src/app/ReduxProvider.tsx:

"use client";

import { Provider } from "react-redux";

import store from "src/store";

import { useEffect } from "react";

import { extendedApi } from "src/store/slices/configuration";

export default function ReduxProvider({ children }: { children: React.ReactNode }) {

useEffect(() => {

void (store.dispatch as any)(

extendedApi.endpoints.getConfiguration.initiate(undefined)

);

}, []);

return {children};

}

src/app/page.tsx

"use client";

import dynamic from "next/dynamic";

import { Github, Linkedin } from "lucide-react";

import StarBorder from "@/components/StarBorder";

import Link from "next/link";

// Arkaplandaki yıldızlar (daha önce oluşturduğumuz Stars.tsx)

const Stars = dynamic(() => import("@/components/Stars"), { ssr: false });

export default function Home() {

return (

className="relative min-h-screen overflow-hidden"

style={{

// SİYAH ZEMİN + KEDİ (GIF)

backgroundColor: "#000000",

backgroundImage: "url(/hero/nyan.gif)", // public/hero/nyan.gif

backgroundRepeat: "no-repeat",

backgroundPosition: "0 50%", // kedi SOL-ORTA

backgroundSize: "60%", // gerekirse 50–65% arası dene

}}

>

{/\* Yıldızlar (sağdan sola akış) \*/}

{/\* === (SPACER) Kedinin üzerindeki yazılar kaldırıldı; yerleşimi korumak için boş alan === \*/}

{/\* === Kedinin ALTINDA kalacak bölüm (buton + sosyal ikonlar) === \*/}

{/\* My Portfolio butonu \*/}

href="/portfolio" // hedefi istersen /portfolio yap

className="rounded-xl bg-white px-6 py-3 text-sm md:text-base font-medium text-zinc-900 shadow hover:shadow-md"

>

My Portfolio

{/\* Sosyal ikonlar \*/}

href="https://github.com/yagmurcemgul1"

target="\_blank"

rel="noopener noreferrer"

aria-label="GitHub"

className="text-white/85 hover:text-white transition"

>

href="https://www.linkedin.com/in/yagmurcemgul"

target="\_blank"

rel="noopener noreferrer"

aria-label="LinkedIn"

className="text-white/85 hover:text-white transition"

>

);

}

src/app/MuiThemeProvider.tsx

"use client";

import { useMemo } from "react";

import { ThemeProvider, createTheme } from "@mui/material/styles";

import palette from "src/theme/palette";

export default function MuiThemeProvider({ children }: { children: React.ReactNode }) {

const theme = useMemo(() => createTheme({ palette }), []);

return {children};

}

src/app/layout.tsx

import type { Metadata } from "next";

import "./globals.css";

import "@/components/PillNav.css";

import ReduxProvider from "src/app/ReduxProvider";

import "slick-carousel/slick/slick.css";

import "slick-carousel/slick/slick-theme.css";

import MUIRegistry from "src/theme/MUIRegistry";

import MuiThemeProvider from "./MuiThemeProvider";

/\* 👇 EKLE: Geist fontlarını next/font ile bağla \*/

import { Geist, Geist\_Mono } from "next/font/google";

const geistSans = Geist({ subsets: ["latin"], variable: "--font-geist-sans" });

const geistMono = Geist\_Mono({ subsets: ["latin"], variable: "--font-geist-mono" });

export const metadata: Metadata = {

title: "Welcome | Yagmur Cem Gul",

description: "Portfolio welcome page",

};

export default function RootLayout({ children }: { children: React.ReactNode }) {

return (

/\* 👇 EKLE: font değişken class’larını ’e bas \*/

{children}

);

}

src/app/globals.css

@import "tailwindcss";

:root {

--background: #ffffff;

--foreground: #171717;

/\* 👇 EKLE: Editor/LSP uyarılarını susturan fallback’ler \*/

--font-geist-sans: ui-sans-serif, system-ui, -apple-system, "Segoe UI",

Roboto, Arial, "Noto Sans", "Liberation Sans", sans-serif;

--font-geist-mono: ui-monospace, SFMono-Regular, Menlo, Monaco, Consolas,

"Liberation Mono", "Courier New", monospace;

}

html, body {

min-height: 100%;

/\* background: #141414; \*/

color-scheme: dark;

}

@theme inline {

--color-background: var(--background);

--color-foreground: var(--foreground);

--font-sans: var(--font-geist-sans);

--font-mono: var(--font-geist-mono);

}

@media (prefers-color-scheme: dark) {

:root {

/\* --background: #0a0a0a; \*/

--foreground: #ededed;

}

}

body {

/\* background: var(--background); \*/

color: var(--foreground);

font-family: var(--font-geist-sans);

}

/\* === NAV THEME (tek noktadan yönetim) === \*/

:root {

/\* Renkler \*/

--nav-base: #000000; /\* nav kapsülü ve logo zemini \*/

--nav-pill-bg: #ffffff; /\* buton/kapsül zemini \*/

--nav-pill-text: #000000; /\* kapsül metni \*/

--nav-hover-text: #ffffff; /\* hover anındaki metin \*/

/\* Boyutlar \*/

--nav-height: 42px;

--nav-pill-pad-x: 18px;

--nav-gap: 3px;

/\* Yazı tipi \*/

--nav-font-family: var(--font-sans, Arial, Helvetica, sans-serif);

--nav-font-size: 16px;

--nav-font-weight: 600;

}

/\* İsteğe bağlı: koyu/açık tema varyantı \*/

[data-theme="dark"] {

--nav-base: #000000;

--nav-pill-bg: #ffffff;

--nav-pill-text: #000000;

--nav-hover-text: #ffffff;

}

[data-theme="light"] {

--nav-base: #ffffff;

--nav-pill-bg: #0a0a0a;

--nav-pill-text: #ffffff;

--nav-hover-text: #ffffff;

}

/\* PillNav.css’in beklediği isimlere map et (değişiklik gerektirmeden çalışsın) \*/

:root {

--base: var(--nav-base);

--pill-bg: var(--nav-pill-bg);

--pill-text: var(--nav-pill-text);

--hover-text: var(--nav-hover-text);

--nav-h: var(--nav-height);

--pill-pad-x: var(--nav-pill-pad-x);

--pill-gap: var(--nav-gap);

}

html, body { background-color: #141414 !important; }

src/app/projects/layout.tsx

"use client";

import Box from "@mui/material/Box";

import DetailModal from "src/components/DetailModal";

import VideoPortalContainer from "src/components/VideoPortalContainer";

import DetailModalProvider from "src/providers/DetailModalProvider";

import PortalProvider from "src/providers/PortalProvider";

import { MAIN\_PATH } from "src/constant";

import { Footer, MainHeader } from "src/components/layouts";

// import MainLoadingScreen from "src/components/MainLoadingScreen"; // Next.js'te loading.tsx kullanılır

import { usePathname } from "next/navigation";

export default function MainLayout({

children,

}: {

children: React.ReactNode;

}) {

const pathname = usePathname();

return (

sx={{

width: "100%",

minHeight: "100vh",

bgcolor: "background.default",

position: "relative", // portal/overlay için güvenli

}}

>

{/\* Header tüm sayfalarda görünsün \*/}

{/\* React Router navigation durumu YOK. Next için app/.../loading.tsx kullanılır \*/}

{/\* \*/}

{/\* Modal ve mini-portal provider hiyerarşisi \*/}

{children}

{/\* Watch sayfası dışında footer \*/}

{pathname !== `/${MAIN\_PATH.watch}` && }

);

}

src/app/projects/page.tsx

"use client";

import Stack from "@mui/material/Stack";

import { COMMON\_TITLES } from "src/constant";

import HeroSection from "src/components/HeroSection";

import { genreSliceEndpoints, useGetGenresQuery } from "src/store/slices/genre";

import { MEDIA\_TYPE } from "src/types/Common";

import { CustomGenre, Genre } from "src/types/Genre";

import SliderRowForGenre from "src/components/VideoSlider";

import store from "src/store";

import DetailModalProvider from "src/providers/DetailModalProvider";

import PortalProvider from "src/providers/PortalProvider";

import VideoPortalContainer from "src/components/VideoPortalContainer";

export default function Page() {

const { data: genres, isLoading, isError, error } = useGetGenresQuery(MEDIA\_TYPE.Movie);

const list = [...COMMON\_TITLES, ...(genres ?? [])] as (Genre | CustomGenre)[];

return (

{isError && (

Genres could not be loaded.

{"\n"}

{JSON.stringify(error, null, 2)}

)}

{list.map((genre: Genre | CustomGenre) => (

key={genre.id || genre.name}

genre={genre}

mediaType={MEDIA\_TYPE.Movie}

/>

))}

{isLoading &&

Loading…

}

{/\* Hover mini modal buradan çizilecek \*/}

);

}

src/app/portfolio/page.tsx

"use client";

import React, { useEffect, useMemo, useState } from "react";

import PillNav from "@/components/PillNav";

const SECTIONS = ['home', 'about', 'projects', 'blog'] as const;

type TabId = typeof SECTIONS[number];

export default function PortfolioPage() {

// --- 🔧 Hydration fix: sadece mount olduktan sonra rastgele yıldızları üret ---

const [mounted, setMounted] = useState(false);

useEffect(() => setMounted(true), []);

const [activeTab, setActiveTab] = useState('home');

// Scroll spy için active section'ı takip et

useEffect(() => {

const handleScroll = () => {

const scrollY = window.scrollY;

const others: TabId[] = SECTIONS.slice(1) as TabId[];

// Home section için özel kontrol (scroll top'ta)

if (scrollY < 100) {

setActiveTab('home');

return;

}

// Diğer section'ları kontrol et

for (const section of others) {

const element = document.getElementById(section);

if (element) {

const rect = element.getBoundingClientRect();

if (rect.top <= 100 && rect.bottom >= 100) {

setActiveTab(section);

break;

}

}

}

};

window.addEventListener('scroll', handleScroll);

return () => window.removeEventListener('scroll', handleScroll);

