Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра Системное программирование

Отчёт по лабораторной работе

по дисциплине «UI/UX»

Выполнила студентка группы БВТ2001:

Епифанова Елизавета

Проверил:

Москва 2023

Оглавление

[Введение 3](#_Toc154784917)

[Цель 4](#_Toc154784918)

[Задания 4](#_Toc154784919)

[Выполнение работы 5](#_Toc154784920)

[Заключение 14](#_Toc154784921)

# **Введение**

На JavaScript выполняется большая часть интерактивных элементов на сайтах и в мобильных приложениях. JavaScript отлично работает с HTML/CSS и интегрирован основные браузеры на рынке. Чистый JavaScript используется в вебе, а для общего применения JavaScript разработчики используют различные среды выполнения, например, Node.js.

Node.js — это среда выполнения кода JavaScript. Она позволяет использовать JavaScript как язык программирования общего назначения: создавать на нем серверную часть и писать полноценные десктопные приложения.

Основа Node.js — движок V8. Этот движок был разработан Google и используется в браузере Google Chrome. Он компилирует код JavaScript в машинный код, который понимает процессор. Однако, чтобы сделать из JavaScript язык общего назначения, одного движка недостаточно. Так, например, для создания серверной части нужно, чтобы язык умел работать с файлами, сетью и т.п. Для решения этой проблемы разработчики добавили к V8 дополнительные возможности, с помощью своего кода и сторонних библиотек.

В итоге у них получился инструмент, который превращает JavaScript в язык общего назначения.

Node.js стала популярна среди разработчиков благодаря возможности

создавать серверную и клиентскую часть на одном языке, скорости работы и NPM.

# **Цель**

Необходимо разработать SPA исходя из описания задачи.

# **Задания**

Имеется API со списком фильмов, с которым нужно наладить взаимодействие. Необходимо визуализировать данные полученные от API в виде таблицы, содержащей только важную (на ваш взгляд) информацию о фильме.

К каждому фильму из таблицы можно оставить комментарий (неограниченное количество), а также удалить его в любой момент времени, локальное состояние.

Методы, с которыми необходимо работать:

• List Movies;

• Retrieve Movie.

Требования к задаче:

• NextJS;

• Отсутствие перезагрузки страницы;

• В приложении должна присутствовать пагинация.

Правила хорошего тона:

• Адаптивная верстка;

• Приятный и удобный UI;

• Простой и лаконичный код;

• DRY.

**Выполнение работы**

Данная лабораторная работа включает в себя следующие части.

1. Стартовая страница, находящаяся по ссылке <http://localhost:3000/>. При нажатии на кнопку «start now» осуществляется переход на страницу со списком фильмов. Страница представлена на рисунке 1.

