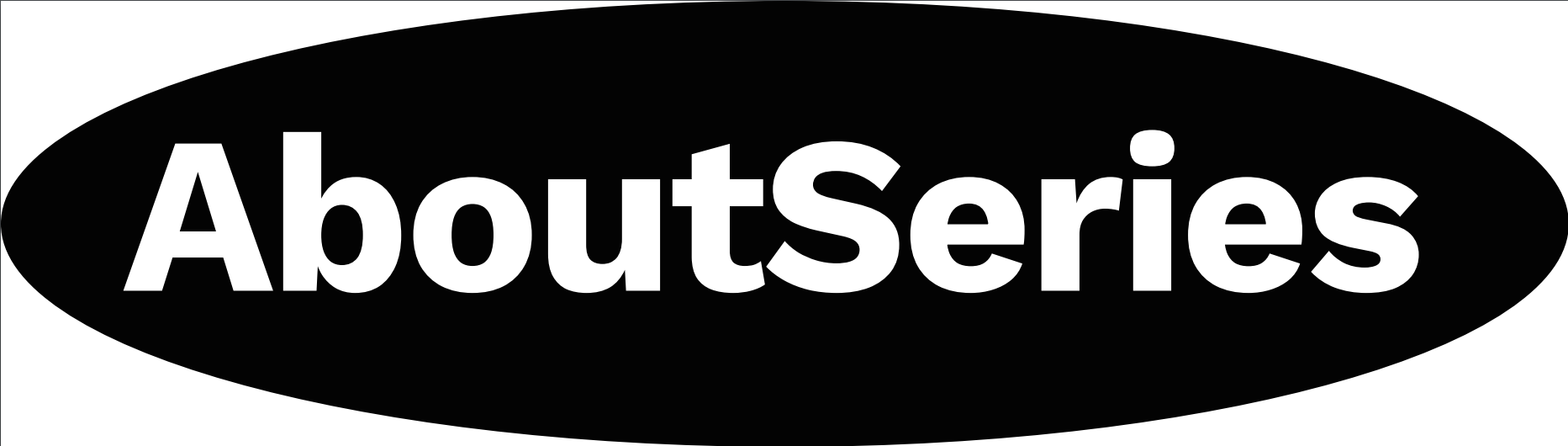


Рисунок 2.1 – Логотип сайта

# 2.6 Разработка пользовательских элементов

В проекте будут реализованы кликабельные слова в шапке и подвале они представлены на рисунках 2.2 и 2.3, при нажатии на которые, страница будет опускаться на определенный раздел, а при нажатии на логотип будет возвращаться в начало, т.е. в раздел популярное пример представлен на рисунке 2.4.

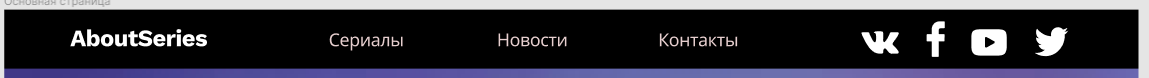


Рисунок 2.2 – Шапка сайта

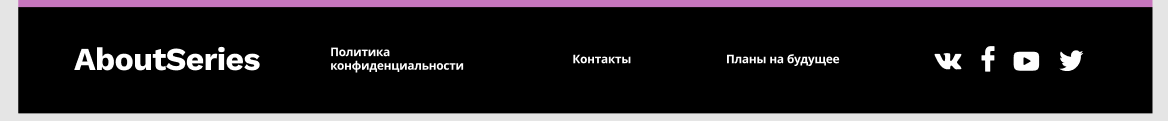


Рисунок 2.3 – Подвал сайта

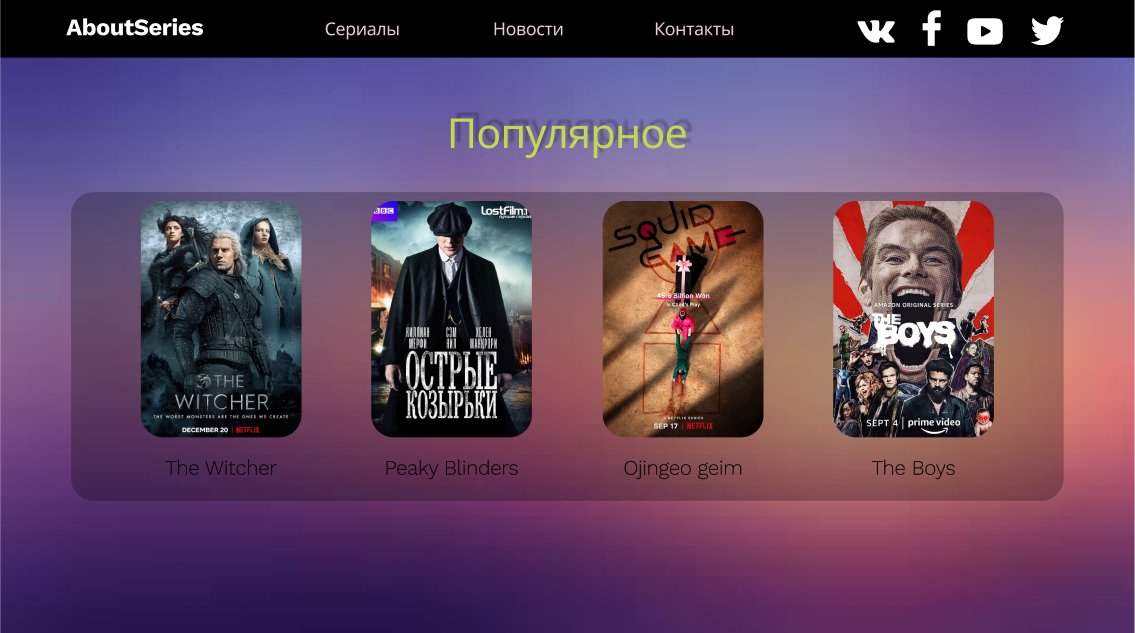


Рисунок 2.4 – Начало главной страницы

При нажатии на социальные сети, то, очевидно, пользователь будет отправлен на страницы в социальных сетях разработчика данного сайта.

Также на странице присутствуют кнопки «Смотреть больше» и «Читать больше». Они представлены на рисунках 2.5 и 2.6. Если пользователь нажмет на них, то откроются дополнительные скрытые постеры сериалов/новости.