}, []);

const stars = useMemo(() => {

if (!mounted) return []; // SSR aşamasında yıldız render etme (mismatch çözümü)

return Array.from({ length: 200 }).map((\_, i) => {

const size = Math.floor(Math.random() \* 7) + 3; // 3..9 px

const dur = Math.floor(Math.random() \* 30) + 15; // 15..45 s

const delay = -Math.floor(Math.random() \* 40); // -40..0 s

const top = Math.floor(Math.random() \* 102) - 1; // -1..100 vh

return (

key={i}

className="star"

style={{

width: `${size}px`,

height: `${size}px`,

top: `${top}vh`,

animationDuration: `${dur}s`,

animationDelay: `${delay}s`,

}}

/>

);

});

}, [mounted]);

// — İKİ SATIR — //

const line1 = "Yagmur Cem Gul";

const line2 =

"Project Manager | Digital Marketing | Game Developer | Creative Producer";

// ← YAZININ BAŞLAMASI İÇİN GENEL DELAY (saniye)

const TEXT\_START\_DELAY\_S = 4; // örnek: 4s sonra yazmaya başla

const STEP = 0.05; // harf başına gecikme (sn) — istersen 0.06/0.08 yap

const makeLetters = (str: string, baseDelay = 0) => {

const chars = str.split("");

return chars.map((ch, idx) => {

const isSpace = ch === " ";

const delay = baseDelay + idx \* STEP; // doğrudan sayıyla hesapla

return (

key={`${str}-${idx}`}

className={isSpace ? "w" : undefined}

style={{ animationDelay: `${delay}s` }} // 👈 inline delay

>

{ch}

);

});

};

const lettersLine1 = useMemo(() => makeLetters(line1, TEXT\_START\_DELAY\_S), [TEXT\_START\_DELAY\_S]);

const lettersLine2 = useMemo(() => makeLetters(line2, TEXT\_START\_DELAY\_S), [TEXT\_START\_DELAY\_S]);

return (

logo="/logo.png"

logoAlt="YCG"

items={[

{ label: "Welcome", href: "/" },

{ label: "About", href: "/about" },

{ label: "Portfolio", href: "/portfolio" },

{ label: "Projects",href: "/projects" },

{ label: "Blog", href: "/blog" },

]}

activeHref={activeTab === "home" ? "/" : `#${activeTab}`}

baseColor="#000000"

pillColor="#ffffff"

hoveredPillTextColor="#ffffff"

pillTextColor="#000000"

/>

{/\* HERO: yıldızlar sadece burada \*/}

{/\* Yıldız katmanı \*/}

{stars}

{/\* Yazı katmanı \*/}

{lettersLine1}

{lettersLine2}

{/\* Crewmate \*/}

className="crewmate"

src="https://particles.js.org/images/amongus\_cyan.png"

alt=""

width={207}

height={265}

/>

{/\* SVG filter \*/}

{/\* PROJECTS: yıldızlı arka plan YOK \*/}

Projects

Öne çıkan projelerim.

Project One

Kısa açıklama…

Project Two

Kısa açıklama…

Project Three

Kısa açıklama…

{/\* İstersen About/Blog bölümlerini de aşağıya koyarsın \*/}

);

}

src/components/animate/variants/fade/FadeIn.ts

const TRANSITION\_ENTER = {

duration: 0.64,

ease: [0.43, 0.13, 0.23, 0.96] as const,

};

const TRANSITION\_EXIT = {

duration: 0.48,

ease: [0.43, 0.13, 0.23, 0.96] as const,

};

export const varFadeIn = {

initial: { opacity: 0 },

animate: { opacity: 1, transition: TRANSITION\_ENTER },

exit: { opacity: 0, transition: TRANSITION\_EXIT },

};

src/components/animate/variants/fade/FadeOut.ts

const TRANSITION\_ENTER = {

duration: 0.64,

ease: [0.43, 0.13, 0.23, 0.96],

};

const TRANSITION\_EXIT = {

duration: 0.48,

ease: [0.43, 0.13, 0.23, 0.96],

};

export const varFadeOut = {

initial: { opacity: 1 },

animate: { opacity: 0, transition: TRANSITION\_ENTER },

exit: { opacity: 1, transition: TRANSITION\_EXIT },

};

src/components/animate/variants/zoom/ZoomIn.ts

import type { Variants, Transition } from "framer-motion";

const DISTANCE = 0;

const IN = { scale: 1, opacity: 1 };

const OUT = { scale: 0, opacity: 0 };

const TRANSITION\_ENTER: Transition = {

duration: 1,

ease: [0.43, 0.13, 0.23, 0.96] as const,

};

const TRANSITION\_EXIT: Transition = {

duration: 1,

ease: [0.43, 0.13, 0.23, 0.96] as const,

};

export const varZoomIn: Variants = {

initial: OUT,

animate: { ...IN, transition: TRANSITION\_ENTER },

exit: { ...OUT, transition: TRANSITION\_EXIT },

};

export const varZoomInLeft: Variants = {

initial: { ...OUT, translateX: -DISTANCE },

animate: { ...IN, translateX: 0, transition: TRANSITION\_ENTER },

exit: { ...OUT, translateX: -DISTANCE, transition: TRANSITION\_EXIT },

};

export const varZoomInRight: Variants = {

initial: { ...OUT, translateX: DISTANCE },

animate: { ...IN, translateX: 0, transition: TRANSITION\_ENTER },

exit: { ...OUT, translateX: DISTANCE, transition: TRANSITION\_EXIT },

};

src/components/animate/variants/Wrap.ts

export const varWrapEnter = {

animate: {

transition: { staggerChildren: 0.1 },

},

};

export const varWrapExit = {

exit: {

transition: { staggerChildren: 0.1 },

},

};

export const varWrapBoth = {

animate: {

transition: { staggerChildren: 0.07, delayChildren: 0.1 },

},

exit: {

transition: { staggerChildren: 0.05, staggerDirection: -1 },

},

};

src/components/animates/MotionContainer.tsx

"use client";

import { motion } from "framer-motion";

import Box, { BoxProps } from "@mui/material/Box";

import { varWrapBoth } from "./variants/Wrap";

interface MotionContainerProps extends BoxProps {

initial?: boolean | string;

open?: boolean;

}

export default function MotionContainer({

open,

children,

...other

}: MotionContainerProps) {

return (

initial={false}

variants={varWrapBoth}

component={motion.div}

animate={open ? "animate" : "exit"}

{...other}

>

{children}

);

}

src/components/layouts/Footer.tsx

"use client";

import Box from "@mui/material/Box";

import Link from "@mui/material/Link";

import Typography from "@mui/material/Typography";

import Divider from "@mui/material/Divider";

export default function Footer() {

return (

component="footer"

sx={{

display: "flex",

flexDirection: "column",

justifyContent: "center",

height: 150,

bgcolor: "inherit",

px: "60px",

}}

>

component="div"

sx={{

"::before, ::after": { top: "0%" },

}}

>

Developed by{" "}

target="\_blank"

underline="none"

sx={{ color: "text.primary" }}

href="https://github.com/yagmurcemgul"

>

Yağmur Cem Gül

);

}

src/components/layouts/index.ts

"use client";

export { default as MainHeader } from "./MainHeader";

export { default as Footer } from "./Footer";

src/components/layouts/MainHeader.tsx

"use client";

import \* as React from "react";

import AppBar from "@mui/material/AppBar";

import Box from "@mui/material/Box";

import Stack from "@mui/material/Stack";

import Toolbar from "@mui/material/Toolbar";

import IconButton from "@mui/material/IconButton";

import Typography from "@mui/material/Typography";

import Menu from "@mui/material/Menu";

import MenuIcon from "@mui/icons-material/Menu";

import Avatar from "@mui/material/Avatar";

import Tooltip from "@mui/material/Tooltip";

import MenuItem from "@mui/material/MenuItem";

import useOffSetTop from "src/hooks/useOffSetTop";

import { APP\_BAR\_HEIGHT } from "src/constant";

import Logo from "../Logo";

import SearchBox from "../SearchBox";

import NetflixNavigationLink from "../NetflixNavigationLink";

const pages = ["My List", "Movies", "Tv Shows"];

const MainHeader = () => {

const isOffset = useOffSetTop(APP\_BAR\_HEIGHT);

const [anchorElNav, setAnchorElNav] = React.useState(

null

);

const [anchorElUser, setAnchorElUser] = React.useState(

null

);

const handleOpenNavMenu = (event: React.MouseEvent) => {

setAnchorElNav(event.currentTarget);

};

const handleOpenUserMenu = (event: React.MouseEvent) => {

setAnchorElUser(event.currentTarget);

};

const handleCloseNavMenu = () => {

setAnchorElNav(null);

};

const handleCloseUserMenu = () => {

setAnchorElUser(null);

};

return (

sx={{

// px: "4%",

px: "60px",

height: APP\_BAR\_HEIGHT,

backgroundImage: "none",

...(isOffset

? {

bgcolor: "primary.main",

boxShadow: (theme) => theme.shadows[4],

}

: { boxShadow: 0, bgcolor: "transparent" }),

}}

>

size="large"

aria-label="account of current user"

aria-controls="menu-appbar"

aria-haspopup="true"

onClick={handleOpenNavMenu}

color="inherit"

>

id="menu-appbar"

anchorEl={anchorElNav}

anchorOrigin={{

vertical: "bottom",

horizontal: "left",

}}

keepMounted

transformOrigin={{

vertical: "top",

horizontal: "left",

}}

open={Boolean(anchorElNav)}

onClose={handleCloseNavMenu}

sx={{

display: { xs: "block", md: "none" },

}}

>

{pages.map((page) => (

{page}

))}

variant="h5"

noWrap

component="a"

href=""

sx={{

mr: 2,

display: { xs: "flex", md: "none" },

flexGrow: 1,

fontWeight: 700,

color: "inherit",

textDecoration: "none",

}}

>

Netflix

direction="row"

spacing={3}

sx={{ flexGrow: 1, display: { xs: "none", md: "flex" } }}

>

{pages.map((page) => (

href="/"

variant="subtitle1"

key={page}

onClick={handleCloseNavMenu}

>

{page}

))}

sx={{ mt: "45px" }}

id="avatar-menu"

anchorEl={anchorElUser}

anchorOrigin={{

vertical: "top",

horizontal: "right",

}}

keepMounted

transformOrigin={{

vertical: "top",

horizontal: "right",

}}

open={Boolean(anchorElUser)}

onClose={handleCloseUserMenu}

>

{["Account", "Logout"].map((setting) => (

{setting}

))}

);