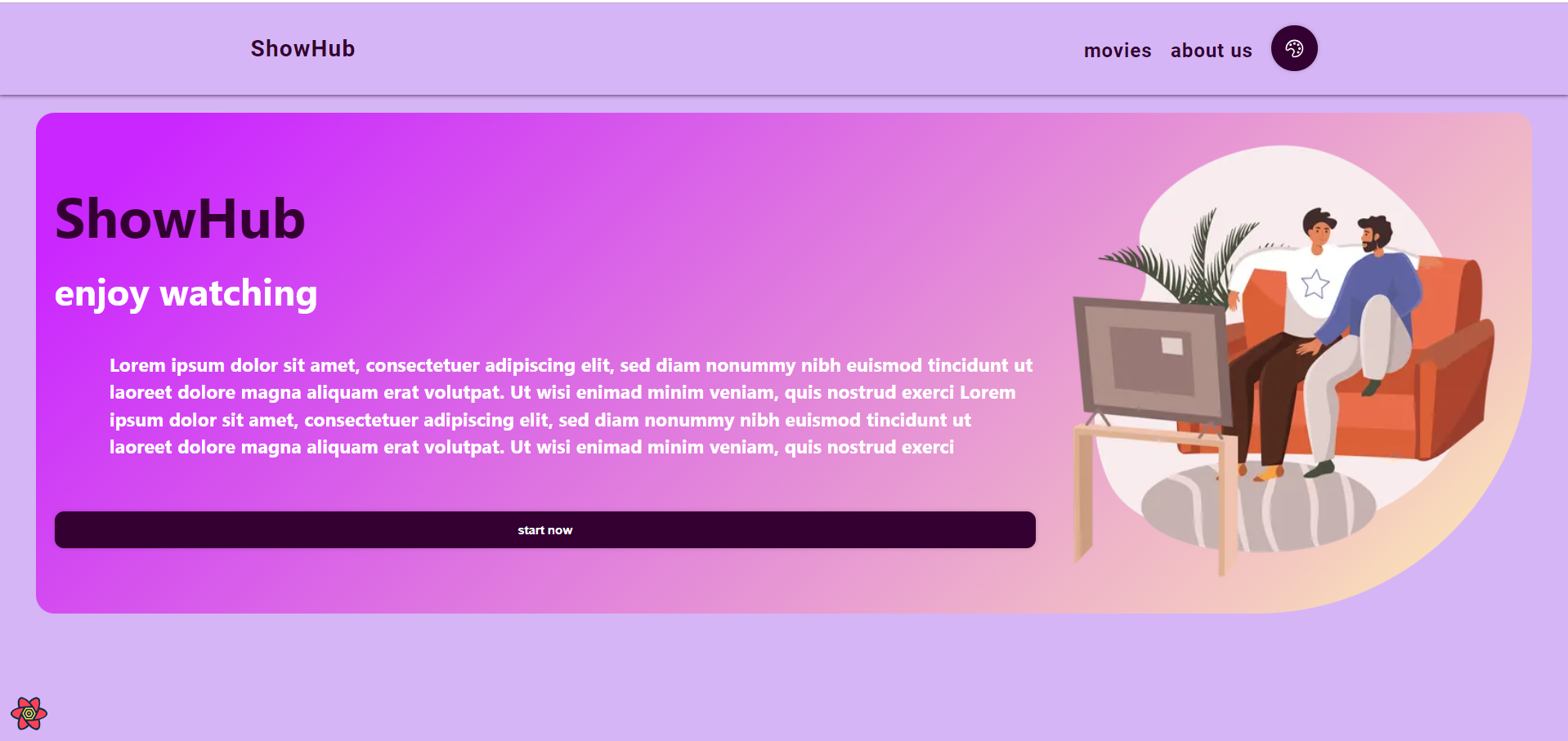


Рисунок 1 – Стартовая страница

1. Страница со список фильмов, находящаяся по ссылке <http://localhost:3000/movies>. Данная страница предоставляет возможность перейти на карточку просмотра конкретного фильма. Для этого необходимо нажать на кнопку, содержащую название фильма. Страница представлена на рисунке 2.