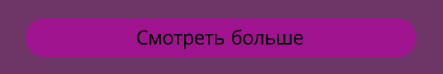


Рисунок 2.5 – Пример кнопки

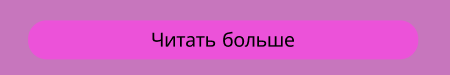


Рисунок 2.6 – Пример кнопки

На дополнительных страницах с информацией о сериалах также присутствуют шапка и подвал. Помимо этого, есть 3 кнопки: описание, актеры, трейлер. Они представлены на рисунках 2.7, 2.8 и 2.9. При посещении страницы всегда будет по стандарту стоять описание, и соответственно выводится описание серила. Но при нажатии на кнопку актеры, описание поменяется на маленькие изображения актеров, с их настоящими именами и сериальными. А при нажатии на кнопку трейлер, описание поменяется на трейлер, который можно будет посмотреть на данном сайте.

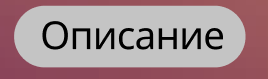


Рисунок 2.7

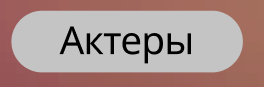


Рисунок 2.8

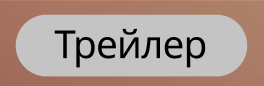


Рисунок 2.9

# 2.7 Разработка спецэффектов

На главной странице будет присутствовать анимация подчеркивания выбранного элемента меню при наведении курсора либо при фокусировке. А также анимация изменения цвета самого слова. То же касается и иконок социальных сетей. При наведении на них они будут менять цвет.

Также будет добавлена анимация пролистывания страницы и анимация активных кнопок сайта таких как: «смотреть больше» и «читать больше».

# 2.8 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 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>.

В теге <header> располагается шапка сайта. В ней находится логотип в формате SVG, иконки социальных сетей, также в формате SVG и тег <nav> для создания блоков навигации. Для работы с SVG иконками использовался тег <svg> и 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. Также в шапке располагается бюргер-меню, которое появляется при уменьшении разрешения. Код шапки сайта представлен в листинге 3.1 (для уменьшения кода внутренние теги были свернуты).

|  |
| --- |
| <header class="header">  <svg class="logo" width="280" height="80" viewBox="0 0 280 80" fill="none" xmlns="http://www.w3.org/2000/svg">  </svg>  <nav class="navigator">  </nav>  <svg class="media" width="350" height="50" viewBox="0 0 350 60" fill="none" xmlns="http://www.w3.org/2000/svg">  </svg>  </header> |

Листинг 3.1 – Шапка сайта

Основное тела сайта у меня располагается в теге <div>, так как я использовал блочную верстку. В тегах <div> огромное количества других тегов. Например, <figure> для вставки изображений и тег <figcaption> для создания подписи изображениям. Использовались также и теги <a> для создания ссылок на другие страницы. Конечно же использовались различные теги заголовков <h1>, <h2>, <h3>. В страницах, на которых использовалось множество маленьких элементов, которые необходимо было правильно расставить, было принято решение использовать тег <table>. Данный тег позволяет создавать удобные таблицы и, в моем случае, удобно привести элементы к нужному виду. В данном теге использовались такие теги как <tr> для создания новой строки таблицы и <td> для создания ячеек строки. Для вывода самого изображения на страницу использовался тег <img>, а для работы со сплошным текстом тег <p>. Код первого основного фрейма сайта представлен в листинге 3.2.

|  |
| --- |
| <div id="first" class="first">  <h1 id="gg">Популярное</h1>  <div class="container">  <figure class="poster">  <a style="text-decoration: none;" href="the\_witcher.html">  <img src="images/Ведьмак.jpg" width="272px" height="400px" alt="">  <figcaption>The Witcher</figcaption>  </a>  </figure>  <figure class="poster">  <a style="text-decoration: none;" href="peaky\_blinders.html">  <img src="images/Козы.jpg" width="272px" height="400px" alt="">  <figcaption>Peaky Blinders</figcaption>  </a>  </figure>  <figure class="poster">  <a style="text-decoration: none;" href="Kalmar.html">  <img src="images/Кальмар.webp" width="272px" height="400px" alt="">  <figcaption>Ojingeo geim</figcaption>  </a>  </figure>  <figure class="poster">  <a style="text-decoration: none;" href="The\_Boys.html">  <img src="images/Пацаны.webp" width="272px" height="400px" alt="">  <figcaption>The Boys</figcaption>  </a>  </figure>  </div>  </div> |

Листинг 3.2 – Первый основной фрейм главной страницы

Заканчивается сайт обязательно тегом <footer>. Его называют подвалом сайта по понятным причинам, так как им заканчивается страница сайта. Здесь также располагается логотип сайта, иконки в формате SVG и навигационные элементы. Код подвала сайта представлен в листинге 3.3 (для уменьшения кода внутренние теги были свернуты).

|  |
| --- |
| <footer class="footer">  <svg class="logo" width="280" height="80" viewBox="0 0 280 80" fill="none" xmlns="http://www.w3.org/2000/svg">  </svg>  <nav class="navigator">  </nav>  <svg class="media" width="350" height="50" viewBox="0 0 350 60" fill="none" xmlns="http://www.w3.org/2000/svg">  </svg>  </footer> |

Листинг 3.3 – Подвал сайта

Весь свернутый код главной страницы представлен в листинге 3.4.

|  |
| --- |
| <div class="main\_page">  <header class="header">  </header>  <div id="first" class="first">  </div>  <div id="second" class="second">  </div>  <div id="third" class="third">  </div>  <footer class="footer">  </footer>  </div> |

Листинг 3.4 – Структура главной страницы

# 3.2 Добавление таблиц стилей 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 (Листинг 3.5).

|  |
| --- |
| <link rel=''stylesheet'' href=''style.css'' /> |

Листинг 3.5 – Пример подключения CSS

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. Подобных свойств очень много, а современные версии CSS позволяют даже работать с анимацией. 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.