};

export default MainHeader;

src/components/slick-slider/CustomNavigation.tsx

"use client";

import { styled } from "@mui/material/styles";

import Box from "@mui/material/Box";

import ArrowBackIosNewIcon from "@mui/icons-material/ArrowBackIosNew";

import ArrowForwardIosIcon from "@mui/icons-material/ArrowForwardIos";

import { MouseEventHandler, ReactNode } from "react";

const ArrowStyle = styled(Box)(({ theme }) => ({

top: 0,

bottom: 0,

position: "absolute",

zIndex: 9,

height: "100%",

opacity: 0.48,

display: "flex",

cursor: "pointer",

alignItems: "center",

justifyContent: "center",

color: theme.palette.common.white,

// background: theme.palette.grey[700],

transition: theme.transitions.create("opacity"),

"&:hover": {

opacity: 0.8,

background: theme.palette.grey[900],

},

[theme.breakpoints.down("sm")]: {

display: "none",

},

}));

interface CustomNaviationProps {

isEnd: boolean;

arrowWidth: number;

children: ReactNode;

activeSlideIndex: number;

onNext: MouseEventHandler;

onPrevious: MouseEventHandler;

}

export default function CustomNavigation({

isEnd,

onNext,

children,

onPrevious,

arrowWidth,

activeSlideIndex,

}: CustomNaviationProps) {

return (

<>

{activeSlideIndex > 0 && (

onClick={onPrevious}

sx={{

left: 0,

width: { xs: arrowWidth / 2, sm: arrowWidth },

borderTopRightRadius: { xs: "4px" },

borderBottomRightRadius: { xs: "4px" },

// backgroundImage: (theme) =>

// `linear-gradient(to right, ${theme.palette.background.default} 0%, rgba(0,0,0,0) 100%)`,

}}

>

)}

{children}

{!isEnd && (

onClick={onNext}

sx={{

right: 0,

width: { xs: arrowWidth / 2, sm: arrowWidth },

borderTopLeftRadius: { xs: "4px" },

borderBottomLeftRadius: { xs: "4px" },

// backgroundImage: (theme) =>

// `linear-gradient(to left, ${theme.palette.background.default} 0%, rgba(0,0,0,0) 100%)`,

}}

>

)}

);

}

src/components/slick-slider/SlickSlider.tsx

"use client";

import { useState, useRef } from "react";

import Slider, { Settings } from "react-slick";

import { motion } from "framer-motion";

import { styled, Theme, useTheme } from "@mui/material/styles";

import Box from "@mui/material/Box";

import Stack from "@mui/material/Stack";

import CustomNavigation from "./CustomNavigation";

import VideoItemWithHover from "src/components/VideoItemWithHover";

import { ARROW\_MAX\_WIDTH } from "src/constant";

import NetflixNavigationLink from "src/components/NetflixNavigationLink";

import MotionContainer from "src/components/animate/MotionContainer";

import { varFadeIn } from "src/components/animate/variants/fade/FadeIn";

import { CustomGenre, Genre } from "src/types/Genre";

import { Movie } from "src/types/Movie";

import { PaginatedMovieResult } from "src/types/Common";

const RootStyle = styled("div")(() => ({

position: "relative",

overflow: "inherit",

}));

const StyledSlider = styled(Slider)(

({ theme, padding }: { theme: Theme; padding: number }) => ({

display: "flex !important",

justifyContent: "center",

overflow: "initial !important",

"& > .slick-list": {

overflow: "visible",

},

[theme.breakpoints.up("sm")]: {

"& > .slick-list": {

width: `calc(100% - ${2 \* padding}px)`,

},

"& .slick-list > .slick-track": {

margin: "0px !important",

},

"& .slick-list > .slick-track > .slick-current > div > .NetflixBox-root > .NetflixPaper-root:hover":

{

transformOrigin: "0% 50% !important",

},

},

[theme.breakpoints.down("sm")]: {

"& > .slick-list": {

width: `calc(100% - ${padding}px)`,

},

},

})

);

interface SlideItemProps {

item: Movie;

}

function SlideItem({ item }: SlideItemProps) {

return (

);

}

interface SlickSliderProps {

data: PaginatedMovieResult;

genre: Genre | CustomGenre;

handleNext: (page: number) => void;

}

export default function SlickSlider({ data, genre }: SlickSliderProps) {

const sliderRef = useRef(null);

const [activeSlideIndex, setActiveSlideIndex] = useState(0);

const [showExplore, setShowExplore] = useState(false);

const [isEnd, setIsEnd] = useState(false);

const theme = useTheme();

const beforeChange = async (currentIndex: number, nextIndex: number) => {

if (currentIndex < nextIndex) {

setActiveSlideIndex(nextIndex);

} else if (currentIndex > nextIndex) {

setIsEnd(false);

}

setActiveSlideIndex(nextIndex);

};

const settings: Settings = {

speed: 500,

arrows: false,

infinite: false,

lazyLoad: "ondemand",

slidesToShow: 6,

slidesToScroll: 6,

// afterChange: (current) => {

// console.log("After Change", current);

// },

beforeChange,

// onEdge: (direction) => {

// console.log("Edge: ", direction);

// },

responsive: [

{

breakpoint: 1536,

settings: {

slidesToShow: 5,

slidesToScroll: 5,

},

},

{

breakpoint: 1200,

settings: {

slidesToShow: 4,

slidesToScroll: 4,

},

},

{

breakpoint: 900,

settings: {

slidesToShow: 3,

slidesToScroll: 3,

},

},

{

breakpoint: 600,

settings: {

slidesToShow: 2,

slidesToScroll: 2,

},

},

],

};

const handlePrevious = () => {

sliderRef.current?.slickPrev();

};

const handleNext = () => {

sliderRef.current?.slickNext();

};

return (

{data.results.length > 0 && (

<>

spacing={2}

direction="row"

alignItems="center"

sx={{ mb: 2, pl: { xs: "30px", sm: "60px" } }}

>

variant="h5"

href={`/genre/${ // <-- href

genre.id || genre.name.toLowerCase().replace(" ", "\_")

}`}

sx={{

display: "inline-block",

fontWeight: 700,

}}

onMouseOver={() => {

setShowExplore(true);

}}

onMouseLeave={() => {

setShowExplore(false);

}}

>

{`${genre.name} Movies `}

open={showExplore}

initial="initial"

sx={{ display: "inline", color: "success.main" }}

>

{"Explore All".split("").map((letter, index) => (

{letter}

))}

isEnd={isEnd}

arrowWidth={ARROW\_MAX\_WIDTH}

onNext={handleNext}

onPrevious={handlePrevious}

activeSlideIndex={activeSlideIndex}

>

ref={sliderRef}

{...settings}

padding={ARROW\_MAX\_WIDTH}

theme={theme}

>

{data.results

.filter((i) => !!i.backdrop\_path)

.map((item) => (

))}

)}

);

}

src/components/watch/PlayerControlButton.tsx

import { forwardRef } from "react";

import IconButton, { IconButtonProps } from "@mui/material/IconButton";

const PlayerControlButton = forwardRef(

({ children, ...others }, ref) => (

ref={ref}

sx={{

padding: { xs: 0.5, sm: 1 },

"& svg, & span": { transition: "transform .3s" },

"&:hover svg, &:hover span": {

msTransform: "scale(1.3)",

WebkitTransform: "scale(1.3)",

transform: "scale(1.3)",

},

}}

{...others}

>

{children}

)

);

export default PlayerControlButton;

src/components/watch/PlayerSeekbar.tsx

import Slider from "@mui/material/Slider";

import { styled } from "@mui/material/styles";

import { formatTime } from "src/utils/common";

const StyledSlider = styled(Slider)({

borderRadius: 0,

"& .NetflixSlider-track": {

backgroundColor: "red !important",

border: 0,

},

"& .NetflixSlider-rail": {

border: "none",

backgroundColor: "white !important",

opacity: 0.85,

},

"& .NetflixSlider-thumb": {

borderRadius: "50%",

height: 10,

width: 10,

backgroundColor: "red",

"&:focus, &:hover, &.Netflix-active, &.Netflix-focusVisible": {

boxShadow: "inherit",

height: 15,

width: 15,

},

"&:before": {

display: "none",

boxShadow: "0 2px 2px 0 #fff",

height: 10,

width: 10,

},

},

// "& .NetflixSlider-valueLabel": {

// lineHeight: 1.2,

// fontSize: 12,

// background: "unset",

// padding: 0,

// width: 32,

// height: 32,

// borderRadius: "50% 50% 50% 0",

// backgroundColor: "#52af77",

// transformOrigin: "bottom left",

// transform: "translate(50%, -100%) rotate(-45deg) scale(0)",

// "&:before": { display: "none" },

// "&.NetflixSlider-valueLabelOpen": {

// transform: "translate(50%, -100%) rotate(-45deg) scale(1)",

// },

// "& > \*": {

// transform: "rotate(45deg)",

// },

// },

});

function PlayerSeekbar({

playedSeconds,

duration,

seekTo,

}: {

playedSeconds: number;

duration: number;

seekTo: (value: number) => void;

}) {

return (

valueLabelDisplay="auto"

valueLabelFormat={(v) => formatTime(v)}

// components={{

// ValueLabel: ValueLabelComponent,

// }}

value={playedSeconds}

max={duration}

onChange={(\_, value) => {

seekTo(value as number);

}}

/>

);

}

export default PlayerSeekbar;

src/components/watch/VideoJSPlayer.tsx

import { useEffect, useRef } from "react";

import Player from "video.js/dist/types/player";

import videojs from "video.js";

import "videojs-youtube";

import "video.js/dist/video-js.css";

export default function VideoJSPlayer({

options,

onReady,

}: {

options: any;

onReady: (player: Player) => void;