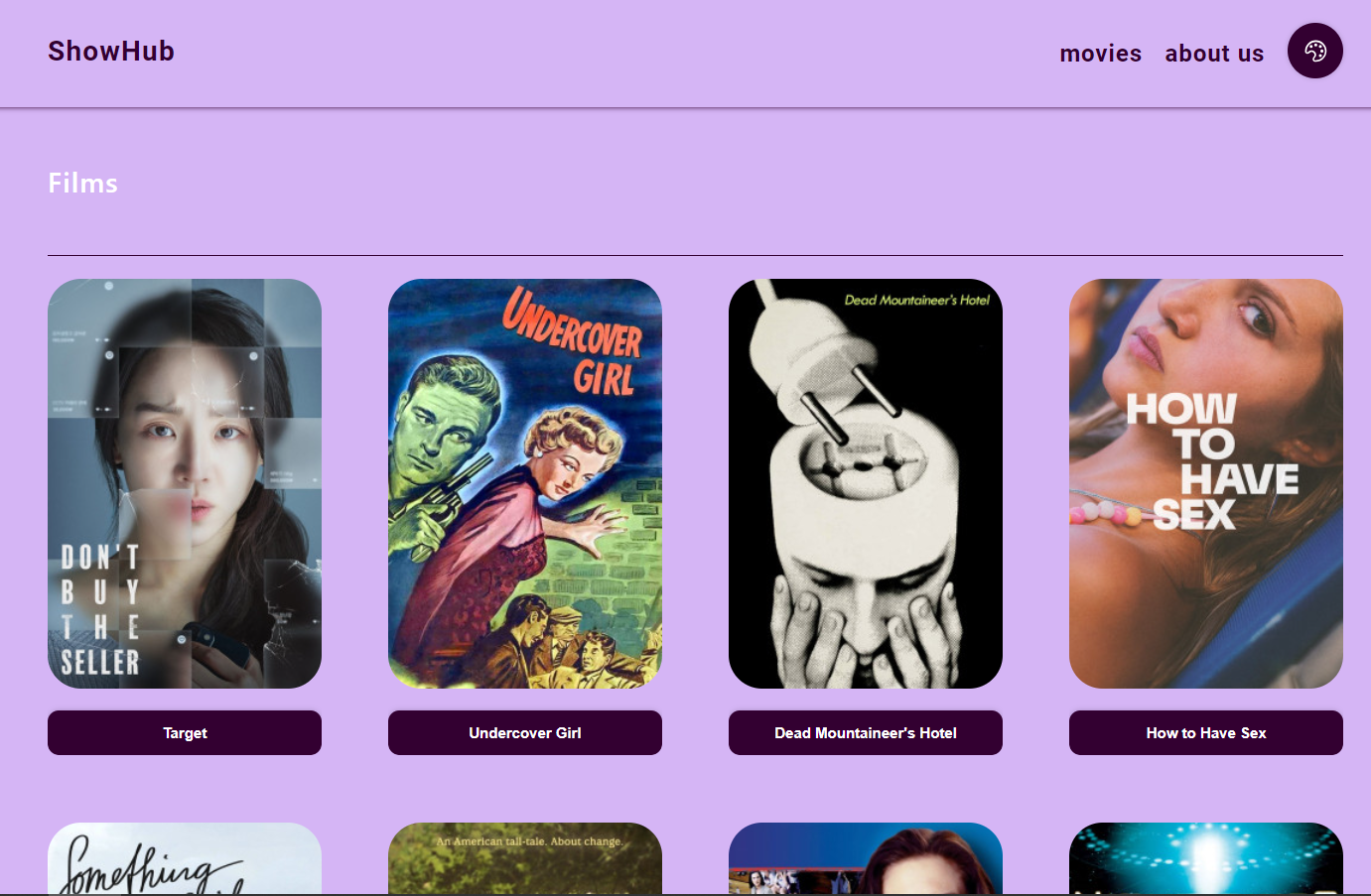


Рисунок 2 – Страница со список фильмов

1. Страница с подробной информацией о фильме, находящаяся п ссылке <http://localhost:3000/movies/58814>, где 58814 – id конкретного фильма. Данная страница предоставляет возможность ознакомится с информацией о фильме и оставить/удалить комментарий к данному фильму. Страница представлена на рисунке 3.