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. (Листинг 3.6). 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.

|  |
| --- |
| @font-face {  font-family: Montserrat;  src: url(fonts/Montserrat-Regular.ttf);  }  body {  font-family: Montserrat;  } |

Листинг 3.6 – Пример использования @font-face

Иногда требуется установить одновременно один стиль для всех элементов веб-страницы, например, задать шрифт или начертание текста. В этом случае поможет универсальный селектор (обозначается как \*), который соответствует любому элементу веб-страницы. (Листинг 3.7). Свойства margin и padding определяют внешние и внутренние отступы соответственно. Значение 0 означает, что отступов не будет, тем самым мы сможем избежать непонятных отступов, которые нам не нужны, а если они нам пригодятся, то мы сами отдельно их пропишем для нужного нам элемента. Согласно спецификации CSS, ширина блока складывается из ширины содержимого (width), значений margin, padding и border. Аналогично обстоит и с высотой блока. Свойство box-sizing позволяет изменить этот алгоритм, чтобы свойства width и height задавали размеры не содержимого, а размеры блока. CSS-свойство scroll-behavior определяет поведение прокрутки для любого элемента на странице. Тем самым мы можешь сделать приятную прокрутку. Например, при использовании кнопок навигации, страницы будет красиво пролистываться до указанных нами элементов.

|  |
| --- |
| \* {  margin: 0;  padding: 0;  box-sizing: border-box;  scroll-behavior: smooth;  } |

Листинг 3.7 – Пример использования универсального селектора

Стоит отметить, что необязательно прописывать весь CSS код в одном файле. Также CSS можно подключать и внутри самого HTML кода через тег <style>. Однако это будет сильно загромождать код, хоть иногда это и удобно для написания каких-то особенных свойств. В моем коде иногда использовалось внутренне подключение из собственных соображений. Также стоит сказать, что весь мой CSS код лежит в одном файле. На мой взгляд, это намного упрощается нахождение и корректировку определенных свойств, так как вы можете использовать встроенный в vs code функционал такой как поиск. Узнать же имя класса, идентификатора или же тега может помочь встроенные во все современные браузеры инструменты разработчика. Помимо этого, для ориентирования по своим CSS свойствам я использовал комментарии, которые разделяли главные блоки моего сайта.

# 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.

# 3.4 Использование jQuery

jQuery — это быстрая, небольшая и многофункциональная библиотека JavaScript. Он значительно упрощает такие вещи, как обход и манипулирование документами HTML, обработку событий, анимацию и Ajax, с помощью простого в использовании API, который работает во множестве браузеров. Благодаря сочетанию универсальности и расширяемости jQuery изменил способ написания JavaScript миллионами людей. Пример подключения jQuery представлен в листинге 3.8. Первая строчка листинга предоставляет нам возможность использования самой библиотеки jQuery. А вторая строчка кода — это файл с расширением js, в котором и будет записан весь основой код.

|  |
| --- |
| <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>  <script src="first\_page.js"></script> |

Листинг 3.8 – Пример подключения jQuery

В своем коде jQuery я использовал для создания функционала мои кнопкам. Например, при нажатии на определенный тег, будут изменяться CSS свойства таким образом, что будет появляться дополнительная скрытые блоки. В CSS это делается через свойство display: none – это свойство временно удаляет элемент из документа и display: block/inline-block – это свойство отображает наш элемент и делает из него нужный нам блок. В jQuery для этого есть методы hide() – удаляет элемент и show() – отображает элемент. Значение внутри этих методов будет определять скорость выполнения. Пример использования данных методов представлен в листинге 3.9. Страницей нельзя безопасно манипулировать, пока документ не будет «готов». jQuery определяет это состояние готовности за вас. Код, включенный внутрь $( document ).ready(), будет выполняться только после того, как объектная модель документа (DOM) страницы будет готова для выполнения кода JavaScript.

|  |
| --- |
| $(document).ready(function() {  $(".buttom").click(() => {  $(".buttom").hide(500);  $(".buttom\_return").show(500);  $(".second\_container\_add").show(300);  });  });  $(document).ready(function() {  $(".buttom\_return").click(() => {  $(".buttom\_return").hide();  $(".buttom").show(500);  $(".second\_container\_add").hide(300);  });  }); |

Листинг 3.9 – Пример использования методов hide() и show()

Как и в случае с CSS я разделял код в js комментариями, чтобы определять к чему относится функционал, который я написал.

# 3.5 Вывод

На данном этапе была реализована структура на HTML, а также созданы таблицы стилей CSS. Было продемонстрировано использование стандартов XML (SVG) и использование библиотеки JavaScript (jQuery), а также веб-сайт был подготовлен к следующему этапу – к тестированию.

# 4. Тестирование веб-сайта

# 4.1 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.

Передо мной стояла задача: создать сайт, который будет сам подстраиваться под разные типы экранов. Главными условиями адаптивности является не фиксированный размер элементов страницы.

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).

Я использовал 3 разрешения после которых применялись новые свойства к элементам: 1280 px, 720 px и 480 px. 1280 px и более — это разрешение для персональных компьютеров. 720 px — разрешение для планшетов. 480 px — разрешение для мобильных устройств. Пример использования CSS3-медиазапросов представлен в приложении Б.

# 4.2 Кроссбраузерность веб-сайта

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.

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 почти во всех браузерах сайт работает отлично. Однако из-за использования jQuery в некоторых браузерах он попросту не работает. Браузеры, которые поддерживают jQuery: Chrome, Edge, Firefox, Internet Explorer: 9+, Safari и Opera.

# 4.3 Руководство пользователя

На главной странице пользователь может сразу ознакомиться с навигационными кнопками в шапке сайта, которые ему помогут в ориентации по сайту. Они представлены на рисунках 4.1 и 4.2.



Рисунок 4.1 – Навигационная кнопка



Рисунок 4.2 – Навигационная кнопка

Пользователь может нажать на любой постер сериала, либо же его название и тогда он окажется на другой странице, где ему предоставят всю необходимую информацию об этом сериале. На той же странице он сможет увидеть три кнопки: описание, актеры и трейлер, при нажатии на которые он сможет ознакомится с дополнительной информацией. Они представлены на рисунке 4.2.