}) {

const videoRef = useRef(null);

const playerRef = useRef(null);

useEffect(() => {

(async function handleVideojs() {

// Make sure Video.js player is only initialized once

if (!playerRef.current) {

// The Video.js player needs to be \_inside\_ the component el for React 18 Strict Mode.

const videoElement = document.createElement("video-js");

// videoElement.classList.add("vjs-big-play-centered", "vjs-16-9");

videoRef.current?.appendChild(videoElement);

const player = (playerRef.current = videojs(

videoElement,

options,

() => {

onReady && onReady(player);

}

));

// import("video.js").then(async ({ default: videojs }) => {

// await import("video.js/dist/video-js.css");

// if (options["techOrder"] && options["techOrder"].includes("youtube")) {

// // eslint-disable-next-line

// await import("videojs-youtube");

// }

// const player = (playerRef.current = videojs(

// videoElement,

// options,

// () => {

// onReady && onReady(player);

// }

// ));

// });

// await import("video.js/dist/video-js.css");

// const videojs = await import("video.js");

// if (options["techOrder"] && options["techOrder"].includes("youtube")) {

// // eslint-disable-next-line

// await import("videojs-youtube");

// }

// const player = (playerRef.current = videojs.default(

// videoElement,

// options,

// () => {

// onReady && onReady(player);

// }

// ));

// You could update an existing player in the `else` block here

// on prop change, for example:

} else {

const player = playerRef.current;

// player.autoplay(options.autoplay);

player.width(options.width);

player.height(options.height);

}

})();

}, [options, videoRef]);

// Dispose the Video.js player when the functional component unmounts

useEffect(() => {

const player = playerRef.current;

return () => {

if (player && !player.isDisposed()) {

player.dispose();

playerRef.current = null;

}

};

}, [playerRef]);

return (

);

}

src/components/watch/VolumeControllers.tsx

import { Stack } from "@mui/material";

import Slider from "@mui/material/Slider";

import { styled } from "@mui/material/styles";

import { SliderUnstyledOwnProps } from "@mui/base/SliderUnstyled";

import VolumeUpIcon from "@mui/icons-material/VolumeUp";

import VolumeOffIcon from "@mui/icons-material/VolumeOff";

import PlayerControlButton from "./PlayerControlButton";

const StyledSlider = styled(Slider)({

height: 5,

borderRadius: 0,

padding: 0,

"& .NetflixSlider-track": {

border: "none",

backgroundColor: "red",

},

"& .NetflixSlider-rail": {

border: "none",

backgroundColor: "white",

opacity: 0.85,

},

"& .NetflixSlider-thumb": {

height: 10,

width: 10,

backgroundColor: "red",

"&:focus, &:hover, &.Netflix-active, &.Netflix-focusVisible": {

boxShadow: "inherit",

height: 15,

width: 15,

},

"&:before": {

display: "none",

},

},

});

export default function VolumeControllers({

value,

handleVolume,

handleVolumeToggle,

muted,

}: {

value: number;

handleVolume: SliderUnstyledOwnProps["onChange"];

handleVolumeToggle: React.MouseEventHandler;

muted: boolean;

}) {

return (

direction="row"

alignItems="center"

spacing={{ xs: 0.5, sm: 1 }}

// sx={{

// "&:hover NetflixSlider-root": {

// display: "inline-block",

// },

// }}

>

{!muted ? : }

max={100}

value={value \* 100}

valueLabelDisplay="auto"

valueLabelFormat={(x: number) => x}

onChange={handleVolume}

sx={{ width: { xs: 60, sm: 80, md: 100 } }}

/>

);

}

src/components/AgeLimitChip.tsx

"use client";

import Chip, { ChipProps } from "@mui/material/Chip";

export default function AgeLimitChip({ sx, ...others }: ChipProps) {

return (

{...others}

sx={{

borderRadius: 0,

p: 0.5,

fontSize: 12,

height: "100%",

"& > span": { p: 0 },

...sx,

}}

variant="outlined"

/>

);

}

src/components/DetailModal.tsx

"use client";

import { forwardRef, useCallback, useRef, useState, useEffect } from "react";

import Box from "@mui/material/Box";

import Grid from "@mui/material/Grid";

import Container from "@mui/material/Container";

import Stack from "@mui/material/Stack";

import IconButton from "@mui/material/IconButton";

import Typography from "@mui/material/Typography";

import Dialog from "@mui/material/Dialog";

import DialogContent from "@mui/material/DialogContent";

import Slide from "@mui/material/Slide";

import { TransitionProps } from "@mui/material/transitions";

import CloseIcon from "@mui/icons-material/Close";

import AddIcon from "@mui/icons-material/Add";

import ThumbUpOffAltIcon from "@mui/icons-material/ThumbUpOffAlt";

import VolumeUpIcon from "@mui/icons-material/VolumeUp";

import VolumeOffIcon from "@mui/icons-material/VolumeOff";

import Player from "video.js/dist/types/player";

import MaxLineTypography from "./MaxLineTypography";

import PlayButton from "./PlayButton";

import NetflixIconButton from "./NetflixIconButton";

import AgeLimitChip from "./AgeLimitChip";

import QualityChip from "./QualityChip";

import { formatMinuteToReadable, getRandomNumber } from "src/utils/common";

import SimilarVideoCard from "./SimilarVideoCard";

import { useDetailModal } from "src/providers/DetailModalProvider";

import { useGetSimilarVideosQuery } from "src/store/slices/discover";

import { MEDIA\_TYPE } from "src/types/Common";

import VideoJSPlayer from "./watch/VideoJSPlayer";

const Transition = forwardRef(function Transition(

props: TransitionProps & {

children: React.ReactElement;

},

ref: React.Ref

) {

return ;

});

export default function DetailModal() {

const { detail, setDetailType } = useDetailModal();

const { data: similarVideos } = useGetSimilarVideosQuery(

{ mediaType: detail.mediaType ?? MEDIA\_TYPE.Movie, id: detail.id ?? 0 },

{ skip: !detail.id }

);

const playerRef = useRef(null);

const [muted, setMuted] = useState(true);

const [uiNums, setUiNums] = useState({ match: 0, age: 0, mins: 0 });

useEffect(() => {

if (detail.mediaDetail?.id) {

setUiNums({

match: getRandomNumber(100),

age: getRandomNumber(20),

mins: getRandomNumber(180),

});

}

}, [detail.mediaDetail?.id]);

const handleReady = useCallback((player: Player) => {

playerRef.current = player;

setMuted(!!player.muted()); // <-- !!

}, []);

const handleMute = useCallback((status: boolean) => {

if (playerRef.current) {

playerRef.current.muted(!status);

setMuted(!status);

}

}, []);

if (detail.mediaDetail) {

return (

fullWidth

scroll="body"

maxWidth="md"

open={!!detail.mediaDetail}

id="detail\_dialog"

TransitionComponent={Transition}

>

sx={{

top: 0,

left: 0,

right: 0,

position: "relative",

mb: 3,

}}

>

sx={{

width: "100%",

position: "relative",

height: "calc(9 / 16 \* 100%)",

}}

>

options={{

loop: true,

autoplay: true,

controls: false,

responsive: true,

fluid: true,

techOrder: ["youtube"],

sources: [

{

type: "video/youtube",

src: `https://www.youtube.com/watch?v=${

detail.mediaDetail?.videos.results[0]?.key ||

"L3oOldViIgY"

}`,

},

],

}}

onReady={handleReady}

/>

sx={{

background: `linear-gradient(77deg,rgba(0,0,0,.6),transparent 85%)`,

top: 0,

left: 0,

bottom: 0,

right: "26.09%",

opacity: 1,

position: "absolute",

transition: "opacity .5s",

}}

/>

sx={{

backgroundColor: "transparent",

backgroundImage:

"linear-gradient(180deg,hsla(0,0%,8%,0) 0,hsla(0,0%,8%,.15) 15%,hsla(0,0%,8%,.35) 29%,hsla(0,0%,8%,.58) 44%,#141414 68%,#141414)",

backgroundRepeat: "repeat-x",

backgroundPosition: "0px top",

backgroundSize: "100% 100%",

bottom: 0,

position: "absolute",

height: "14.7vw",

opacity: 1,

top: "auto",

width: "100%",

}}

/>

onClick={() => {

setDetailType({ mediaType: undefined, id: undefined });

}}

sx={{

top: 15,

right: 15,

position: "absolute",

bgcolor: "#181818",

width: { xs: 22, sm: 40 },

height: { xs: 22, sm: 40 },

"&:hover": {

bgcolor: "primary.main",

},

}}

>

sx={{ color: "white", fontSize: { xs: 14, sm: 22 } }}

/>

sx={{

position: "absolute",

left: 0,

right: 0,

bottom: 16,

px: { xs: 2, sm: 3, md: 5 },

}}

>

{detail.mediaDetail?.title}

size="large"

onClick={() => handleMute(muted)}

sx={{ zIndex: 2 }}

>

{!muted ? : }

sx={{

p: "0px !important",

}}

>

variant="subtitle1"

sx={{ color: "success.main" }}

>{`${uiNums.match}% Match`}

{detail.mediaDetail?.release\_date.substring(0, 4)}

{formatMinuteToReadable(uiNums.mins)}

maxLine={3}

variant="body1"

sx={{ mt: 2 }}

>

{detail.mediaDetail?.overview}

{`Genres : ${detail.mediaDetail?.genres

.map((g) => g.name)

.join(", ")}`}

{`Available in : ${detail.mediaDetail?.spoken\_languages

.map((l) => l.name)

.join(", ")}`}

{similarVideos && similarVideos.results.length > 0 && (

sx={{

py: 2,

px: { xs: 2, sm: 3, md: 5 },

}}

>

More Like This

{similarVideos.results.map((sm) => (

))}

)}

);

}

return null;

}

src/components/GenreBreadcrumbs.tsx

"use client";

import Box from "@mui/material/Box";

import Typography from "@mui/material/Typography";

import Breadcrumbs, { BreadcrumbsProps } from "@mui/material/Breadcrumbs";

const Separator = (

component="span"

sx={{

width: 4,

height: 4,

borderRadius: "50%",

bgcolor: "text.disabled",

}}

/>

);

interface GenreBreadcrumbsProps extends BreadcrumbsProps {

genres: string[];