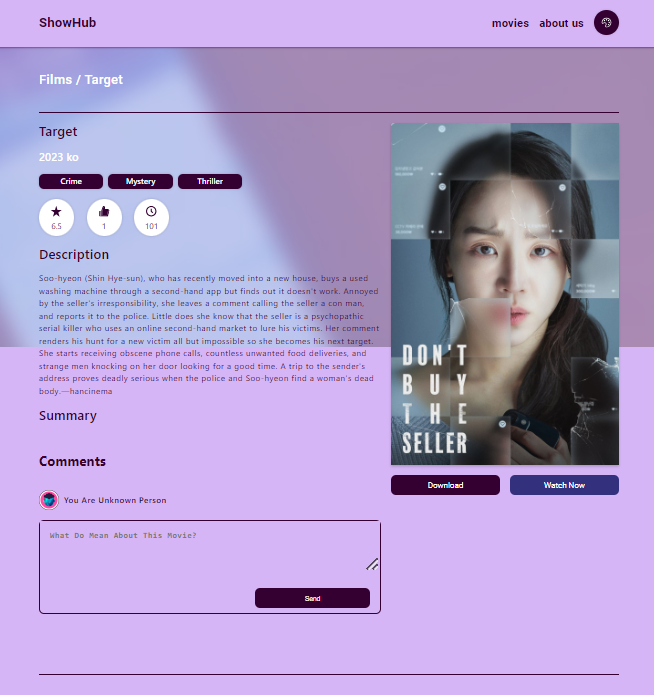


Рисунок 3 – Страница с подробной информацией о фильме

В ходе лабораторной работы была продумана адаптивность дизайна. На рисунках 4 и 5 соответственно представлена часть реализации адаптивности на странице со списком фильмов и подробной информацией о фильме.

