**npm create-react-app .**

**- React ı yükləyirik olduğumuz papka ya.**

**npm start**

**- React proyekt imizi başladırıq. ( Ayağa qaldırırıq )**

**\* Lazımsız şeyləri silirik təmizləmə aparırıq ilk öncə.**

**\* src içində components adlı papka açırıq.**

**- Component lərimiz burda olacaq.**

**- Uzantıları ister js istərsədə jsx yazarıq fərq etmir seçim bizimdir.**

**- Header.jsx component imizi yaradırıq.**

**- rfce tab edirik əlbətdə ki )))**

**return (**

**<header>**

**<div *className*="container">**

**<div *className*="inner-content">**

**<div *className*="brand">İzlenecekler</div>**

**<ul *className*='nav-links'>**

**<li>İzlenenler</li>**

**<li><i *className*='fas fa-plus'></i></li>**

**</ul>**

**</div>**

**</div>**

**</header>**

**)**

**App.js**

**<div *className*="App">**

**<*Header* />**

**</div>**

**- Hələki icon zad düşmür səhifəyə düzəldəcik tələsmə.**

**\* Link ləndirmələr Router lər üçün React Router DOM işlədəcik.**

**- Terminal 1 ini də açırıq + a klik edərək.**

**- Kitab a oxşar şeyə klik edib aralarında ox qoyuruq.**

**npm i react-router-dom**

**- React Router DOM paket ini yükləyirik.**

**return (**

**<*Router*>**

**<*Header* />**

**</*Router*>**

**);**

**import { Router } from "react-router-dom";**

**- Router i yazanda özü import edir içində də Header component ini yazırıq.**

**- Ama belə yazanda xəta verir.**

**import { *BrowserRouter* as Router } from "react-router-dom";**

**- BrowserRouter as lə yazanda xəta aradan qalxır artıq.**

**\* Router lərdə hansı Component lərin işləyəcəyini Route ə yazırıq.**

**- 1 dən çox olduğu üçün Routes yazacıq.**

**- Routes də yuxarı import olmalıdı diqqətli olaq.**

**return (**

**<*Router*>**

**<*Header* />**

**<*Routes*>**

**</*Routes*>**

**</*Router*>**

**);**

**import { *BrowserRouter* as Router, Routes } from "react-router-dom";**

**- import da edilib artıq Route lərimizi yaza bilərik.**

**<*Routes*>**

**<*Route* *path*="/" *element*={<*Watchlist* />} />**

**</*Routes*>**

**- İlk Route umuzu da yazdıq.**

**- path”/” Ana Səhifədə olanda. ( / ana səhifədə olanda deməkdir )**

**- Element Watchlist adlı Component imiz olacaq onu işlədəcək.**

**import { *BrowserRouter* as Router, Route, Routes } from "react-router-dom";**

**- Route da import olmalıdır diqqətli ol KəLə.**

**\* Watchlist component də yaradırıq src də.**

**- rfce edirik yəni )))**

**- App də Watchlist i yaratmamışdan öncə yazmışdıq.**

**- / dan qabaq ctrl + space edirik Watchlist yazılır.**

**- Ona klik edirik özü import edir Watchlist i App ə.**

**\* Header Component imiz Router in içindədir o bütün səhifələrdə görsənəcək.**

**\* React Router DOM u yüklədiyimizə görə Link işlədə bilərik artıq.**

**- Link də React Router DOM un yönləndiricisidir.**

**Header.jsx**

**<div *className*="brand">**

**<*Link*>İzlenecekler</*Link*>**

**</div>**

**- Ama düşmür səhifəyə.**

**- Bunlara da 1 link verməliyik.**

**- Bunu da to ilə veririk. ( to özəlliyi )**

**<*Link* *to*={"/"}>İzlenecekler</*Link*>**

**- Şarə Gəldi Şur qutardı getdi ))) Artıq düşür səhifəyə.**

**<*Link* *to*={"/watched"}>İzlenenler</*Link*>**

**- İzlenenler ə klik edəndə watched a gedir yuxarıda link yerində göstərir.**

**- Sadəcə hələki yazmadığımız üçün xəta verir watched ı.**

**\* src də Watched Component ini də yaradaq.**

**- Yəni bilirsən artıq rfce tab )))**

**App.js**

**<*Route* *path*="/watched" *element*={<*Watched* />} />**

**- Yeni 1 Route yaradırıq.**

**- Bu səfər deyirik ki path="/watched" olanda Watched component ini ver.**

**- Klik edəndə elə bil İzlenenler səhifəsine keçir.**

**- İzlənəcəklər a klik edəndə isə Ana Səhifəyə keçir sanki.**

**Header.jsx**

**<*Link* *to*={"/add"} *className*="btn"><i *className*="fas fa-plus"></i></*Link*>**

**- + icon una da klik edəndə Add Component inə yönləndirəcək.**

**- icon un görsənməsi üçün cdn dən qoşuruq.**

**App.js**

**<*Route* *path*="/add" *element*={<*Add* />} />**

**- Artıq icon a klik edəndə Add Component inə keçir.**

**\* Şablon u qurduq indi isə keçək içini dolduraq.**

**Add.jsx**

**return (**

**<div *className*='add-page'>**

**<div *className*="container">**

**<div *className*="add-content">**

**<img *src*="https://image.tmdb.org/t/p/w1920\_and\_h600\_multi\_faces\_filter(duotone,032541,01b4e4)/9ZyAUZrfccsjtDwYgc7yvOBnqM9.jpg" *alt*="#" />**