}

export default function GenreBreadcrumbs({

genres,

...others

}: GenreBreadcrumbsProps) {

return (

{genres.map((genre, idx) => (

{genre}

))}

);

}

src/components/GridPage.tsx

"use client";

import withPagination from "src/hoc/withPagination";

import { MEDIA\_TYPE } from "src/types/Common";

import { CustomGenre, Genre } from "src/types/Genre";

import GridWithInfiniteScroll from "./GridWithInfiniteScroll";

interface GridPageProps {

genre: Genre | CustomGenre;

mediaType: MEDIA\_TYPE;

}

export default function GridPage({ genre, mediaType }: GridPageProps) {

const Component = withPagination(

GridWithInfiniteScroll,

mediaType,

genre

);

return ;

}

src/components/GridWithInfiniteScroll.tsx

"use client";

import { useRef, useEffect } from "react";

import Grid from "@mui/material/Grid";

import Box from "@mui/material/Box";

import Container from "@mui/material/Container";

import Typography from "@mui/material/Typography";

import VideoItemWithHover from "./VideoItemWithHover";

import { CustomGenre, Genre } from "src/types/Genre";

import { PaginatedMovieResult } from "src/types/Common";

import useIntersectionObserver from "src/hooks/useIntersectionObserver";

interface GridWithInfiniteScrollProps {

genre: Genre | CustomGenre;

data: PaginatedMovieResult;

handleNext: (page: number) => void;

}

export default function GridWithInfiniteScroll({

genre,

data,

handleNext,

}: GridWithInfiniteScrollProps) {

const intersectionRef = useRef(null);

const intersection = useIntersectionObserver(intersectionRef);

useEffect(() => {

if (

intersection &&

intersection.intersectionRatio === 1 &&

data.page < data.total\_pages

) {

handleNext(data.page + 1);

}

}, [intersection]);

return (

<>

maxWidth={false}

sx={{

px: { xs: "30px", sm: "60px" },

pb: 4,

pt: "150px",

bgcolor: "inherit",

}}

>

variant="h5"

sx={{ color: "text.primary", mb: 2 }}

>{`${genre.name} Movies`}

{data.results

.filter((v) => !!v.backdrop\_path)

.map((video, idx) => (

key={`${video.id}\_${idx}`}

item

xs={6}

sm={3}

md={2}

sx={{ zIndex: 1 }}

>

))}

);

}

src/components/HeroSection.tsx

"use client";

import { useEffect, useState, useMemo, useCallback, useRef } from "react";

import Box from "@mui/material/Box";

import Stack from "@mui/material/Stack";

import VolumeUpIcon from "@mui/icons-material/VolumeUp";

import VolumeOffIcon from "@mui/icons-material/VolumeOff";

import Player from "video.js/dist/types/player";

import { getRandomNumber } from "src/utils/common";

import MaxLineTypography from "./MaxLineTypography";

import PlayButton from "./PlayButton";

import MoreInfoButton from "./MoreInfoButton";

import NetflixIconButton from "./NetflixIconButton";

import MaturityRate from "./MaturityRate";

import useOffSetTop from "src/hooks/useOffSetTop";

import { useDetailModal } from "src/providers/DetailModalProvider";

import { MEDIA\_TYPE } from "src/types/Common";

import {

useGetVideosByMediaTypeAndCustomGenreQuery,

useLazyGetAppendedVideosQuery,

} from "src/store/slices/discover";

import { Movie } from "src/types/Movie";

import VideoJSPlayer from "./watch/VideoJSPlayer";

interface TopTrailerProps {

mediaType: MEDIA\_TYPE;

}

export default function TopTrailer({ mediaType }: TopTrailerProps) {

const { data } = useGetVideosByMediaTypeAndCustomGenreQuery({

mediaType,

apiString: "popular",

page: 1,

});

const [getVideoDetail, { data: detail }] = useLazyGetAppendedVideosQuery();

const [video, setVideo] = useState(null);

const [muted, setMuted] = useState(true);

const playerRef = useRef(null);

const [threshold, setThreshold] = useState(0);

const isOffset = useOffSetTop(threshold);

const { setDetailType } = useDetailModal();

const maturityRate = useMemo(() => {

return getRandomNumber(20);

}, []);

const handleReady = useCallback((player: Player) => {

playerRef.current = player;

}, []);

useEffect(() => {

setThreshold(window.innerWidth \* 0.5625);

if (playerRef.current) {

if (isOffset) {

playerRef.current.pause();

} else {

if (playerRef.current.paused()) {

playerRef.current.play();

}

}

}

}, [isOffset]);

useEffect(() => {

if (data && data.results) {

const videos = data.results.filter((item) => !!item.backdrop\_path);

setVideo(videos[getRandomNumber(videos.length)]);

}

// eslint-disable-next-line react-hooks/exhaustive-deps

}, [data]);

useEffect(() => {

if (video) {

getVideoDetail({ mediaType, id: video.id });

}

// eslint-disable-next-line react-hooks/exhaustive-deps

}, [video]);

const handleMute = useCallback((status: boolean) => {

if (playerRef.current) {

playerRef.current.muted(!status);

setMuted(!status);

}

}, []);

return (

sx={{

mb: 3,

pb: "40%",

top: 0,

left: 0,

right: 0,

position: "relative",

}}

>

sx={{

width: "100%",

height: "56.25vw",

position: "absolute",

}}

>

{video && (

<>

sx={{

top: 0,

left: 0,

right: 0,

bottom: 0,

position: "absolute",

}}

>

{detail && (

options={{

loop: true,

muted: true,

autoplay: true,

controls: false,

responsive: true,

fluid: true,

techOrder: ["youtube"],

sources: [

{

type: "video/youtube",

src: `https://www.youtube.com/watch?v=${

detail.videos.results[0]?.key || "L3oOldViIgY"

}`,

},

],

}}

onReady={handleReady}

/>

)}

sx={{

background: `linear-gradient(77deg,rgba(0,0,0,.6),transparent 85%)`,

top: 0,

left: 0,

bottom: 0,

right: "26.09%",

opacity: 1,

position: "absolute",

transition: "opacity .5s",

}}

/>

sx={{

backgroundColor: "transparent",

backgroundImage:

"linear-gradient(180deg,hsla(0,0%,8%,0) 0,hsla(0,0%,8%,.15) 15%,hsla(0,0%,8%,.35) 29%,hsla(0,0%,8%,.58) 44%,#141414 68%,#141414)",

backgroundRepeat: "repeat-x",

backgroundPosition: "0px top",

backgroundSize: "100% 100%",

bottom: 0,

position: "absolute",

height: "14.7vw",

opacity: 1,

top: "auto",

width: "100%",

}}

/>

direction="row"

spacing={2}

sx={{

alignItems: "center",

position: "absolute",

right: 0,

bottom: "35%",

}}

>

size="large"

onClick={() => handleMute(muted)}

sx={{ zIndex: 2 }}

>

{!muted ? : }

{`${maturityRate}+`}

sx={{

position: "absolute",

top: 0,

left: 0,

right: 0,

bottom: 0,

width: "100%",

height: "100%",

}}

>

spacing={4}

sx={{

bottom: "35%",

position: "absolute",

left: { xs: "4%", md: "60px" },

top: 0,

width: "36%",

zIndex: 10,

justifyContent: "flex-end",

}}

>

variant="h2"

maxLine={1}

color="text.primary"

>

{video.title}

variant="h5"

maxLine={3}

color="text.primary"

>

{video.overview}

size="large"

onClick={() => {

setDetailType({ mediaType, id: video.id });

}}

/>

)}

);

}

src/components/Logo.tsx

// src/components/Logo.tsx

"use client";

import Link from "next/link";

import Box, { BoxProps } from "@mui/material/Box";

import { MAIN\_PATH } from "src/constant";

export default function Logo({ sx, ...rest }: BoxProps) {

return (

component="img"

alt="Netflix Logo"

src="/assets/netflix-logo.png"

width={87}

height={25}

sx={sx}

{...rest}

/>

);

}

src/components/MainLoadingScreen.tsx

"use client";

import CircularProgress from "@mui/material/CircularProgress";

function MainLoadingScreen() {

return (

style={{

top: 0,

left: 0,

right: 0,

bottom: 0,

display: "flex",

justifyContent: "center",

alignItems: "center",

position: "fixed",

backgroundColor: "#141414",

opacity: 0.75,

zIndex: 2,

}}

>

);

}

export default MainLoadingScreen;

src/components/MaturityRate.tsx

"use client";

import Box from "@mui/material/Box";

import { ReactNode } from "react";

export default function MaturityRate({ children }: { children: ReactNode }) {

return (

sx={{

py: 1,

pl: 1.5,

pr: 3,

fontSize: 22,

display: "flex",

alignItem: "center",

color: "text.primary",

border: "3px #dcdcdc",

borderLeftStyle: "solid",

bgcolor: "#33333399",

}}

>

{children}

);

}

src/components/MaturityRate.tsx

"use client";

import Box from "@mui/material/Box";

import { ReactNode } from "react";

export default function MaturityRate({ children }: { children: ReactNode }) {

return (

sx={{

py: 1,

pl: 1.5,

pr: 3,

fontSize: 22,

display: "flex",

alignItem: "center",

color: "text.primary",

border: "3px #dcdcdc",

borderLeftStyle: "solid",

bgcolor: "#33333399",

}}

>

{children}

);

}

src/components/MaxLineTypography.tsx

"use client";

import { forwardRef } from "react";

import Typography, { TypographyProps } from "@mui/material/Typography";

const MaxLineTypography = forwardRef<

HTMLDivElement,

TypographyProps & { maxLine: number }

>(({ maxLine, children, sx, ...others }, ref) => {

return (

ref={ref}

sx={{

overflow: "hidden",

textOverflow: "ellipsis",

display: "-webkit-box",

WebkitLineClamp: maxLine,

WebkitBoxOrient: "vertical",

...sx,

}}

{...others}

>

{children}

);