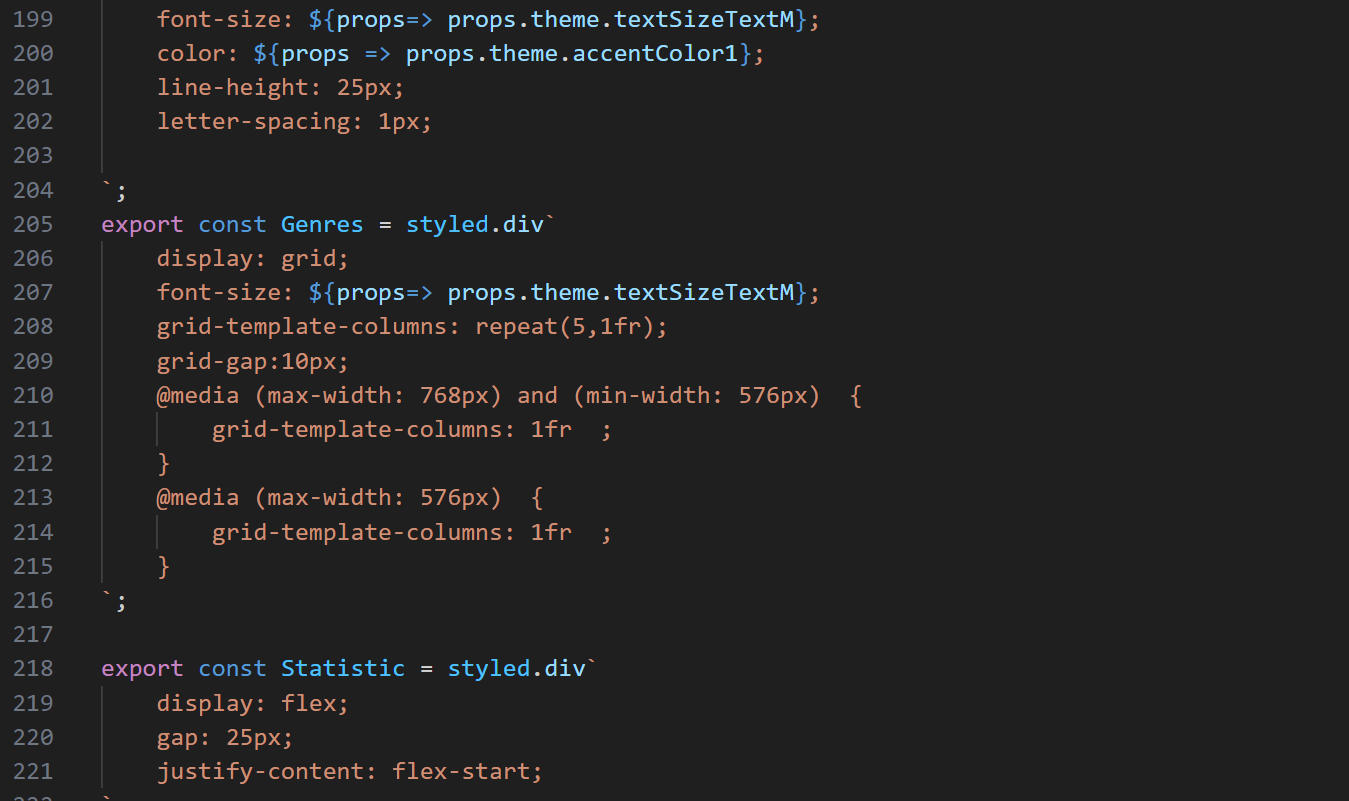


Рисунок 4 – Часть реализации адаптивности

Этот код создает стилизованный компонент Genres с использованием сетки, задает размер шрифта, шаблон колонок и применяет различные стили в зависимости от ширины экрана.

@media (max-width: 768px) and (min-width: 576px) { ... } - это медиа-запрос, который применяет стили внутри блока только если ширина экрана находится в пределах от 576px до 768px. В данном случае, он переопределяет grid-template-columns для одной колонки.

@media (max-width: 576px) { ... } - это еще один медиа-запрос, который применяет стили внутри блока только если ширина экрана меньше или равна 576px. Здесь также переопределяется grid-template-columns для одной колонки.