**<div *className*="titles">**

**<h1>Hoş Geldiniz.</h1>**

**<h2>Milyonlarca film, TV şovu ve keşfedilecek kişi. Şimdi keşfedin.</h2>**

**</div>**

**<div *className*="input-wrapper">**

**<input *type*="text" *placeholder*='Film, dizi, kişi ara...' />**

**</div>**

**</div>**

**</div>**

**</div>**

**)**

**- İndi bizə input da yazdığlarımızı tutacaq 1 dəyişən lazımdır.**

**const [ query , setQuery ] = useState("");**

**- Ona görə də State yaradırıq.**

**const [ query , setQuery ] = useState**

**- Yazıb Update olanı seçəndə.**

**import React, { useState } from 'react'**

**- React la eyni import olur.**

**const [ query , setQuery ] = useState**

**- Yazıb Add olanı seçəndə.**

**import { useState } from 'react'**

**- Ayrıca import edir.**

**<input *type*="text" *onChange*={ (*e*) => setQuery(*e*.target.value) } *placeholder*='Film, dizi, kişi ara...' />**

**- onChange lə input da dəyişiklik olduğca dəyərini setQuery() yə veririk.**

**- O da ötürür query yə.**

**console.log(query);**

**- input a yazdığca query nin dəyərlərinə log da baxa bilərik.**

**- Component daxilində return içində {query} yazaraq baxa da bilərik hər yerdə.**

**\* Bunu API dən axtarış edəcəyimiz üçün 1 Funksiya ya baxlayaq.**

**<input *type*="text" *onChange*={ onChange } *placeholder*='Film, dizi, kişi ara...' />**

**- input dəyişəndə onChange() Funksiya sı işləsin deyirik.**

**function onChange(*e*) {**

**setQuery(*e*.target.value);**

**}**

**- Funksiya da input un dəyərini ötürür query yə.**

**-** [**https://www.themoviedb.org/**](https://www.themoviedb.org/)

**- Bu saytdan çəkəcik data ları önce qeydiyyatdan keçirik.**

**- Ayarlar dan APİ yə giririk.**

**- APİ Anahtar ını götürürük.**

**- e4d71eac5d3f1a72564a09ad457e7bfd**

**- Bu açarı da google da tmdb api yazaraq search edirik.**

**- TMDb - The Movie Database API link i gəlir.**

**-** [**Getting Started**](https://developers.themoviedb.org/3/getting-started/images) **var aşağıda ora klik edirik.**

**- https://developers.themoviedb.org/3/getting-started/introduction**

**- Bu sayt açılır klik edəndə.**

**- Edə biləcəyimiz hər şey var burda.**

**- SEARCH ə giririk biz.**

**- Ordan da Search Movies ə.**

**- Try it out var orda api key ə**

**- APİ Anahtar ını yazırıq.**

[**https://api.themoviedb.org/3/search/movie?api\_key=e4d71eac5d3f1a72564a09ad457e7bfd&language=en-US&page=1&include\_adult=false**](https://api.themoviedb.org/3/search/movie?api_key=e4d71eac5d3f1a72564a09ad457e7bfd&language=en-US&page=1&include_adult=false)

**- Belə 1 link verir onu da Fetch ə yazıb data ları çəkirik.**

**function onChange(*e*) {**

**setQuery(*e*.target.value);**

**fetch(`https://api.themoviedb.org/3/search/movie?api\_key=e4d71eac5d3f1a72564a09ad457e7b fd&language=en-US&page=1&include\_adult=false&query=${*e*.target.value}`)**

**.then( *res* => *res*.json() )**

**.then( *data* => console.log(*data*) )**

**}**

**- onChange() Funksiya sında yazırıq Fetch i input dəyişdikcə işləyir.**

**- input in value suna uyğun data ları verir.**

**<input *value*={query} *type*="text" *onChange*={ onChange } *placeholder*='Film, dizi, kişi ara...' />**

**- input un value sunu da query edirik.**

**const [ results , setResults ] = useState( [] );**

**- Gələn data nın içindəki results var bizə o lazımdı onu alacıq.**

**.then( *data* => setResults(*data*.results) )**

**- Xeirli uğurlu olsun artıq result da data dakı results lar olacaq value ya uyğun.**

**{**

**results.map( ( *movie* ) => (**

**<h1>{*movie*.title}</h1>**

**) )**

**}**

**- input div inin altında yazırıq.**

**- Artıq title şəkil kimi şeylər əl çatandır işlədə bilərik.**

**- Bu formada yazdığımız üçün zatən title lar düşür ekrana.**

**\* .env.local sənədi yaradırıq ümumidə bilgisayar-genetigi nin içində.**

**- APİ key ni açıqda qoymaq istəmirik.**

**- Təhlükəsizlik üçün belə edirik.**

**- Github a push edərkəndə bunu push etmirik.**

**- dat i en vi sənədi deyilir deyəsən )))**

**- İçində dəyişən yaradırıq başqa heccənəz.**

**REACT\_APP\_TMDB\_KEY = e4d71eac5d3f1a72564a09ad457e7bfd  
  
 - APİ Key i 1 dəyişənə məsimsədirik.**

**fetch(`https://api.themoviedb.org/3/search/movie?api\_key=e4d71eac5d3f1a72564a09ad457e7bfd&language=en-US&page=1&include\_adult=false&query=${*e*.target.value}`)**

**- Belə açıqca görsənir APİ key.**

**fetch(`https://api.themoviedb.org/3/search/movie?api\_key=${process.env.REACT\_APP\_TMDB\_KEY}&language=en-US&page=1&include\_adult=false&query=${*e*.target.value}`)**