});

export default MaxLineTypography;

src/components/MoreInfoButton.tsx

"use client";

import Button, { ButtonProps } from "@mui/material/Button";

import InfoOutlinedIcon from "@mui/icons-material/InfoOutlined";

export default function MoreInfoButton({ sx, ...others }: ButtonProps) {

return (

variant="contained"

startIcon={

sx={{

fontSize: {

xs: "24px !important",

sm: "32px !important",

md: "40px !important",

},

}}

/>

}

{...others}

sx={{

...sx,

px: { xs: 1, sm: 2 },

py: { xs: 0.5, sm: 1 },

fontSize: { xs: 18, sm: 24, md: 28 },

lineHeight: 1.5,

fontWeight: "bold",

textTransform: "capitalize",

bgcolor: "#6d6d6eb3",

whiteSpace: "nowrap",

"&:hover": { bgcolor: "#6d6d6e66" },

}}

>

More Info

);

}

src/components/NetflixIconButton.tsx

"use client";

import { forwardRef } from "react";

import IconButton, { IconButtonProps } from "@mui/material/IconButton";

const NetflixIconButton = forwardRef(

({ children, sx, ...others }, ref) => {

return (

sx={{

color: "white",

borderWidth: "2px",

borderStyle: "solid",

borderColor: "grey.700",

"&:hover, &:focus": {

borderColor: "grey.200",

},

...sx,

}}

{...others}

ref={ref}

>

{children}

);

}

);

export default NetflixIconButton;

src/components/NetflixNavigationLink.tsx

"use client";

import NextLink from "next/link";

import Link, { LinkProps } from "@mui/material/Link";

export default function NetflixNavigationLink({

sx,

children,

...others

}: LinkProps & { href: string }) {

return (

{...others}

component={NextLink}

sx={{ color: "text.primary", textDecoration: "none", ...sx }}

>

{children}

);

}

src/components/PlayButton.tsx

"use client";

import Button, { ButtonProps } from "@mui/material/Button";

import PlayArrowIcon from "@mui/icons-material/PlayArrow";

import { useRouter } from "next/navigation";

import { MAIN\_PATH } from "src/constant";

export default function PlayButton({ sx, ...others }: ButtonProps) {

const router = useRouter();

return (

color="inherit"

variant="contained"

startIcon={

sx={{

fontSize: {

xs: "24px !important",

sm: "32px !important",

md: "40px !important",

},

}}

/>

}

{...others}

sx={{

px: { xs: 1, sm: 2 },

py: { xs: 0.5, sm: 1 },

fontSize: { xs: 18, sm: 24, md: 28 },

lineHeight: 1.5,

fontWeight: "bold",

whiteSpace: "nowrap",

textTransform: "capitalize",

...sx,

}}

onClick={() => router.push(`/${MAIN\_PATH.watch}`)}

>

Play

);

}

src/components/QualityChip.tsx

"use client";

import Chip from "@mui/material/Chip";

import type { ChipProps } from "@mui/material/Chip";

export default function QualityChip({ sx, ...others }: ChipProps) {

return (

variant="outlined"

{...others}

sx={{

borderRadius: "4px",

p: 0.5,

fontSize: 12,

height: "100%",

"& .MuiChip-label": { p: 0 },

...sx,

}}

/>

);

}

src/components/SearchBox.tsx

"use client";

import { useState, useRef } from "react";

import { styled } from "@mui/material/styles";

import InputBase from "@mui/material/InputBase";

import SearchIcon from "@mui/icons-material/Search";

const Search = styled("div")(({ theme }) => ({

position: "relative",

width: "100%",

display: "flex",

alignItems: "center",

}));

const SearchIconWrapper = styled("div")(({ theme }) => ({

cursor: "pointer",

padding: theme.spacing(0, 1),

height: "100%",

display: "flex",

alignItems: "center",

justifyContent: "center",

}));

const StyledInputBase = styled(InputBase)(({ theme }) => ({

color: "inherit",

// MUI'nin input class'ı:

"& .MuiInputBase-input": {

width: 0,

padding: theme.spacing(0.75, 1),

transition: theme.transitions.create("width", {

duration: theme.transitions.duration.complex,

easing: theme.transitions.easing.easeIn,

}),

// Odaklanınca genişlet

"&:focus": {

width: 220, // px cinsinden sabit bir genişlik daha sorunsuz çalışır

},

},

}));

export default function SearchBox() {

const [isFocused, setIsFocused] = useState(false);

const searchInputRef = useRef(null);

const handleClickSearchIcon = () => {

if (!isFocused) {

searchInputRef.current?.focus();

}

};

return (

sx={

isFocused

? { border: "1px solid white", backgroundColor: "black" }

: undefined

}

>

inputRef={searchInputRef}

placeholder="Titles, people, genres"

inputProps={{

"aria-label": "search",

onFocus: () => setIsFocused(true),

onBlur: () => setIsFocused(false),

}}

/>

);

}

src/components/SimilarVideoCard.tsx

"use client";

import { useMemo } from "react";

import Stack from "@mui/material/Stack";

import Card from "@mui/material/Card";

import CardContent from "@mui/material/CardContent";

import Typography from "@mui/material/Typography";

import AddIcon from "@mui/icons-material/Add";

import { Movie } from "src/types/Movie";

import NetflixIconButton from "./NetflixIconButton";

import MaxLineTypography from "./MaxLineTypography";

import AgeLimitChip from "./AgeLimitChip";

import { useGetConfigurationQuery } from "src/store/slices/configuration";

// SSR ↔ CSR uyumsuzluğunu önlemek için videoya bağlı deterministik “random”

function seededInt(id: number, salt: number, min: number, max: number) {

const x = Math.imul((id ^ salt) >>> 0, 2654435761) >>> 0;

return min + (x % (max - min + 1));

}

function formatMinuteToReadable(mins: number) {

const h = Math.floor(mins / 60);

const m = mins % 60;

return h > 0 ? `${h}h ${m}m` : `${m}m`;

}

interface SimilarVideoCardProps {

video: Movie;

}

export default function SimilarVideoCard({ video }: SimilarVideoCardProps) {

const { data: configuration } = useGetConfigurationQuery(undefined);

// Değerleri video.id’ye göre sabitle (SSR ve client aynı olur)

const ui = useMemo(

() => ({

match: seededInt(video.id, 1, 50, 99), // 50–99 arası “% Match”

age: seededInt(video.id, 2, 7, 18), // 7–18 arası yaş etiketi

mins: seededInt(video.id, 3, 60, 180), // 60–180 dk

}),

[video.id]

);

const year =

(video as any)?.release\_date?.substring?.(0, 4) ??

(video as any)?.first\_air\_date?.substring?.(0, 4) ??

"";

return (

style={{

width: "100%",

position: "relative",

paddingTop: "calc(9 / 16 \* 100%)",

}}

>

src={`${configuration?.images.base\_url}w780${video.backdrop\_path}`}

alt={video.title || "Backdrop"}

loading="lazy"

style={{

left: 0,

top: 0,

width: "100%",

height: "100%",

position: "absolute",

objectFit: "cover",

}}

/>

style={{

top: 10,

right: 15,

position: "absolute",

}}

>

{formatMinuteToReadable(ui.mins)}

style={{

left: 0,

right: 0,

bottom: 0,

paddingLeft: "16px",

paddingRight: "16px",

paddingBottom: "4px",

position: "absolute",

}}

>

maxLine={1}

sx={{ width: "80%", fontWeight: 700 }}

variant="subtitle1"

>

{video.title}

{ui.match}% Match

{!!year && {year}}

{video.overview}

);

}

src/components/VideoCardPortal.tsx

"use client";

import { useRouter } from "next/navigation";

import Stack from "@mui/material/Stack";

import Card from "@mui/material/Card";

import CardContent from "@mui/material/CardContent";

import Typography from "@mui/material/Typography";

import VolumeUpIcon from "@mui/icons-material/VolumeUp";

import PlayCircleIcon from "@mui/icons-material/PlayCircle";

import ThumbUpOffAltIcon from "@mui/icons-material/ThumbUpOffAlt";

import AddIcon from "@mui/icons-material/Add";

import ExpandMoreIcon from "@mui/icons-material/ExpandMore";

import { Movie } from "src/types/Movie";

import { usePortal } from "src/providers/PortalProvider";

import { useDetailModal } from "src/providers/DetailModalProvider";

import { formatMinuteToReadable, getRandomNumber } from "src/utils/common";

import NetflixIconButton from "./NetflixIconButton";

import MaxLineTypography from "./MaxLineTypography";

import AgeLimitChip from "./AgeLimitChip";

import QualityChip from "./QualityChip";

import GenreBreadcrumbs from "./GenreBreadcrumbs";

import { useGetConfigurationQuery } from "src/store/slices/configuration";

import { MEDIA\_TYPE } from "src/types/Common";

import { useGetGenresQuery } from "src/store/slices/genre";

import { MAIN\_PATH } from "src/constant";

interface VideoCardModalProps {

video: Movie;

anchorElement: HTMLElement;

}

export default function VideoCardModal({

video,

anchorElement,

}: VideoCardModalProps) {

const router = useRouter();

const { data: configuration } = useGetConfigurationQuery(undefined);

const { data: genres } = useGetGenresQuery(MEDIA\_TYPE.Movie);

const setPortal = usePortal();

const rect = anchorElement.getBoundingClientRect();

const { setDetailType } = useDetailModal();

return (

onPointerLeave={() => {

setPortal(null, null);

}}

sx={{

width: rect.width \* 1.5,

height: "100%",

}}

>

style={{

width: "100%",

position: "relative",

paddingTop: "calc(9 / 16 \* 100%)",

}}

>

src={`${configuration?.images.base\_url}w780${video.backdrop\_path}`}

alt={video.title || "Video backdrop"}

style={{

top: 0,

height: "100%",

objectFit: "cover",

position: "absolute",

backgroundPosition: "50%",

}}

/>

style={{

display: "flex",

flexDirection: "row",

alignItems: "center",

left: 0,

right: 0,

bottom: 0,

paddingLeft: "16px",

paddingRight: "16px",

paddingBottom: "4px",

position: "absolute",

}}

>

maxLine={2}

sx={{ width: "80%", fontWeight: 700 }}

variant="h6"

