**ОТЧЕТ**

**Step-1**

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Автоматически созданное описание

Для начала создали стили и нужные нам компоненты

import React from "react";

import ReactDOM from "react-dom";

import App from "./components/App/App";

import "./index.css";

ReactDOM.render(<App />, document.getElementById("root"));

Мы импортируем библиотеку React, которая используется для создания компонентов в React-приложениях.

Далее импортируем модуль ReactDOM из библиотеки React, который используется для взаимодействия с DOM (Document Object Model), а именно для отрисовки React-компонентов в реальном DOM.

Дальше мы импортируем компонент App из файла "./components/App/App". Важно отметить, что путь к файлу начинается с текущего каталога (./), что означает, что файл App находится в подкаталоге "components", который в свою очередь расположен в корневом каталоге проекта. <App />: Это синтаксис JSX, который предоставляет удобный способ создания React-элемента на основе компонента App.

document.getElementById("root"): Это DOM-элемент, в который будет производиться отрисовка компонента App. Обычно в HTML-документе должен существовать элемент с идентификатором "root".

ReactDOM.render(<App />, document.getElementById("root"));: Эта строка кода отвечает за отображение React-компонента App в элементе с идентификатором "root" внутри HTML-документа.

**Step-2**

import React from "react";

import "./Header.css";

const Header = () => {

    return (

        <div className="rmdb-header">

            <div className="rmdb-header-content">

                <img src="./images/reactMovie\_logo.png"

                    className="rmdb-logo"

                    alt="rmdb-logo"/>

                <img src="../images/tmdb\_logo.png"

                    className="rmdb-tmdb-logo"

                    alt="rmdb-tmdb-logo"/>

            </div>

        </div>

    );

};

export default Header;

Этот код представляет собой React-компонент Header, который, предназначен для отображения верхней части (заголовка) интерфейса приложения. Для начала мы определяем функциональный компонент Header. Функциональные компоненты представляют собой простые функции, которые принимают props (свойства) и возвращают React-элементы.

**Step-3**

import React, { Component } from "react";

import "./Home.css";

class Home extends Component {

  state = {};

  render() {

    return <div>Home</div>;

  }

}

export default Home;

Также импортировали нужные библиотеки и создали React-компонент Home, который является классовым компонентом и предназначен для отображения главной (страницы) интерфейса приложения. Для начала Мы создаем классовый компонент Home, который расширяет базовый класс Component из библиотеки React. Классовые компоненты обычно используются, когда требуется состояние (state) или методы жизненного цикла компонента. Далее здесь определено состояние компонента. В данном случае, у компонента Home нет конкретного состояния, поэтому используется пустой объект. Если бы у компонента было бы состояние, мы могли бы инициализировать его здесь.

**Step-4**

import React from "react";

import Header from "../elements/Header/Header";

import Home from "../Home/Home";

class App extends React.Component{

    render() {

        return(

            <div>

                <Home />

                <Header />

            </div>

        );

    }

}

export default App;

Этот код представляет собой основной компонент приложения App. Импортируем Реакт и нужные компоненты. Далее мы создаем классовый компонент App, который расширяет базовый класс Component из библиотеки React. Класс App представляет собой основной компонент всего приложения.

render() { ... }: Этот метод отвечает за отображение компонента App. Внутри метода render мы возвращаем JSX-элемент, который представляет собой корневой элемент приложения.