**- Api Key yerinə artıq dəyişəni yazırıq.**

**- Ama işləmir xəta verir.**

**- Çünki env yaradandan sonra proyekt i yenidən başlatmaq lazımdır.**

**- Terminal da ctrl + c edirik Y/N soruşur y deyirik.**

**- npm start edirik.**

**- Və budur DəFiKo gül kimi işləyir.**

**- Ama siləndə xəta verir map geri qayıda bilmədiyi üçün React da xəta verir.**

**- Səhifə açılmır.**

**.then( *data* => {**

**if ( !*data*.errors ){**

**setResults(*data*.results);**

**}else {**

**setResults( [] );**

**}**

**} )**

**- İndi deyirik ki əgər Error yoxdursa data dakı results ları göstər.**

**- Else ə ehtiyac yoxdur əslində yazmasaq da işləyir.**

**- Hər ehtimala qarşı da deyirik ki xəta olarsa boş Array ver.**

**- Ki map xəta verməsin.**

**{**

**results.length > 0 && (**

**<ul *className*='results'>**

**{**

**results.map( ( *movie* ) => (**

**<li *key*={*movie*.id}>**

**<*ResultCart* *movie*={*movie*} />**

**</li>**

**) )**

**}**

**</ul>**

**)**

**}**

**- Deyirik ki əgər yazdığımız şey varsa log da göstər yoxdursa boş göstərmə.**

**- key göndəririk ki xəta verməsin.**

**- ResultCart component i yaradıb bura veririk.**

**- Props la da movie ni göndəririk yəni data ları.**

**\* src də ResultCart.jsx Component ini yaradırıq.**

**- rfce və ya race tab edə bilərik dədə baba üsulu )))**

**- Add.jsx ə import etmişik bunu o zaman içini dolduraq.**

**return (**

**<div *className*="result-card">**

**<div *className*="poster-wrapper">**

**<img *src*={`https://image.tmdb.org/t/p/w200/${*movie*.poster\_path}`} *alt*={`${*movie*.poster\_path}`} />**

**</div>**

**</div>**

**)**

**- https ilə olan sayt ın özünün verdiyi linkdi arxasına movie dəki data nı əlavə edirik.**

**- Bütün şəkilləri verir ama şəkil olmayanları da verir çirkin görünür.**

**- Şəkil açılmayanda balaca əzik necə gəlirsə elə gəlir.**

**return (**

**<div *className*="result-card">**

**<div *className*="poster-wrapper">**

**{**

***movie*.poster\_path ? (**

**<img *src*={`https://image.tmdb.org/t/p/w200/${*movie*.poster\_path}`} *alt*={`${*movie*.poster\_path}`} />**

**) : (**

**<div *className*="filler-poster"></div>**

**)**

**}**

**</div>**

**</div>**

**)**

**- Belə yazanda artıq poster\_path varsa image i verir yoxdursa biz yazan dizayn ı verir.**

**- Artıq çirkin görüntü olmur.**

**<div *className*="info">**

**<div *className*="header">**

**<h3 *className*="title">{ *movie*.title }</h3>**

**<h4 *className*="release-date">**

**{**

***movie*.release\_date ? *movie*.release\_date.substring(0,4) : "-"**

**}**

**</h4>**

**<h4 *className*="release-date">**

**IMDB: <b>{**

***movie*.vote\_average ? *movie*.vote\_average : "-"**

**}</b>**

**</h4>**

**</div>**

**</div>**

**- Tarixi tam istəmirik substringlə ilk 4 rəqəmi götürürük.**

**- Tarix varsa verir yoxdursa – verir.**

**- IMDB varsa verir yoxdursa – verir.**

**<div *className*="controls">**

**<button *className*="btn">Add to Watchlist</button>**

**</div>**

**- info nun içində header ın paralelində yaradırıq.**

**- Klik edəndə İzlənəcəklər ə düşməlidir.**

**- Çox yerdə istifadə edəcik ona görə globalda tutmalıyıq.**

**\* src də context papka sı yaradırıq.**

**- Onun da içində GlobalState.jsx yaradırıq.**

**- React ın məntiqi düzdür parent dən child a göndərir.**

**- Proses çox uzanır ötür – ötür düşür.**

**- Ona görə də Context APİ işlədilir.**

**- Bütün sənədləri bütün Component lərə göndərir.**

**- Sənin tək tək göndərməyin lazım deyil deyir.**

**\* 1 dənə Global Context yaradacıq.**

**export const GlobalContext = createContext();**

**- Yazırıq özü import edir yuxarı.**

**import { createContext } from "react";**

**- React ın özəlliyidir yükləyəndə içində gəlir.**

**\* 1 dənə də Global Provider yaradırıq.**

**export const GlobalProvider = (*props*) => {**

**return <*GlobalContext.Provider*>{ *props*.children }</*GlobalContext.Provider*>**

**}**

**- Alt dakı bütün data ları almaq üçün props deyirik.**

**- props.children deyərəkdə burda olanları istədiyimiz yerdə işlədirik.**

**\* Sonra da App.js də işlədirik.**

**- Router i atırıq GlobalProvider a.**

**return (**

**<*GlobalProvider*>**

**<*Router*>**

**<*Header* />**

**<*Routes*>**

**<*Route* *path*="/" *element*={<*Watchlist* />} />**

**<*Route* *path*="/watched" *element*={<*Watched* />} />**

**<*Route* *path*="/add" *element*={<*Add* />} />**

**</*Routes*>**

**</*Router*>**

**</*GlobalProvider*>**

**);**

**import { GlobalProvider } from "./context/GlobalState";**

**- Özü import olur yuxarıda diqqətli olmaq lazımdır yenədə.**

**- Artıq işlədə bilərik sadəcə dəyər olmadığı üçün hələki xəta verir.