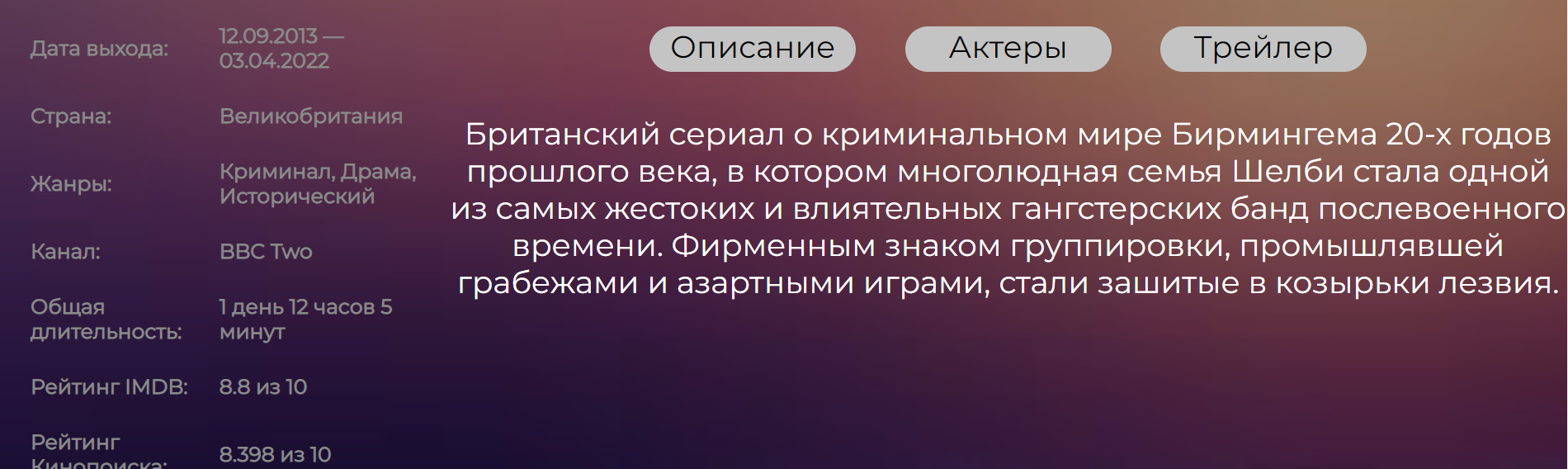


Рисунок 4.3 – Пример кнопок

Если пользователь захочет вернуться на главную страницу, он, конечно, может использовать функционал своего браузера, но также можется воспользоваться функционалом данного сайта. По нажатию на навигационные кнопки он сможет вернуться к интересующему ему разделу на главной странице.

Помимо всего прочего пользователь может ознакомиться с новостями, связанные с тематикой сайта. Также на главной странице можно нажать на кнопку подробнее, которая располагается на интересующей его новости. Тогда он перейдет на другую страницу, где сможет прочитать подробную информацию о новости, на которую он нажал. Также он может выбрать следующую новость, которая его заинтересует, при этом не возвращаясь на главную страницу, ему просто достаточно будет снова нажать на интересующую его новость. Пример можно увидеть на рисунке 4.4.

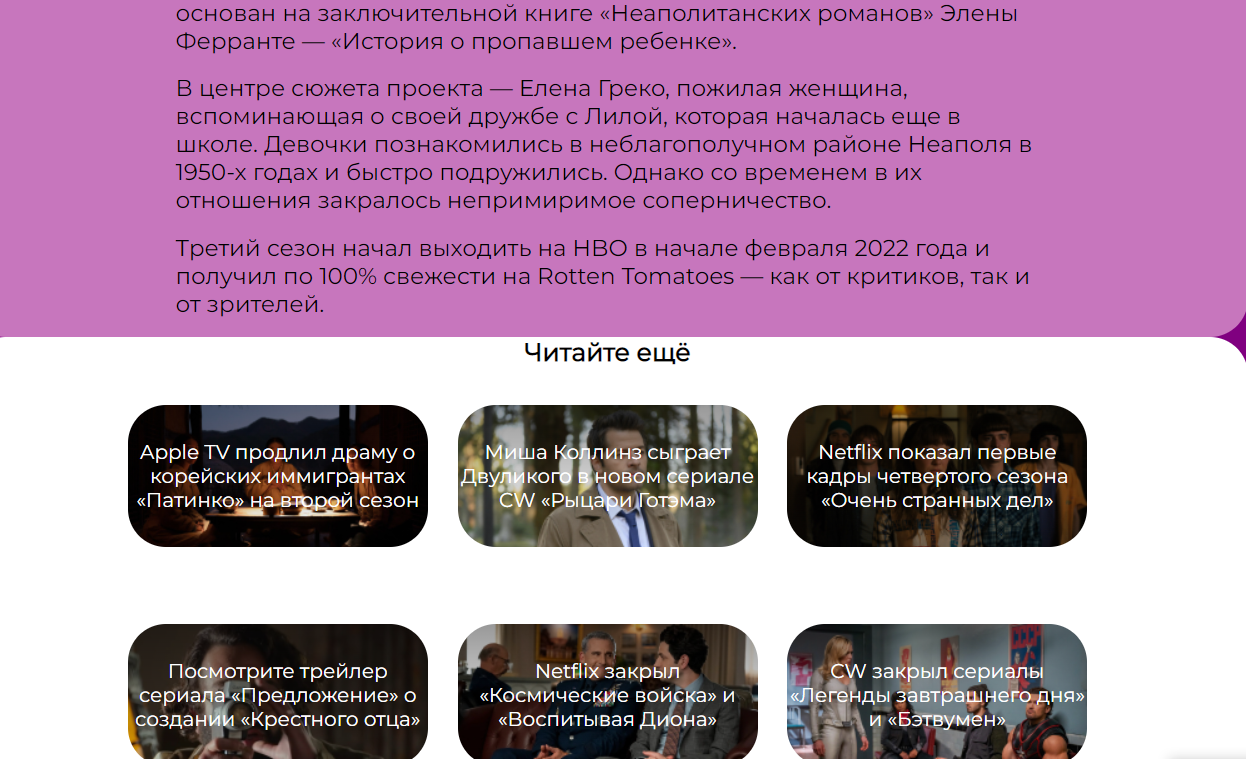


Рисунок 4.4

Также пользователь может смутить, что помимо описанных выше кнопок есть еще кнопки, которые расположены под постерами сериалов и новостей. Но ему будет достаточно просто прочитать, что написано на данных кнопках, чтобы понять их функционал. Такие кнопки на сайте две: «смотреть больше» и «читать больше». При нажатии на которые, на той же главной странице пользователь сможет ознакомится с дополнительными сериалами или же новостями. Кнопки представлены на рисунках 4.5 и 4.6.