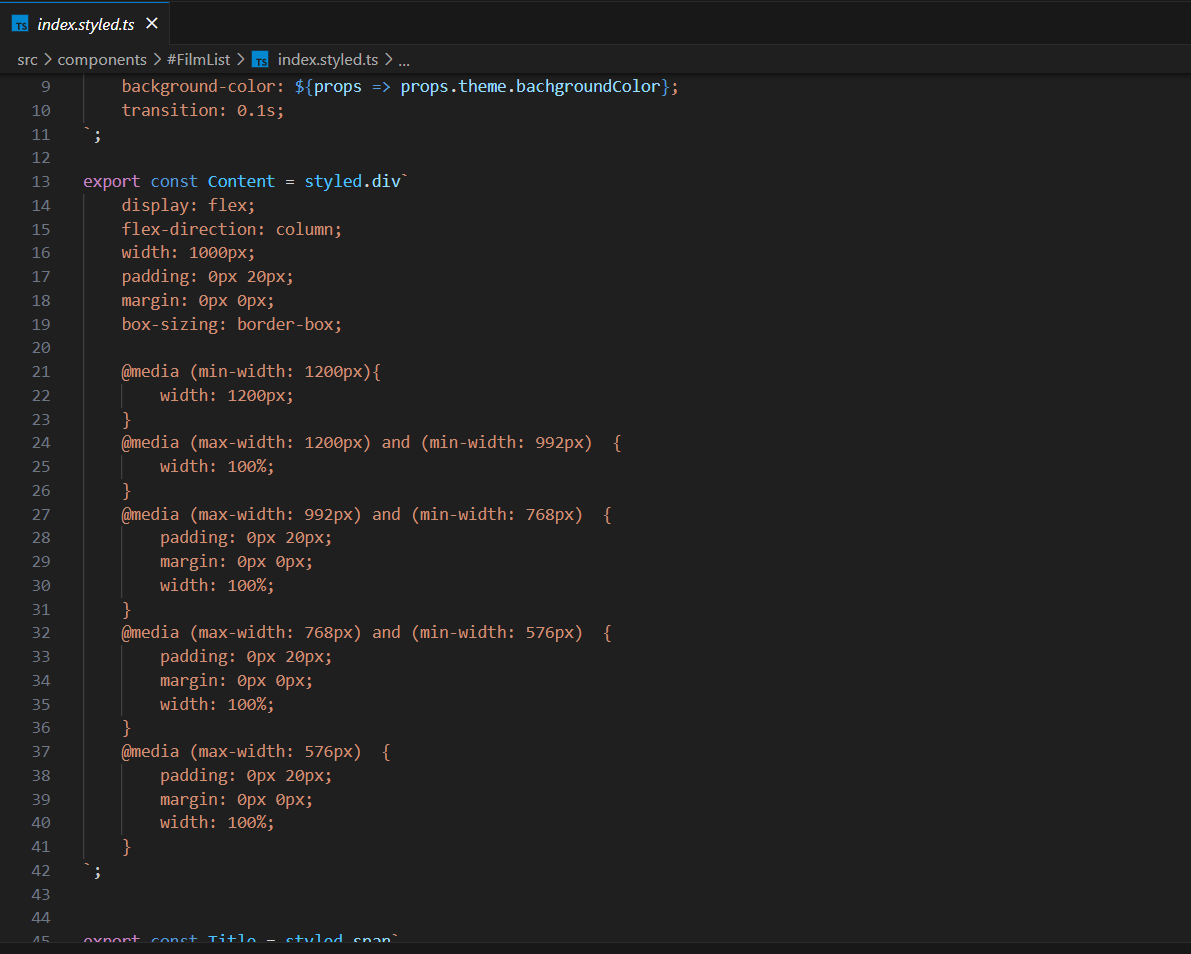


Рисунок 5 – Часть реализации адаптивности

Этот код создает стилизованный компонент Content, который адаптируется к разным размерам экрана, изменяя ширину и отступы в зависимости от заданных правил для каждого диапазона ширины экрана.

Медиа-запросы:

@media (min-width: 1200px) {...}: Если ширина экрана больше или равна 1200px, устанавливается ширина элемента в 1200 пикселей.

@media (max-width: 1200px) and (min-width: 992px) {...}: Если ширина экрана находится в пределах от 992px до 1200px, устанавливается ширина элемента на 100%.

Аналогичные медиа-запросы для других диапазонов ширин экрана, где ширина элемента также устанавливается на 100%, а отступы и размеры контента могут быть изменены в зависимости от разрешения.

Так же в ходе лобораторной работы был создан один из новых компонентов «Comments», который используется на странице с подробной информацией о фильме. На рисунках 6-8 показан код для этого компонента.