**

**export const GlobalProvider = (*props*) => {**

**return <*GlobalContext.Provider***

***value*={{emin: "developer"}}>{*props*.children}**

**</*GlobalContext.Provider*>;**

**};**

**- Dəyər göndərdik artıq bunu harda istəsək işlədə bilərik.**

**Add.jsx**

**function Add() {**

**const { emin } = useContext(GlobalContext);**

**- emin deyə 1 dəyər var deyirik.**

**- 1 context işlətmək istəyirik deyirik.**

**- useContext i özü yuxarı import edir yazan kimi.**

**- Hansı Context i işlədəcəyimizi yazırıq useContext() də.**

**- GlobalContext yazan kimi özü import edir yuxarı.**

**- İndi harda istəsək işlədirik.**

**console.log(emin);**

**developer**

**<h1>Hoş Geldiniz.</h1><b>{emin}</b>**

**developer**

**- return içində də bu cür işlədirik bəzəkli mötərizə içində çünki JS kodu yazırıq.**

**\* useReducer Hook u.**

**- 1 dən çox bağlantılı State varsa bunları daha oxunaqlı etmək üçün işlədilir.**

**- Kodlar daha oxunaqlı olur və lazımsız yerə Render olunmur.**

**- useReducer 2 parametr alır.**

**- 1 i Reducer Funksiya sı.**

**- 2 cisi başlanğıc dəyəri.**

**GlobalState.jsx**

**export const GlobalProvider = (*props*) => {**

**const [ state , dispatch ] = useReducer(AppReducer , initialState);**

**- Yazan kimi özü import edir.**

**import { useReducer } from "react";**

**- AppReducer – Reducer Funksiya sıdır.**

**- initialState – Başlanğıc dəyəridir.**

**const initialState = {**

**watchlist: [],**

**watched: [],**

**};**

**- Başlanğıc dəyəri boş Array elədik.**

**\* context in içində AppReducer.jsx yaradırıq.**

**export default ( *state* , *action* ) => {}**

**GlobalState.jsx**

**- useReducer() də AppReducer in yanında ctrl + space edərək import edirik.**

**- Yazıb sonradan yaradanda belə edirik.**

**console.log(state);**

**{watchlist: Array(0), watched: Array(0)}**

**- log da baxa bilərik maraqlıdırsa nədir bu state )))**

**- 2 buton qoyuruq biri İzlənənlər biri İzlənəcəklər.**

**- Hansına klik etsək uyğun olaraq bu 2 sindən 1 inə düşəcək.**

**AppReducer.jsx**

**- action göndəririk.**

**- type nı bildiririk.**

**- type lar = olarsa işini görəcək.**

**- switch case işlədəcik.**

**- 1 action var və action içində type gələcək.**

**export default ( *state* , *action* ) => {**

**switch( *action*.type ){**

**case "ADD\_MOVIE\_TO\_WATCHLIST":**

**return {**

**...*state*,**

**watchlist: [ ...*state*.watchlist , *action*.payload ],**

**};**

**default:**

**return *state*;**

**}**

**};**

**- type ı case ə yazırıq 1 də Funksiya kimi yazacıq.**

**- ...state lə bütün State ləri alırıq. ( initialState də olanların hamısını )**

**- watchlist i düzəldirik.**

**- state in içindəki watchlist i alır.**

**- Sonra action da hansı dəyəri göndərmişiksə onu əlavə edir.**

**- Və yeni 1 Array yaradır.**

**- Heç 1 i olmayanda da default state i qaytarır.**

**- Bunu yazdıq indi bunu harda işlədəcik ?**

**GlobalState.jsx**

**- GlobalState də 1 Funksiya yazacıq. ( Provider ın içində )**

**- İçində də dispatch i işlədəcik.**

**const addMovieToWatchlist = ( *movie* ) => {**

**dispatch( { type: "ADD\_MOVIE\_TO\_WATCHLIST" , payload: *movie* } );**

**}**

**- Deyirik ki tıkladığımız film i watchlist Array inə at.**

**- Buna görə də Global da tuturuq.**

**- Parametr olaraq movie ni alır.**

**- Biz göndərəcik.**

**- dispatch – type götürür.**

**- type da AppReducer də case ə verdiyimizdir.**

**- 1 də payload götürür.**

**- payload – göndərdiyimiz dəyərdir.  
  
AppReducer.jsx**

**- state – başlanğıc dəyərini göndərdiyimiz State dəyəridir.**

**- ...state – bütün State dəyərini al deyirik.**

**const initialState = {**

**watchlist: [],**

**watched: [],**

**};**

**- Yəni ki bunu.**

**watchlist: [ ...*state*.watchlist , *action*.payload ],**

**- ...*state*.watchlist – state in içində ki watchlist i al deyirik.**

**- watchlist: [], - state in içində ki watchlist budur.**

**- Yanına da action dan gələn payload ı əlavə et deyirik.**

**dispatch( { type: "ADD\_MOVIE\_TO\_WATCHLIST" , payload: *movie* } );**

**- action dan gələn payload isə movie dur.**

**\* Bu movie uda tıkladığımız yerdən alacıq yəni ResultCart dan.**

**- Funksiya nı o zaman ResultCart.jsx ə göndərməliyik.**

**return (**

**<*GlobalContext.Provider* *value*={{ addMovieToWatchlist }}>**

**{*props*.children}**

**</*GlobalContext.Provider*>**

**);**

**- Funksiya GlobalState.jsx də GlobalProvider dadır burdan yollayırıq.**

**- Harda lazımdırsa orda tutub işlədirik.**

**ResultCart.jsx**

**function ResultCart( { *movie* } ) {**

**const { addMovieToWatchlist } = useContext(GlobalContext);**

**- useContext() i import edirik.**

**- Deyirik ki bu Funksiya nı götür yerini də useContext də yazırıq.**

**- GlobalContext dən addMovieToWatchlist funksiya sını götürür.**

**import React, { useContext } from 'react'**

**import { GlobalContext } from '../context/GlobalState';**

**- Yazdığca özü import edir yenədə diqqətli olmaq lazımdır 1 dən etməz.