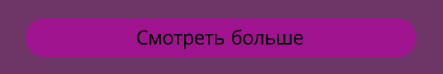


Рисунок 4.5 – Пример кнопки

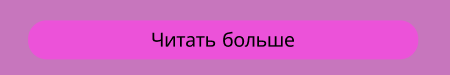


Рисунок 4.6 – Пример кнопки

# 4.4 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 литературы

- Документация по JavaScrip/HTML [Электронный ресурс] / Справочный Режим доступа : <https://developer.mozilla.org>

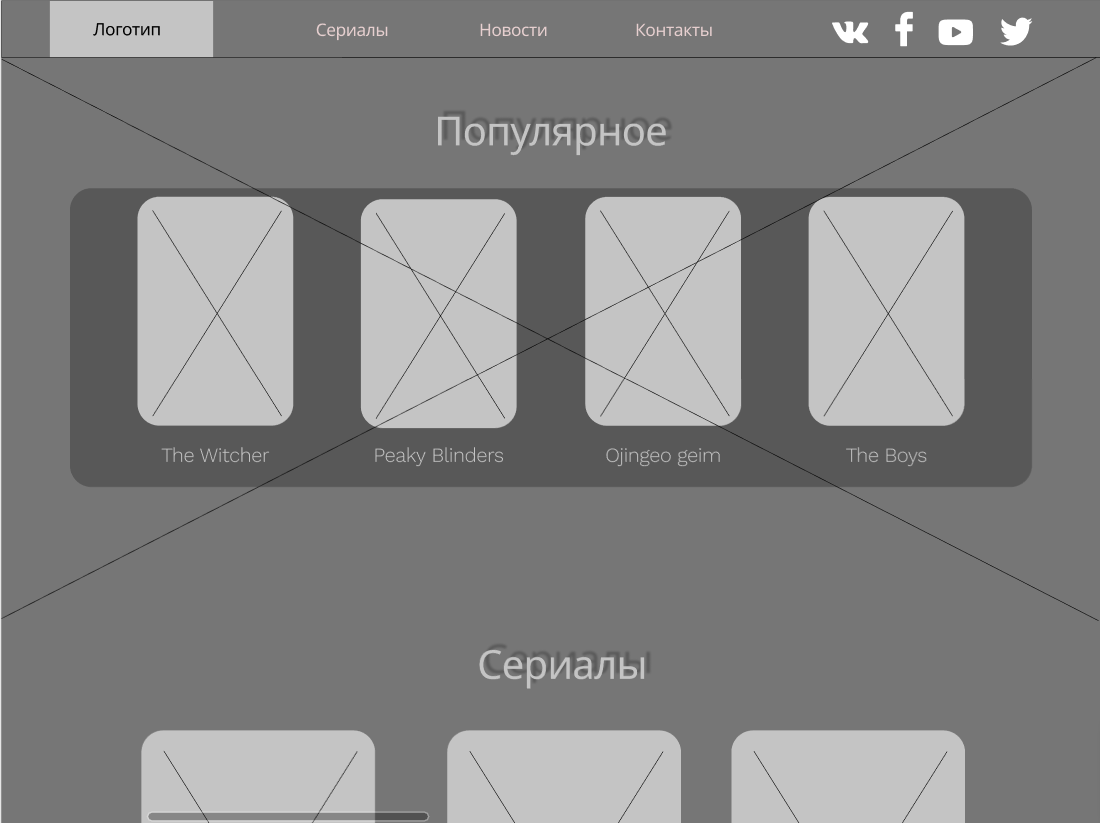
- Документация по Figma [Электронный ресурс] / Справочный Режим доступа : <https://help.figma.com>

- Документация по HTML [Электронный ресурс] / Справочный режим доступа : <https://developer.mozilla.org/ru/docs/Web/HTML>

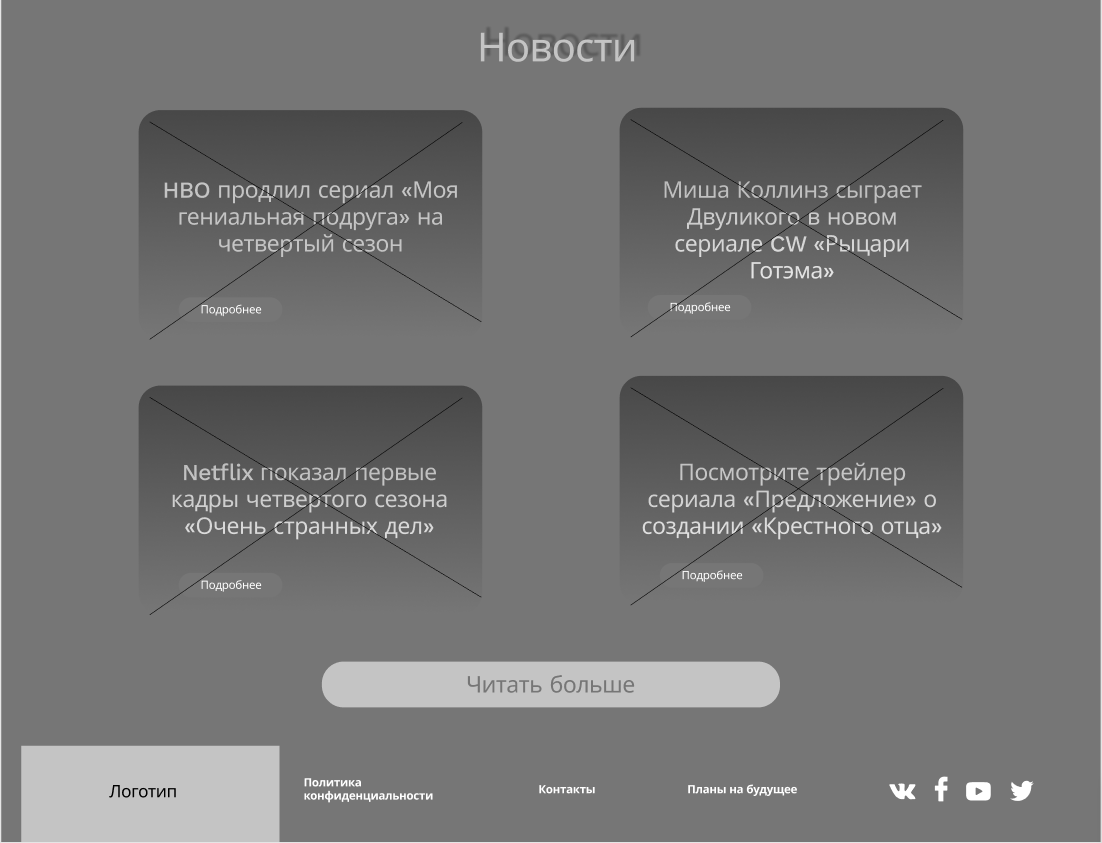
- Документация по HTML [Электронный ресурс] / Справочный режим доступа : <https://html5book.ru/html-html5/>