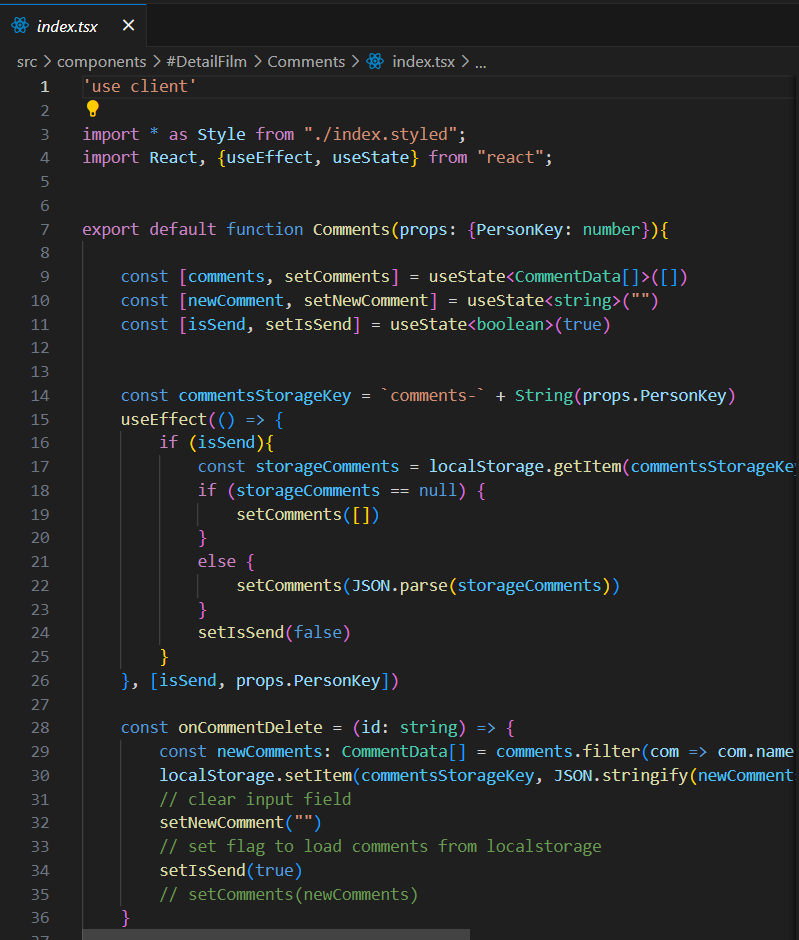


Рисунок 6 – Код для компонента «Comments»

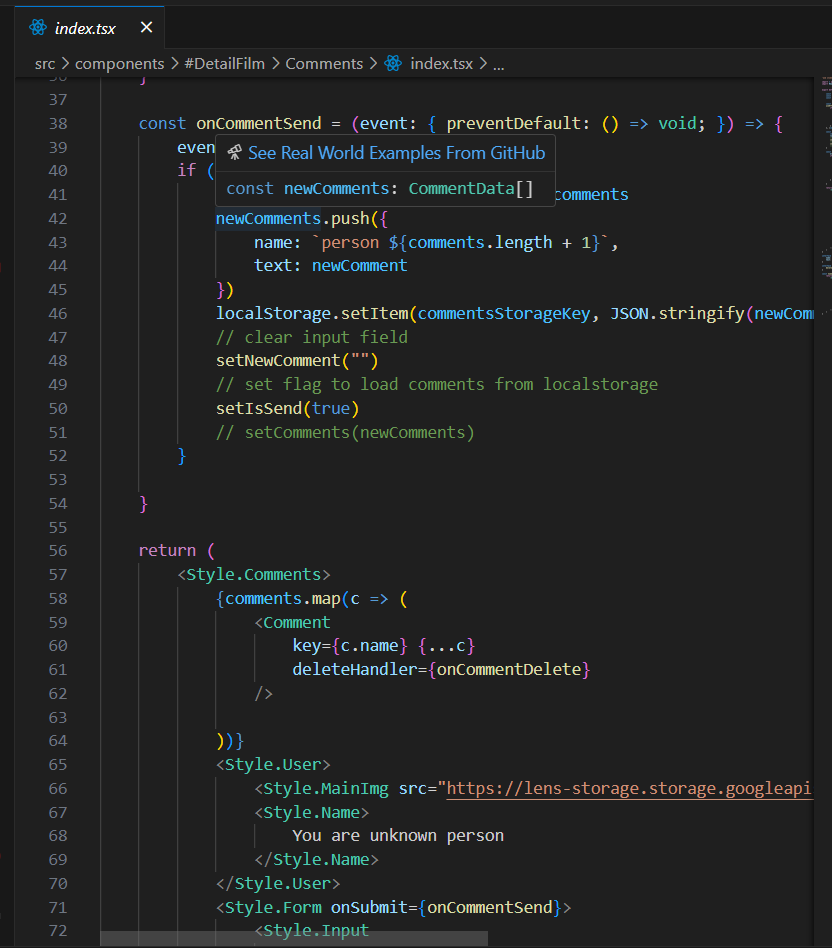


Рисунок 7 – Код для компонента «Comments»