**

**<button *className*="btn" *onClick*={ () => addMovieToWatchlist(*movie*) }>Add to Watchlist</button>**

**- Funksiya nı göndərib aldığımıza görə işlədə bilərik.**

**- Artıq buton a klik edəndə Funksiya işləyir və parametr olaraq movie götürür.**

**- Artıq nəyə klik etsək atacaq Array ə.**

**{watched: Array(0), watchlist: Array(3)}**

**- F12 – Components – GlobalProvider**

**- Da sağda hook da Reducer də göstərir.**

**- Klik etdikcə watchlist artır.**

**GlobalState.jsx**

**- watchlist i göndərək o zaman.**

**return (**

**<*GlobalContext.Provider* *value*={{**

**watchlist: state.watchlist,**

**addMovieToWatchlist }}>**

**{*props*.children}**

**</*GlobalContext.Provider*>**

**);**

**- Globalda göndərdik harda lazımdırsa tutub işlədəcik.**

**Watchlist.jsx**

**- watchlist də tuturuq məlum məsələdir ki.**

**- İzlənəcəklər səhifəmizdə.**

**import React, { useContext } from 'react'**

**import { GlobalContext } from '../context/GlobalState';**

**function Watchlist() {**

**const { watchlist } = useContext(GlobalContext);**

**return (**

**<div>**

**{**

**watchlist.map( ( *movie* ) => (**

**<h1>{*movie*.title}</h1>**

**) )**

**}**

**</div>**

**)**

**}**

**export default Watchlist**

**- watchlist i işlədəcəyimizi deyirik.**

**- Bunun üçün useContext və GlobalContext i import edirik.**

**- Klik etdiyimizin title ını yazdırırıq Array ə.**

**- Yəniki İzlənəcəklər ə.**

**- Yəniki watchlist də.**

**- Səhifəni yeniləyəndə silinir qalmır.**

**\* Storage ə atmaq lazımdır ona görədə.**

**- GlobalState də deyəcik ki.**

**- State dəyişəndə bunu Global olaraq Local Storage a qeyd elə.**

**- Asılılıq var ona görədə useEffect() işlədəcik.**

**useEffect( () => {**

**localStorage.setItem("watchlist" , JSON.stringify(state.watchlist) )**

**}, [state] );**

**- watchlist deyə 1 yer açırıq Local Storage də. ( "watchlist" )**

**- Bunun içinədə watchlist ı ötürürük. ( state.watchlist )**

**- JSON a çevirib sonra ötürürük.**

**- Nəyə klik etsək Storage ə atır ama yeniləyəndə səhifəni silinir.**

**- Bunun da qarşısına başlanğıc dəyəri ilə keçirik.**

**- Deyəcik ki səhifə yenilənəndə əgər local storage in içində watchlist varsa.**

**- Onun dəyərlərini al.**

**- Yoxdursa boş göndər.**

**const initialState = {**

**watchlist: localStorage.getItem("watchlist") ? JSON.parse(localStorage.getItem("watchlist")) : [],**

**watched: [],**

**};**

**- Belə 1 şey ay KəLə.**

**- Artıq səhifə yenilənəndə silinmir qalır Storage.**

**- JSON u varsa alır JS ə çevirir onuda watchlist ə atır.**

**\* O zaman watchlist i düzəldək təkcə title deyil normal şəkildə düşsün.**

**return (**

**<div *className*="movie-page">**

**<div *className*="container">**

**<div *className*="header">**

**<h1 *className*="heading">**

**İzlenecek Filmler**

**</h1>**

**</div>**

**{**

**watchlist.length > 0 && (**

**<div *className*="movie-grid">**

**{**

**watchlist.map( ( *movie* ) => (**

**<*MovieCard* *movie*={ *movie* } />**

**) )**

**}**

**</div>**

**)**

**}**

**</div>**

**</div>**

**)**

**- Əgər watchlist varsa içindən movie la keçib MovieCard component ini qaytarırıq.**

**- movie uda props la göndəririk.**

**\* components də MovieCard.jsx Component ini yaradırıq.**

**- rfce və ya race tab )))**

**const MovieCard = ({ *movie* }) => {**

**return (**

**<div *className*="movie-card">**

**<div *className*="overlay"></div>**

**{**

***movie*.poster\_path ? (**

**<img**

***src*={`https://image.tmdb.org/t/p/w200/${*movie*.poster\_path}`}**

***alt*={`${*movie*.poster\_path}`}**

**/>**

**) : (**

**<div *className*="filler-poster"></div>**

**)**

**}**

**</div>**

**);**

**};**

**- Props u götürürük sonra da deyirik ki.**

**- true olarsa şəkili göstər false olarsa dizayn ımızı.**

**- ResutCard da yazdığımızın eynisidir.**

**- Artıq nəyə klik eləsək İzlənəcəklərə düşür adam balası kimi )))**

**\* Qaldı buton ları əlavə etmək.**

**- Həm İzlənənlər də var həm də ki İzlənəcəklər də.**

**- Ona görə də mərkəzi Component də idarə edəcik.**

**- components də MovieControls.jsx yaradırıq.**

**- rfce və ya race tab.**

**MovieCard.jsx**

**}**

**<*MovieControls* *movie*={*movie*} />**

**</div>**

**);**

**- MovieControls a props göndəririk.**

**MovieControls**

**return (**

**<div *className*="inner-card-controls">**

**<button *className*="ctrl-btn">**

**<i *className*="fa-fw far fa-eye"></i>**

**</button>**

**<button *className*="ctrl-btn">**

**<i *className*="fa-fw fa fa-times"></i>**

**</button>**

**</div>**

**)**

**- Film lərin üzərinə gələndə göz və x icon ları çıxır.**

**\* x ə klik edəndə silinsin istəyirik həmin film siyahı dan.