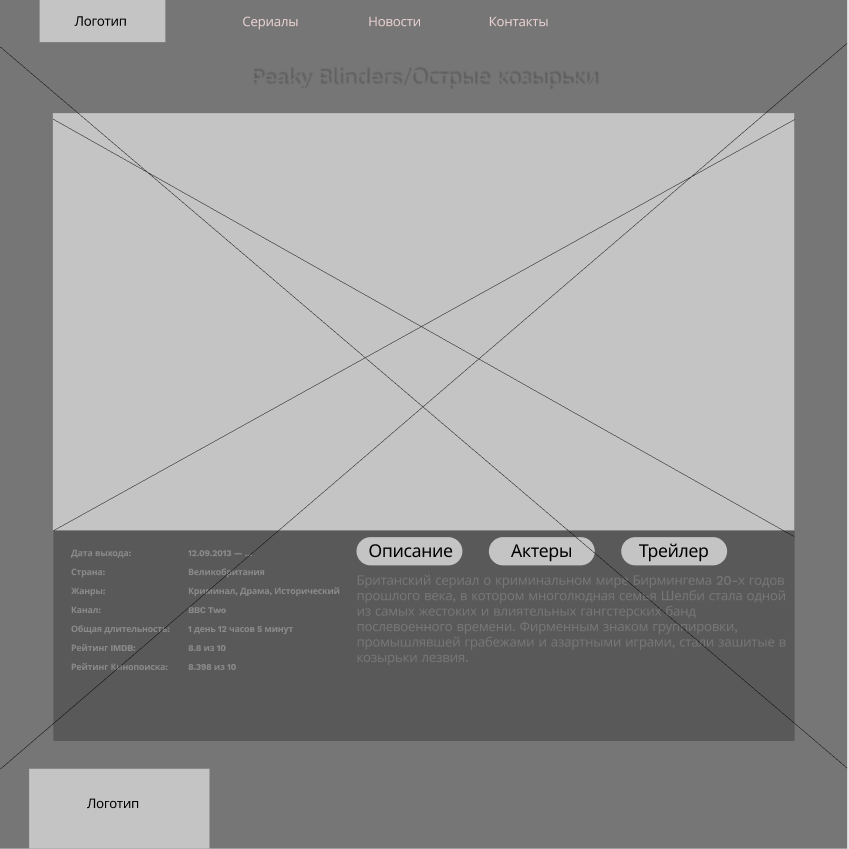
# Приложение А

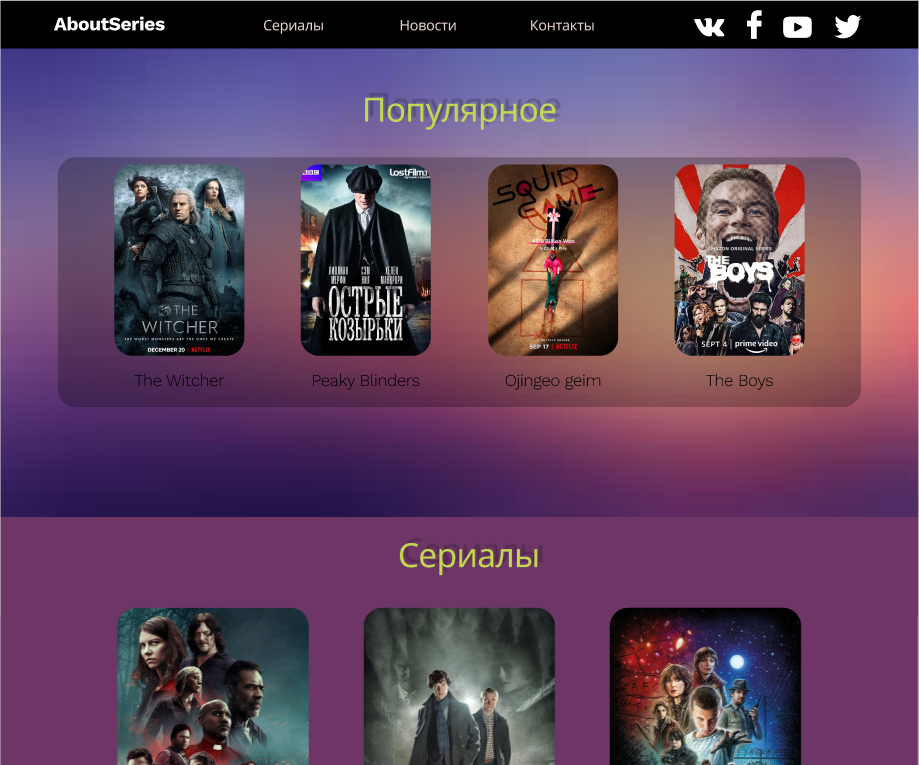
Прототипы страниц и макет

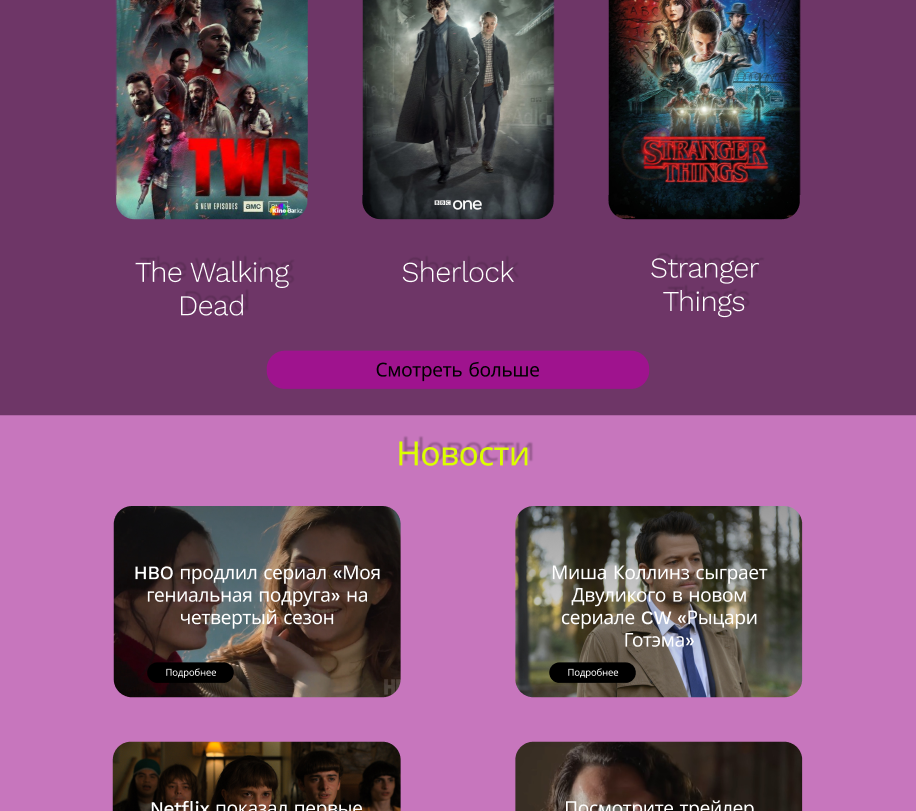
















# Приложение Б

CSS

|  |
| --- |
| @media (max-width:1280px) {  .header .nav {  font-size: 15px;  margin: 0 15px;  }  .header .logo {  margin: 0;  width: 187px;  height: 54px;  }  .header .media {  width: 235px;  height: 34px;  margin: 0;  }  } |

Пример CSS3-медиазапроса при разрешении 1280 px

|  |
| --- |
| @media (max-width:720px) {  .header .nav {  font-size: 12px;  margin: 10px;  }  .header .logo {  width: 92px;  height: 26px;  margin: 0;  }  .header .media {  width: 116px;  height: 17px;  margin: 0;  }  } |

Пример CSS3-медиазапроса при разрешении 720 px

|  |
| --- |
| @media (max-width:480px) {  .header .nav {  font-size: 8px;  margin: 0;  }  .header .logo {  width: 92px;  height: 26px;  margin: 0;  }  .header .media {  width: 116px;  height: 17px;  margin: 0;  }  } |