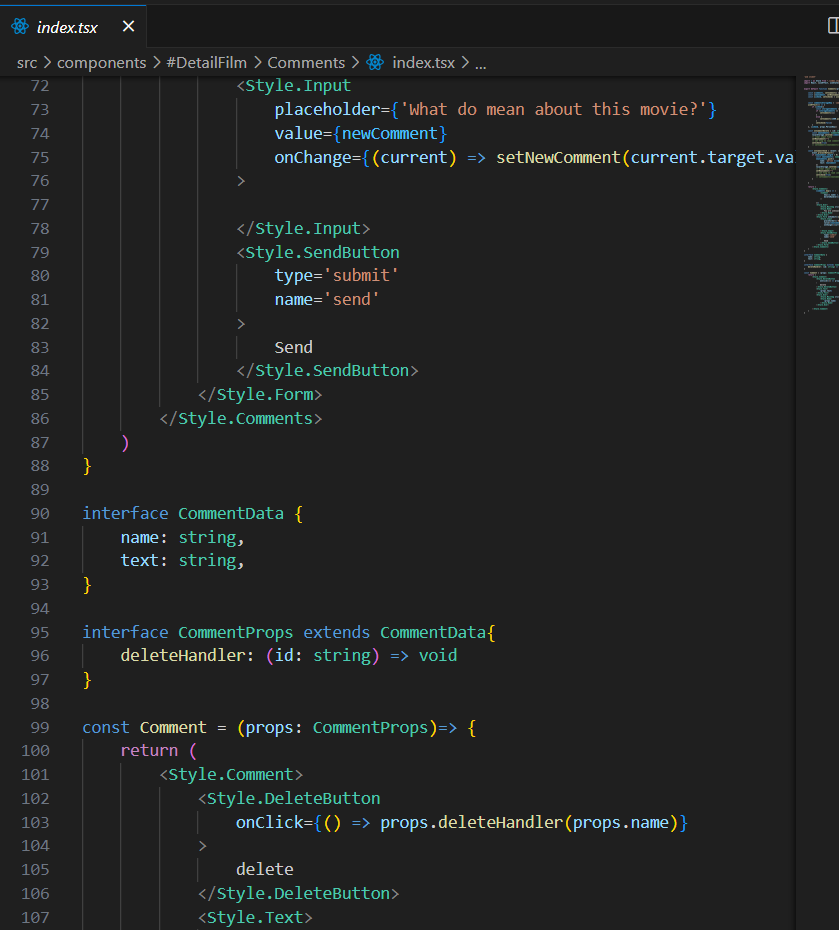


Рисунок 8 – Код для компонента «Comments»

Компонент создает интерфейс для взаимодействия с комментариями, сохраняет их в локальное хранилище, отображает на странице и позволяет добавлять и удалять комментарии.

Внутри компонента определены следующие состояния с использованием хука useState:

comments: массив комментариев.

newComment: новый комментарий, который пользователь вводит в поле ввода.

isSend: флаг для отслеживания отправки комментариев.

const commentsStorageKey = comments- + String(props.PersonKey); - Формирование ключа для сохранения комментариев в локальном хранилище браузера. Ключ зависит от PersonKey из props.

useEffect - Хук useEffect используется для выполнения действий после рендеринга компонента. В данном случае, при каждом изменении isSend и props.PersonKey, он проверяет локальное хранилище на наличие комментариев и обновляет состояние comments.

onCommentDelete - Функция для удаления комментария. Обновляет состояние comments, сохраняет его в локальное хранилище, очищает поле ввода и устанавливает флаг isSend для перезагрузки комментариев.

onCommentSend - Функция для отправки нового комментария. Добавляет новый комментарий в состояние comments, сохраняет его в локальное хранилище, очищает поле ввода и устанавливает флаг isSend для перезагрузки комментариев.

Возвращаемое значение return (...) - JSX-разметка компонента, включая отображение комментариев, формы ввода и кнопки отправки.

Comment - Дополнительный компонент, который отображает отдельный комментарий. Принимает props с данными о комментарии и функцию удаления комментария.

Интерфейсы CommentData и CommentProps определяют структуру данных для комментариев и их свойств соответственно.

# Заключение

В ходе данной лабораторной работы было разработано SPA исходя из описания задачи. Были визуализированы данные полученные от API в виде таблицы, содержащей только важную информацию о фильме. К каждому фильму из таблицы была реализована возможность оставлять комментарий (неограниченное количество), а также удалить его в любой момент времени.