**

**- GlobalState də yazırıq bunu.**

**const removeMovieFromWatchlist = ( *id* ) => {**

**dispatch( { type: "REMOVE\_MOVIE\_TO\_WATCHLIST" , payload: *id* } );**

**}**

**- Klik etdiyimiz buton dan ID ni göndərib ID yə görə siləcik.**

**- type adını nə istəsək verə bilərik ama mənalı olmalıdır.**

**- AppReducer dəki type ilə eyni olmalıdır.**

**- Çünki action type a görə yoxlayır.**

**AppReducer.jsx**

**case "REMOVE\_MOVIE\_TO\_WATCHLIST":**

**return {**

**...*state*,**

**watchlist: *state*.watchlist.filter(**

**( *movie* ) => *movie*.id !== *action*.payload**

**),**

**};**

**- Normal React silmə prosesidir.**

**- ID si eyni olandan başqa hamısını göstərir.**

**GlobalState.jsx**

**return (**

**<*GlobalContext.Provider* *value*={{**

**watchlist: state.watchlist,**

**removeMovieFromWatchlist,**

**addMovieToWatchlist }}>**

**{*props*.children}**

**</*GlobalContext.Provider*>**

**);**

**- remove funksiya sını Global da göndəririk ki lazım olan yerdə işlədək.**

**MovieControls.jsx**

**const MovieControls = (*movie*) => {**

**const { removeMovieFromWatchlist } = useContext(GlobalContext);**

**- MovieCard.jsx də movie u göndərmişdik burda tutub işlədirik.**

**- Funksiya nı işlətmək üçün də useContext və GlobalContext i import edirik.**

**<button *className*="ctrl-btn" *onClick*={ () => removeMovieFromWatchlist(*movie*.id) }>**

**<i *className*="fa-fw fa fa-times"></i>**

**</button>**

**- Funksiya nıda buton a bağlayırıq klik olanda ID ni göndərir.**

**- Artıq hansı film də x ə klik etsək siləcək.**

**\* Göz ə klik edəndə isə watched ə düşməlidir.**

**- Məntiq eynidir yenə Global da 1 Funksiya yazacıq.**

**- ID sinə görə göndərəcik.**

**GlobalState.jsx**

**const addMovieToWatched = ( *movie* ) => {**

**dispatch( { type: "ADD\_MOVIE\_TO\_WATCHED" , payload: *movie* } );**

**}**

**return (**

**<*GlobalContext.Provider* *value*={{**

**watchlist: state.watchlist,**

**removeMovieFromWatchlist,**

**addMovieToWatched,**

**addMovieToWatchlist }}>**

**{*props*.children}**

**</*GlobalContext.Provider*>**

**);**

**- Funksiya nı yaradıb Global da göndəririk.**

**- payload da movie göndəririk ID ni götürəcik içindən.**

**MovieControls.jsx**

**const MovieControls = ( { *movie* } ) => {**

**const { removeMovieFromWatchlist , addMovieToWatched } = useContext(GlobalContext);**

**- Gələn addMovieToWatched Funksiya sını da götürürük.**

**<button *className*="ctrl-btn" *onClick*={ () => addMovieToWatched( *movie* ) }>**

**<i *className*="fa-fw far fa-eye"></i>**

**</button>**

**- Göz ə klik edəndə movie parametr ini Funksiya ya ötürür.**

**AppReducer.jsx**

**- Göz ə klik edəndə watchlist dən silinir və watched ə əlavə olunur.**

**case "ADD\_MOVIE\_TO\_WATCHED":**

**return {**

**...*state*,**

**watchlist: *state*.watchlist.filter(**

**( *movie* ) => *movie*.id !== *action*.payload.id**

**),**

**};**

**- state i parçalayır watchlist i filter edir.**

**- movie nin ID si action da göndərdiyimiz payload ın ID sinə = deyilsə göstər.**

**- = olanı da at siyahıdan.**

**- göz ə klik edəndə silinir ama biz watched ə əlavə olunmasını istəyirik.**

**case "ADD\_MOVIE\_TO\_WATCHED":**

**return {**

**...*state*,**

**watchlist: *state*.watchlist.filter(**

**( *movie* ) => *movie*.id !== *action*.payload.id**

**),**

**watched: [...*state*.watched , *action*.payload],**

**};**

**- Deyirik ki watched i parçala sildiyin ID ni əlavə et.**

**- F12 – Components – GlobalProvider – Sağda Hook**

**- Da baxa bilərik klik etdikcə film lərə düşür watchlist də.**

**- Göz klik edəndə isə watchlist dən silinir watched ə düşür.**

**\* Watched də görsənməsi üçün watchlist dəki kimi buranı hazırlamaq lazımdır.**

**- watchlist.jsx copy edib atırıq bura.**

**- watchlist ləri watched edirik əlbətdə ki ))))**

**<*GlobalContext.Provider* *value*={{**

**watched: state.watched,**

**watchlist: state.watchlist,**

**- Global da da watchlist kimi watched i də göndəririk.**

**\* Add buton unu disabled etmək lazımdır ki 1 dəfə əlavə eyni şeyi sonsuza dək deyil.**

**- buton u disabled edəndə ümumiyətcə klik etmək olmur.**

**- Deyirik ki əgər klik olunan element watchlist də varsa 1 də əlavə olmasın.**

**- Klik olmasın yəni varsa disabled olsun buton.**

**ResultCart.jsx**

**const { watchlist , addMovieToWatchlist } = useContext(GlobalContext);**

**- watchlist i Global dan götürürük.**

**const storedMovie = watchlist.find( ( *f* ) => *f*.id === *movie*.id );**

**- watchlist in içindən keçirik bütün ID lərə baxırıq ID lər bənzərsiz olur.**

**- Əgər varsa true olacaq storedMovie yoxdursa false olacaq klik edə biləcik.**

**<button *disabled*={storedMovie} *className*="btn" *onClick*={ () => addMovieToWatchlist(*movie*) }>Add to Watchlist</button>**