Пример CSS3-медиазапроса при разрешении 480 px

|  |
| --- |
| @font-face {  font-family: Montserrat;  src: url(fonts/Montserrat-Regular.ttf);  }  body {  font-family: Montserrat;  }  \* {  margin: 0;  padding: 0;  box-sizing: border-box;  scroll-behavior: smooth;  }  .main\_page {  width: 100%;  max-width: 2000px;  margin-left: auto;  margin-right: auto;  }  /\* Header \*/  .header {  display: flex;  background-color: black;  width: 100%;  justify-content: space-between;  align-items: center;  }  .logo {  margin-left: 20px;  }  .navigator {  font-size: 30px;  font-weight: 400;  }  .nav {  color: #E7D0D0;  text-decoration: none;  transition: color 0.2s linear;  transition: fill 0.1s linear;  margin: 0 30px;  position: relative;  }  .nav:hover {  color: #C4D751;  fill: #C4D751;  }  .nav:hover:after {  opacity: 1;  }  .nav:after {  content: "";  opacity: 0;  width: 100%;  height: 3px;  transition: opacity 0.2s linear;  background-color: #C4D751;  position: absolute;  top: 100%;  left: 0;  }  .media {  margin-right: 20px;  }  /\* First \*/  .first {  background: url(images/Без\ названия.jpg) center no-repeat;  background-size: cover;  text-align: center;  width: 100%;  height: 50;  padding-bottom: 2%;  }  .first h1 {  padding-top: 2%;  font-weight: 500;  font-size: 70px;  color: #C4D751;  text-shadow: 10px -10px 4px rgba(0, 0, 0, 0.25);  }  .container {  max-width: 1680px;  margin-left: auto;  margin-right: auto;  background: url(images/Без\ названия.jpg) center no-repeat;  background-size: cover;  border-radius: 38px;  background: rgba(0, 0, 0, .25);  margin-top: 2%;  }  .poster {  display: inline-block;  margin: 2% 3%;  }  .poster img {  border-radius: 38px;  }  .poster figcaption {  margin-top: 1%;  font-weight: 300;  font-size: 35px;  color: #fff;  }  /\* Second \*/  .second {  background-color: #6E3667;  text-align: center;  width: 100%;  padding-bottom: 2%;  }  .second h1 {  padding-top: 2%;  font-weight: 500;  font-size: 70px;  color: #C4D751;  text-shadow: 10px -10px 4px rgba(0, 0, 0, 0.25);  }  .second\_container {  margin-left: auto;  margin-right: auto;  width: 100%;  margin-top: 2%;  }  .second .poster {  display: inline-block;  margin: 0 1%;  }  .second figcaption {  margin-top: 2%;  font-weight: 300;  font-size: 40px;  color: #FFF;  text-shadow: 10px -10px 4px rgba(0, 0, 0, 0.25);  }  .buttom {  cursor: pointer;  margin-left: auto;  margin-right: auto;  margin-top: 4%;  max-width: 500px;  background-color: #9F138E;  border-radius: 38px;  text-align: center;  font-weight: 400;  font-size: 40px;  }  .buttom\_return {  cursor: pointer;  display: none;  margin-left: auto;  margin-right: auto;  margin-top: 4%;  max-width: 500px;  background-color: #9F138E;  border-radius: 38px;  text-align: center;  font-weight: 400;  font-size: 40px;  }  .second\_container\_add {  display: none;  margin-left: auto;  margin-right: auto;  width: 100%;  margin-top: 4%;  }  /\* Third \*/  .third {  background-color: #C776BD;  text-align: center;  width: 100%;  padding-bottom: 2%;  }  .third h1 {  padding-top: 2%;  font-weight: 500;  font-size: 70px;  color: #C4D751;  text-shadow: 10px -10px 4px rgba(0, 0, 0, 0.25);  }  .third\_container {  margin-left: auto;  margin-right: auto;  width: 100%;  margin-top: 2%;  }  .news {  margin: 0 1%;  margin-bottom: 1%;  display: inline-block;  background-size: cover;  max-width: 600px;  max-height: 400px;  border-radius: 38px;  text-align: center;  font-weight: 400;  font-size: 40px;  padding-top: 8%;  padding-bottom: 8%;  color: #FFF;  }  .buttom\_news {  color: #E7D0D0;  transition: color 0.2s linear;  display: block;  text-decoration: none;  font-weight: 300;  font-size: 20px;  margin-top: 8%;  background-color: #000000;  max-width: 180px;  padding: 1% 0 1% 0;  border-radius: 38px;  margin-left: 10%;  position: relative;  }  .buttom\_news:hover {  color: #C4D751;  }  .buttom\_news:hover:after {  opacity: 1;  }  .buttom\_news:after {  content: "";  opacity: 0;  width: 66%;  height: 5%;  transition: opacity 0.2s linear;  background-color: #C4D751;  position: absolute;  top: 85%;  left: 17%;  }  .buttom\_read {  cursor: pointer;  margin-left: auto;  margin-right: auto;  max-width: 500px;  background-color: #EC52D9;  border-radius: 38px;  text-align: center;  font-weight: 400;  font-size: 40px;  margin-top: 4%;  }  .buttom\_read\_return {  cursor: pointer;  display: none;  margin-left: auto;  margin-right: auto;  max-width: 500px;  background-color: #EC52D9;  border-radius: 38px;  text-align: center;  font-weight: 400;  font-size: 40px;  