>

{video.title}

sx={{ p: 0 }}

onClick={() => router.push(`/${MAIN\_PATH.watch}`)}

>

onClick={() => {

setDetailType({ mediaType: MEDIA\_TYPE.Movie, id: video.id });

}}

>

variant="subtitle1"

sx={{ color: "success.main" }}

>{`${getRandomNumber(100)}% Match`}

{`${formatMinuteToReadable(

getRandomNumber(180)

)}`}

{genres && (

genres={genres

.filter((genre) => video.genre\_ids.includes(genre.id))

.map((genre) => genre.name)}

/>

)}

);

}

src/components/VideoItemWithHover.tsx

"use client";

import { useEffect, useState, useRef } from "react";

import { Movie } from "src/types/Movie";

import { usePortal } from "src/providers/PortalProvider";

import { useGetConfigurationQuery } from "src/store/slices/configuration";

import VideoItemWithHoverPure from "./VideoItemWithHoverPure";

interface VideoItemWithHoverProps {

video: Movie;

}

const CLOSE\_DELAY = 180; // kartı kapatmadan önce bekleme

const ENTER\_LOCK\_MS = 220; // karta girer girmez kısa süreli "kapatma kilidi"

export default function VideoItemWithHover({ video }: VideoItemWithHoverProps) {

const setPortal = usePortal();

const elementRef = useRef(null);

const [isHovered, setIsHovered] = useState(false);

const closeTimerRef = useRef(null);

const leaveTimerRef = useRef(null);

const fromPortalRef = useRef(false); // imleç mini-portal içindeyse true

const enterLockRef = useRef(0); // ilk giriş sonrası kısa kilit

const { data: configuration } = useGetConfigurationQuery(undefined);

// Mini portal "enter/leave" olaylarını dinle (VideoCardPortal tetikliyor)

useEffect(() => {

const onPortalEnter = () => {

// Yakın kapatma zamanlayıcısı varsa iptal et

if (closeTimerRef.current) {

window.clearTimeout(closeTimerRef.current);

closeTimerRef.current = null;

}

fromPortalRef.current = true;

enterLockRef.current = Date.now() + ENTER\_LOCK\_MS; // biraz süre kapatma yok

};

const onPortalLeave = () => {

fromPortalRef.current = false;

};

// TS: custom event adları olduğu için EventListener cast ediyoruz

window.addEventListener("miniportal:enter", onPortalEnter as unknown as EventListener);

window.addEventListener("miniportal:leave", onPortalLeave as unknown as EventListener);

return () => {

window.removeEventListener("miniportal:enter", onPortalEnter as unknown as EventListener);

window.removeEventListener("miniportal:leave", onPortalLeave as unknown as EventListener);

};

}, []);

// Hover durumuna göre portalı aç/kapat

useEffect(() => {

if (isHovered && elementRef.current) {

// varsa kapatma zamanlayıcısını iptal et

if (closeTimerRef.current) {

window.clearTimeout(closeTimerRef.current);

closeTimerRef.current = null;

}

// aynı hedefle portali aç/güncelle

setPortal(elementRef.current, video);

} else {

// anında kapatma yerine küçük gecikme

closeTimerRef.current = window.setTimeout(() => {

setPortal(null, null);

closeTimerRef.current = null;

}, CLOSE\_DELAY);

}

// güvenlik: unmount’ta temizle

return () => {

if (closeTimerRef.current) {

window.clearTimeout(closeTimerRef.current);

closeTimerRef.current = null;

}

};

}, [isHovered, setPortal, video]);

// Pure bileşene verilecek hover kontrolü

const handleHoverChange = (v: boolean) => {

if (v) {

// hover başladı

if (leaveTimerRef.current) {

window.clearTimeout(leaveTimerRef.current);

leaveTimerRef.current = null;

}

fromPortalRef.current = false; // yeniden girişte sıfırla

enterLockRef.current = Date.now() + ENTER\_LOCK\_MS;

setIsHovered(true);

} else {

// hover biter gibi—portal’a geçiş olabilir, kısa geciktir

if (!leaveTimerRef.current) {

leaveTimerRef.current = window.setTimeout(() => {

// kilit sürerken veya portal içindeyken kapatma

if (!fromPortalRef.current && Date.now() > enterLockRef.current) {

setIsHovered(false);

}

leaveTimerRef.current = null;

}, 160);

}

}

};

return (

ref={elementRef}

handleHover={handleHoverChange}

src={`${configuration?.images.base\_url ?? ""}w300${video.backdrop\_path}`}

/>

);

}

src/components/VideoItemWithHoverPure.tsx

"use client";

import { PureComponent, ForwardedRef, forwardRef } from "react";

type VideoItemWithHoverPureType = {

src: string;

innerRef: ForwardedRef;

handleHover: (value: boolean) => void;

};

class VideoItemWithHoverPure extends PureComponent {

render() {

return (

ref={this.props.innerRef}

style={{

zIndex: 9,

cursor: "pointer",

borderRadius: 0.5,

width: "100%",

position: "relative",

paddingTop: "calc(9 / 16 \* 100%)",

}}

>

src={this.props.src}

alt=""

loading="lazy"

style={{

top: 0,

height: "100%",

objectFit: "cover",

position: "absolute",

borderRadius: "4px",

}}

onPointerEnter={() => {

this.props.handleHover(true);

}}

onPointerLeave={() => {

this.props.handleHover(false);

}}

/>

);

}

}

const VideoItemWithHoverRef = forwardRef<

HTMLDivElement,

Omit

>((props, ref) => );

VideoItemWithHoverRef.displayName = "VideoItemWithHoverRef";

export default VideoItemWithHoverRef;

src/components/VideoPortalContainer.tsx

"use client";

import { useRef, useEffect, useState, useCallback } from "react";

import { motion } from "framer-motion";

import Portal from "@mui/material/Portal";

import VideoCardPortal from "./VideoCardPortal";

import MotionContainer from "./animate/MotionContainer";

import {

varZoomIn,

varZoomInLeft,

varZoomInRight,

} from "./animate/variants/zoom/ZoomIn";

import { usePortalData } from "src/providers/PortalProvider";

export default function VideoPortalContainer() {

const { miniModalMediaData, anchorElement } = usePortalData();

const container = useRef(null);

const rafRef = useRef(null);

const [coords, setCoords] = useState<{ top: number; left: number; width: number }>({

top: 0,

left: 0,

width: 0,

});

const hasToRender = !!miniModalMediaData && !!anchorElement;

const computeAndSet = useCallback(() => {

if (!anchorElement) return;

// aktif slide kenarlarını tespit et

let isFirstElement = false;

let isLastElement = false;

const parentEl = anchorElement.closest(".slick-active") as HTMLElement | null;

const nextSib = parentEl?.nextElementSibling as HTMLElement | null;

const prevSib = parentEl?.previousElementSibling as HTMLElement | null;

if (!prevSib?.classList.contains("slick-active")) {

isFirstElement = true;

} else if (!nextSib?.classList.contains("slick-active")) {

isLastElement = true;

}

const rect = anchorElement.getBoundingClientRect();

const vw = document.documentElement.clientWidth;

const cardWidth = rect.width \* 1.5;

// yatay konum (clamp'li)

let left = isFirstElement ? rect.left : rect.left - 0.25 \* rect.width;

if (isLastElement) left = rect.right - cardWidth;

left = Math.max(8, Math.min(left, vw - cardWidth - 8));

// dikey konum (fixed kullandığımız için pageYOffset eklemiyoruz)

const top = Math.max(8, rect.top - 0.6 \* rect.height);

setCoords({ top, left, width: cardWidth });

}, [anchorElement]);

// ilk render + scroll/resize’larda yeniden konumlandır

useEffect(() => {

if (!hasToRender) return;

computeAndSet();

const onMove = () => {

if (rafRef.current) cancelAnimationFrame(rafRef.current);

rafRef.current = requestAnimationFrame(computeAndSet);

};

window.addEventListener("scroll", onMove, { passive: true });

window.addEventListener("resize", onMove);

return () => {

window.removeEventListener("scroll", onMove);

window.removeEventListener("resize", onMove);

if (rafRef.current) cancelAnimationFrame(rafRef.current);

rafRef.current = null;

};

}, [hasToRender, computeAndSet]);

// animasyon yönü

let variant = varZoomIn;

if (hasToRender && anchorElement) {

const parentEl = anchorElement.closest(".slick-active") as HTMLElement | null;

const nextSib = parentEl?.nextElementSibling as HTMLElement | null;

const prevSib = parentEl?.previousElementSibling as HTMLElement | null;

if (!prevSib?.classList.contains("slick-active")) {

variant = varZoomInLeft;

} else if (!nextSib?.classList.contains("slick-active")) {

variant = varZoomInRight;

}

}

return (

<>

ref={container}

variants={variant}

style={{

zIndex: 1300,

position: "fixed",

display: hasToRender ? "inline-block" : "none",

...(hasToRender && {

top: coords.top,

left: coords.left,

width: coords.width,

}),

}}

/>

{hasToRender && (

video={miniModalMediaData}

anchorElement={anchorElement}

/>

)}

);

}

src/components/VideoSlider.tsx

"use client";

import { useMemo } from "react";

import withPagination from "src/hoc/withPagination";

import { MEDIA\_TYPE } from "src/types/Common";

import { CustomGenre, Genre } from "src/types/Genre";

import SlickSlider from "./slick-slider/SlickSlider";

interface SliderRowForGenreProps {

genre: Genre | CustomGenre;

mediaType: MEDIA\_TYPE;

}

export default function SliderRowForGenre({ genre, mediaType }: SliderRowForGenreProps) {

// HOC ile oluşturulan bileşenin referansını her render'da yeniden yaratmamak için sabitle

const Component = useMemo(

() => withPagination(SlickSlider, mediaType, genre),

[mediaType, genre]

);

return ;