**- Artıq 1 film ə 2 dəfə klik etmək olur 1 dəfə klik olunubsa buton disabled olur.**

**Watchlist.jsx & Watched.jsx**

**<*MovieCard* *movie*={ *movie* } *key*={*movie*.id} />**

**- 2 sində də MovieCard a key göndəririk ki xəta verməsin.**

**GlobalState.jsx**

**const initialState = {**

**watchlist: localStorage.getItem("watchlist") ? JSON.parse(localStorage.getItem("watchlist")) : [],**

**watched: [],**

**};**

**- Başlanğıc dəyərlərində watchlist i Storage ə almışıq ona görə də yenilənmədə silinmir.**

**- Ama watched dəki data lar silinir çünki Storage ə almamışıq.**

**useEffect( () => {**

**localStorage.setItem("watchlist" , JSON.stringify(state.watchlist) )**

**}, [state] );**

**- useEffect də watchlist üçün Storage yaratmışıq.**

**- watched üçün yaratmamışıq.**

**const initialState = {**

**watchlist: localStorage.getItem("watchlist") ? JSON.parse(localStorage.getItem("watchlist")) : [],**

**watched: localStorage.getItem("watched") ? JSON.parse(localStorage.getItem("watched")) : [],**

**};**

**useEffect( () => {**

**localStorage.setItem("watchlist" , JSON.stringify(state.watchlist) )**

**localStorage.setItem("watched" , JSON.stringify(state.watched) )**

**}, [state] );**

**- watched üçün də Local Storage yaradıb başlanğıc dəyərinə veririk.**

**- watched də də data lar yenilənmə olanda silinmir qalır hamısı.**

**Watchlist.jsx & Watched.jsx**

**<*MovieCard* *movie*={ *movie* } *key*={*movie*.id} *type*="watchlist" />**

**<*MovieCard* *movie*={ *movie* } *key*={*movie*.id} *type*="watched" />**

**- Ayıra bilmək üçün 2 sində də type göndəririk MovieCard a.**

**MovieCard.jsx**

**const MovieCard = ({ *movie* , *type* }) => {**

**- Burda da tuturuq type ı.**

**<*MovieControls* *movie*={*movie*} *type*={*type*} />**

**- type da props olaraq göndəririk MovieControls a.**

**MovieControls.jsx**

**const MovieControls = ( { *movie* , *type* } ) => {**

**- Burda da tutub işlədirik type ı.**

**\* Button ların işləri fərqlidir.**

**- watchlist də ki göz watched a əlavə edir X isə watchlist dən silir.**

**- watched də ki göz watchlist a əlavə edir X isə watched dən silir.**

**return (**

**<div *className*="inner-card-controls">**

**{*type* === "watchlist" && (**

**<>**

**<button *className*="ctrl-btn" *onClick*={() => addMovieToWatched(*movie*)}>**

**<i *className*="fa-fw far fa-eye"></i>**

**</button>**

**<button *className*="ctrl-btn" *onClick*={() => removeMovieFromWatchlist(*movie*.id)}>**

**<i *className*="fa-fw fa fa-times"></i>**

**</button>**

**</>**

**)}**

**</div>**

**);**

**- Artıq watched bu buton lar çıxmır sadəcə watchlist də çıxır.**

**- Eynisini watched üçün edəcik type a görə dəyişəcək.**

**- watched sadəcə olaraq göz icon u dəyişəcək üstündən xətt çəkilmiş göz olacaq.**

**- Və Funksiya lar dəyişəcək işləri fərqli olduğu üçün.**

**- Belə şablon eynidir.**

**\* GlobalState.jsx də watched üçün funksiya larımızı yazaq.**

**const moveToWatchlist = ( *movie* ) => {**

**dispatch( { type: "MOVE\_TO\_WATCHED" , payload: *movie* } );**

**}**

**<*GlobalContext.Provider* *value*={{**

**moveToWatchlist,**

**- Funksiya nı yaradıb Global da göndəririk.**

**AppReducer.jsx**

**case "MOVE\_TO\_WATCHED":**

**return {**

**...*state*,**

**watched: *state*.watched.filter(**

**(*movie*) => *movie*.id !== *action*.payload.id**

**),**

**watchlist: [...*state*.watchlist, *action*.payload],**

**};**

**- Üstü Xətli Göz ə klik edəndə watchlist də əlavə edir.**

**MovieControls.jsx**

**const { removeMovieFromWatchlist, addMovieToWatched , moveToWatchlist } = useContext(GlobalContext);**

**- Ama ilk öncə Funksiya nı useContext lə götürüb buton a bağlamaq lazımdır.**

**<button *className*="ctrl-btn" *onClick*={() => moveToWatchlist(*movie*)}>**

**<i *className*="fa-fw far fa-eye-slash"></i>**

**</button>**

**- İndi işləyir artıq.