margin-top: 4%;  }  .third\_container\_add {  display: none;  margin-left: auto;  margin-right: auto;  width: 100%;  }  /\* Footer \*/  .footer {  background-color: #000000;  width: 100%;  display: flex;  justify-content: space-between;  align-items: center;  }  .footer .nav {  font-size: 20px;  }  /\* News Page \*/  .new1 {  background-color: purple;  }  .main\_content {  border-radius: 38px;  margin-left: auto;  margin-right: auto;  max-width: 1280px;  background-color: #C776BD;  text-align: center;  font-size: 20px;  }  .text {  margin-left: auto;  margin-right: auto;  background-color: #C776BD;  max-width: 960px;  border-radius: 38px;  }  .main\_content h2 {  font-weight: 500;  color: #C4D751;  padding-top: 2%;  padding-bottom: 2%;  }  .main\_content p {  margin-left: 5%;  margin-right: 5%;  font-weight: lighter;  padding-bottom: 2%;  text-align: left;  font-size: 23px;  }  .also {  text-align: center;  border-radius: 38px;  margin-left: auto;  margin-right: auto;  max-width: 1280px;  background-color: #FFF;  }  .also h3 {  text-align: center;  font-size: 25px;  }  .news\_also {  text-decoration: none;  text-align: center;  margin: 3% 1%;  color: #FFF;  padding-top: 35px;  padding-bottom: 35px;  display: inline-block;  background-size: cover;  max-width: 300px;  max-height: 200px;  border-radius: 38px;  position: relative;  font-size: 20px;  }  .news\_also:hover {  color: #C4D751;  }  .news\_also:hover:after {  opacity: 1;  }  .news\_also:after {  content: "";  opacity: 0;  width: 90%;  height: 3px;  transition: opacity 0.2s linear;  background-color: #C4D751;  position: absolute;  top: 75%;  left: 5%;  }  /\* Description \*/  .main {  background: url(images/Без\ названия.jpg) center no-repeat;  background-size: cover;  width: 100%;  text-align: center;  }  .main h1 {  font-weight: 400;  font-size: 50px;  color: #FFF;  padding-top: 2%;  }  .pic {  display: inline-block;  max-width: 1680px;  padding-top: 2%;  margin-left: auto;  margin-right: auto;  }  .des {  display: flex;  justify-content: space-between;  text-align: left;  margin-left: auto;  margin-right: auto;  margin-top: 2%;  max-width: 1680px;  background-size: cover;  background: rgba(0, 0, 0, .35);  }  .qwe table {  display: inline-block;  max-width: 610px;  font-weight: 700;  font-size: 20px;  text-align: left;  color: #8A8A8A;  }  .knop {  display: inline-block;  width: 100%;  }  .knop nav {  margin-top: 1%;  margin-right: auto;  }  .ewq {  text-align: center;  }  .kn {  cursor: pointer;  color: #000000;  text-align: center;  padding-top: 2px;  border-radius: 38px;  margin: 2% 2%;  width: 200px;  height: 44px;  font-weight: 400;  font-size: 30px;  background-color: #C4C4C4;  display: inline-block;  }  .tex {  margin-top: 20px;  color: #FFF;  font-weight: 500;  font-size: 30px;  text-align: center;  }  .act {  text-align: center;  display: none;  }  .avatar {  width: 100px;  height: 100px;  border-radius: 100px;  }  .tact {  display: inline-block;  color: #fff;  font-weight: 400;  font-size: 20px;  border-spacing: 30px;  }  .tact\_for\_adapt {  font-weight: 400;  font-size: 20px;  border-spacing: 30px;  color: #fff;  display: none;  }  .chname {  color: #C4C4C4;  font-size: 15px;  }  .video {  margin-left: auto;  margin-right: auto;  display: none;  } |

# Приложение В

|  |
| --- |
| function initialize() {  /\* First\_buttom \*/  $(document).ready(function() {  $(".buttom").click(() => {  $(".buttom").hide(500);  $(".buttom\_return").show(500);  $(".second\_container\_add").show(300);  });  });  $(document).ready(function() {  $(".buttom\_return").click(() => {  $(".buttom\_return").hide();  $(".buttom").show(500);  $(".second\_container\_add").hide(300);  });  });  /\* Second\_buttom \*/  $(document).ready(function() {  $(".buttom\_read").click(() => {  $(".buttom\_read").hide(500);  $(".buttom\_read\_return").show(500);  $(".third\_container\_add").show(300);  });  });  $(document).ready(function() {  $(".buttom\_read\_return").click(() => {  $(".buttom\_read\_return").hide(500);  $(".buttom\_read").show(500);  $(".third\_container\_add").hide(300);  });  });  $(document).ready(function() {  $("#future").click(() => {  alert("Никаких:)")  });  });  $(document).ready(function() {  $("#des").click(() => {  $(".tex").show();  $(".act").hide();  $(".video").hide();  });  $("#actors").click(() => {  $(".tex").hide();  $(".act").show();  $(".video").hide();  });  $("#trailer").click(() => {  $(".tex").hide();  $(".act").hide();  $(".video").show();  });  });  }  $(initialize); |

Листинг jQuery