**

**AppReducer.jsx**

**case "REMOVE\_MOVIE\_TO\_WATCHED":**

**return {**

**...*state*,**

**watched: *state*.watched.filter(**

**(*movie*) => *movie*.id !== *action*.payload**

**),**

**};**

**- X ə klik edəndə watched dən sil deyirik.**

**GlobalState.jsx**

**const removeMovieFromWatched = ( *id* ) => {**

**dispatch( { type: "REMOVE\_MOVIE\_TO\_WATCHED" , payload: *id* } );**

**}**

**<*GlobalContext.Provider* *value*={{**

**removeMovieFromWatched,**

**- Funksiya nı yazıb Global da göndəririk.**

**MovieControls.jsx**

**const { removeMovieFromWatchlist, addMovieToWatched , moveToWatchlist , removeMovieFromWatched } = useContext(GlobalContext);**

**<button *className*="ctrl-btn" *onClick*={() => removeMovieFromWatched(*movie*.id)}>**

**<i *className*="fa-fw fa fa-times"></i>**

**</button>**

**- useContext lə funksiya nı götürüb buton a bağlayırıq.**

**- Artıq X ə klik edəndə watched dan silinir.**

**\* İncə Detallar Serisi.**

**- Watchlist boş olanda yoxdur desin qaqaş.**

**{**

**watchlist.length > 0 ? (**

**<div *className*="movie-grid">**

**{**

**watchlist.map( ( *movie* ) => (**

**<*MovieCard* *movie*={ *movie* } *key*={*movie*.id} *type*="watchlist" />**

**) )**

**}**

**</div>**

**) : (**

**<h2 *className*="no-movies">Listenizde film yok...</h2>**

**)**

**}**

**- Eynisini Watched üçün də edirik.**

**</h1>**

**<div *className*="count-pill">**

**{ watched.length } { watched.length < 2 ? "Movie" : "Movies" }**

**</div>**

**- Bu film sayını göstərir.**

**- Eynisini Watchlist üçün də yazırıq.**

**\* Qaldı son toxunuşlar.**

**- Watchlist buton u var Watched buton unu da əlavə edək.**

**ResultCart.jsx**

**- Watchlist buton unun eynisini copy edirik sadəcə işi dəyişəcək.**

**<div *className*="controls">**

**<button *disabled*={storedMovie} *className*="btn" *onClick*={ () => addMovieToWatchlist(*movie*) }>Add to Watchlist</button>**

**<button *disabled*={storedMovie} *className*="btn" *onClick*={ () => addMovieToWatched(*movie*) }>Add to Watched</button>**

**</div>**

**const { watchlist , addMovieToWatchlist , addMovieToWatched } = useContext(GlobalContext);**

**- Funksiya nı götürüb buton a bağlayırıq.**

**- Klik edəndə Watched a düşür.**

**\* Ama Watchlist ə klik edəndə Watched da disabled olur.**

**const { watchlist , watched , addMovieToWatchlist , addMovieToWatched } = useContext(GlobalContext);**

**const storedMovieWatched = watched.find( ( *f* ) => *f*.id === *movie*.id );**

**- watched ıda Global dan götürürük.**

**- Və stored buna da yaradırıq.**

**const storedMovie = watchlist.find( ( *f* ) => *f*.id === *movie*.id )**

**? true**

**:storedMovieWatched**

**? true**

**: false ;**

**- storedMovie true olarsa true alır false olarsa storedMovieWatched i.**

**- storedMovieWatched true olarsa true alır false olarsa false.**

**<div *className*="controls">**

**<button *disabled*={storedMovie} *className*="btn" *onClick*={ () => addMovieToWatchlist(*movie*) }>Add to Watchlist</button>**

**<button *disabled*={storedMovieWatched} *className*="btn" *onClick*={ () => addMovieToWatched(*movie*) }>Add to Watched</button>**

**</div>**

**- İndi oldu.**

**\* Saytın linkini kiməsə atanda işləmir çünli eyni Şəbəkədə olmaq lazımdır.**

**- Özümüz telle girsək açar çünki eyni Şəbəkə dəyik.**

**\* Ama pulsuz Global da paylaşa bilirik.**

**- package.json u açırıq.**

**- build var burda React Npm dir.**

**npm run build**

**- Yazırıq deployment üçün hazır olur proyekt.**

**- Build papka sı yaradır node un üstündə.**

**-** [**https://www.netlify.com/**](https://www.netlify.com/) **saytında JS və React proyekt lərimizi**

**- Pulsuz paylaşa bilirik. ( Hosting Şirkətidir )**

**- Yaranan Build papka sını atırıq bura.**

**- ...or deploy manually - - - olan yerə atırıq.**