}

src/constant/index.ts

import { CustomGenre } from "src/types/Genre";

export const API\_ENDPOINT\_URL = process.env.NEXT\_PUBLIC\_API\_ENDPOINT\_URL as string;

export const TMDB\_V3\_API\_KEY = process.env.NEXT\_PUBLIC\_TMDB\_V3\_API\_KEY as string;

export const MAIN\_PATH = {

root: "",

browse: "browse",

genreExplore: "genre",

watch: "watch",

};

export const ARROW\_MAX\_WIDTH = 60;

export const COMMON\_TITLES: CustomGenre[] = [

{ name: "Popular", apiString: "popular" },

{ name: "Top Rated", apiString: "top\_rated" },

{ name: "Now Playing", apiString: "now\_playing" },

{ name: "Upcoming", apiString: "upcoming" },

];

export const YOUTUBE\_URL = "https://www.youtube.com/watch?v=";

export const APP\_BAR\_HEIGHT = 70;

export const INITIAL\_DETAIL\_STATE = {

id: undefined,

mediaType: undefined,

mediaDetail: undefined,

};

src/hoc/withPagination.tsx

import { ElementType, useCallback, useEffect } from "react";

import MainLoadingScreen from "src/components/MainLoadingScreen";

import { useAppDispatch, useAppSelector } from "src/hooks/redux";

import {

initiateItem,

useLazyGetVideosByMediaTypeAndGenreIdQuery,

useLazyGetVideosByMediaTypeAndCustomGenreQuery,

} from "src/store/slices/discover";

import { MEDIA\_TYPE } from "src/types/Common";

import { CustomGenre, Genre } from "src/types/Genre";

export default function withPagination(

Component: ElementType,

mediaType: MEDIA\_TYPE,

genre: Genre | CustomGenre

) {

return function WithPagination() {

const dispatch = useAppDispatch();

const itemKey = genre.id ?? (genre as CustomGenre).apiString;

const mediaState = useAppSelector((state) => state.discover[mediaType]);

const pageState = mediaState ? mediaState[itemKey] : undefined;

const [getVideosByMediaTypeAndGenreId] =

useLazyGetVideosByMediaTypeAndGenreIdQuery();

const [getVideosByMediaTypeAndCustomGenre] =

useLazyGetVideosByMediaTypeAndCustomGenreQuery();

useEffect(() => {

if (!mediaState || !pageState) {

dispatch(initiateItem({ mediaType, itemKey }));

}

}, [mediaState, pageState]);

useEffect(() => {

if (pageState && pageState.page === 0) {

handleNext(pageState.page + 1);

}

}, [pageState]);

const handleNext = useCallback((page: number) => {

if (genre.id) {

getVideosByMediaTypeAndGenreId({

mediaType,

genreId: genre.id,

page,

});

} else {

getVideosByMediaTypeAndCustomGenre({

mediaType,

apiString: (genre as CustomGenre).apiString,

page,

});

}

// dispatch(setNextPage({ mediaType, itemKey }));

}, []);

if (pageState) {

return (

);

}

return ;

};

}

src/hooks/redux.ts

import { TypedUseSelectorHook, useDispatch, useSelector } from "react-redux";

import type { RootState, AppDispatch } from "../store";

// Use throughout your app instead of plain `useDispatch` and `useSelector`

export const useAppDispatch: () => AppDispatch = useDispatch;

export const useAppSelector: TypedUseSelectorHook = useSelector;

src/hooks/useIntersectionObserver.ts

import { useEffect, useState } from "react";

// import { buildThresholdList } from "src/utils";

export default function useIntersectionObserver(

ref: React.RefObject,

options?: IntersectionObserverInit

) {

const [entry, setEntry] = useState(null);

const node = ref.current;

if (!node) return;

// SSR ve destek kontrolü

if (typeof window === "undefined" || !("IntersectionObserver" in window)) {

return;

}

useEffect(() => {

const observer = new IntersectionObserver(

([entry]) => {

setEntry(entry);

},

options ?? {

root: null,

rootMargin: "0px",

// threshold: buildThresholdList(),

threshold: 1,

}

);

if (ref.current) {

observer.observe(node);

}

return () => observer.disconnect();

}, [ref, options?.root, options?.rootMargin, options?.threshold]);

return entry;

}

src/hooks/useOffSetTop.ts

import { useState, useEffect, useCallback } from "react";

export default function useOffSetTop(top: number) {

const [offsetTop, setOffSetTop] = useState(false);

const onScroll = useCallback(() => {

if (window.pageYOffset > top) {

setOffSetTop(true);

} else {

setOffSetTop(false);

}

}, [top]);

useEffect(() => {

window.addEventListener("scroll", onScroll);

return () => {

window.removeEventListener("scroll", onScroll);

};

}, [top]);

return offsetTop;

}

src/hooks/useWindowSize.ts

import { useState, useEffect } from "react";

export default function useWindowSize() {

// Initialize state with undefined width/height so server and client renders match

// Learn more here: https://joshwcomeau.com/react/the-perils-of-rehydration/

const [windowSize, setWindowSize] = useState<{

width: number | undefined;

height: number | undefined;

}>({

width: undefined,

height: undefined,

});

useEffect(() => {

// Handler to call on window resize

function handleResize() {

// Set window width/height to state

setWindowSize({

width: window.innerWidth,

height: window.innerHeight,

});

}

// Add event listener

window.addEventListener("resize", handleResize);

// Call handler right away so state gets updated with initial window size

handleResize();

// Remove event listener on cleanup

return () => window.removeEventListener("resize", handleResize);

}, []); // Empty array ensures that effect is only run on mount

return windowSize;

}

src/layouts/MainLayout.tsx

"use client";

import { usePathname } from "next/navigation";

import Box from "@mui/material/Box";

import DetailModal from "src/components/DetailModal";

import VideoPortalContainer from "src/components/VideoPortalContainer";

import DetailModalProvider from "src/providers/DetailModalProvider";

import PortalProvider from "src/providers/PortalProvider";

import { MAIN\_PATH } from "src/constant";

import { Footer, MainHeader } from "src/components/layouts";

// import MainLoadingScreen from "src/components/MainLoadingScreen";

export default function MainLayout({

children,

}: {

children: React.ReactNode;

}) {

const pathname = usePathname();

return (

sx={{

width: "100%",

minHeight: "100vh",

bgcolor: "background.default",

}}

>

{/\* Next.js App Router'da loading için route'a özel `loading.tsx` veya Suspense kullanın \*/}

{/\* \*/}

{children}

{pathname !== `/${MAIN\_PATH.watch}` && }

);

}

src/lib/createSafeContext.ts

import React from "react";

export default function createSafeContext() {

const context = React.createContext(undefined);

function useContext() {

const value = React.useContext(context);

if (value === undefined) {

throw new Error("useContext must be inside a Provider with a value");

}

return value;

}

return [useContext, context.Provider] as const;

}

src/pages/GenreExplore.tsx

"use client";

import { COMMON\_TITLES } from "src/constant";

import GridPage from "src/components/GridPage";

import { MEDIA\_TYPE } from "src/types/Common";

import { CustomGenre, Genre } from "src/types/Genre";

import store from "src/store";

import { useGetGenresQuery } from "src/store/slices/genre";

import { useParams } from "next/navigation";

export default function Page() {

const params = useParams<{ genreId: string }>();

const genreId = params?.genreId;

const { data: genres } = useGetGenresQuery(MEDIA\_TYPE.Movie);

let genre: CustomGenre | Genre | undefined;

if (genreId) {

if (Number.isNaN(Number(genreId))) {

// string türündeki özel başlıklar (Popular / Top Rated vs.)

genre = COMMON\_TITLES.find((t: CustomGenre) => t.apiString === genreId);

} else {

// sayısal TMDB genre id

genre = genres?.find((t: Genre) => String(t.id) === String(genreId));

}

}

return genre ? : null;

}

src/pages/HomePage.tsx

import Stack from "@mui/material/Stack";

import { COMMON\_TITLES } from "src/constant";

import HeroSection from "src/components/HeroSection";

import { useGetGenresQuery } from "src/store/slices/genre";

import { MEDIA\_TYPE } from "src/types/Common";

import { CustomGenre, Genre } from "src/types/Genre";

import SliderRowForGenre from "src/components/VideoSlider";

import store from "src/store";

export default function Page() {

const { data: genres, isSuccess } = useGetGenresQuery(MEDIA\_TYPE.Movie);

if (isSuccess && genres && genres.length > 0) {

return (

{[...COMMON\_TITLES, ...genres].map((genre: Genre | CustomGenre) => (

key={genre.id || genre.name}

genre={genre}

mediaType={MEDIA\_TYPE.Movie}

/>

))}

);

}

return null;

}

src/pages/WatchPage.tsx

import { useState, useRef, useMemo } from "react";

import { useRouter } from "next/navigation";

import Player from "video.js/dist/types/player";

import { Box, Stack, Typography } from "@mui/material";

import PlayArrowIcon from "@mui/icons-material/PlayArrow";

import PauseIcon from "@mui/icons-material/Pause";

import SkipNextIcon from "@mui/icons-material/SkipNext";

import FullscreenIcon from "@mui/icons-material/Fullscreen";

import SettingsIcon from "@mui/icons-material/Settings";

import BrandingWatermarkOutlinedIcon from "@mui/icons-material/BrandingWatermarkOutlined";

import KeyboardBackspaceIcon from "@mui/icons-material/KeyboardBackspace";

import useWindowSize from "src/hooks/useWindowSize";

import { formatTime } from "src/utils/common";

import MainLoadingScreen from "src/components/MainLoadingScreen";

import MaxLineTypography from "src/components/MaxLineTypography";

import VolumeControllers from "src/components/watch/VolumeControllers";

import VideoJSPlayer from "src/components/watch/VideoJSPlayer";

import PlayerSeekbar from "src/components/watch/PlayerSeekbar";

import PlayerControlButton from "src/components/watch/PlayerControlButton";

export function Component() {

const playerRef = useRef(null);

const [playerState, setPlayerState] = useState({

paused: false,

muted: false,

playedSeconds: 0,

duration: 0,

volume: 0.8,

loaded: 0,

});

const router = useRouter();

const [playerInitialized, setPlayerInitialized] = useState(false);

const windowSize = useWindowSize();

const videoJsOptions = useMemo(() => {

return {

preload: "metadata",

autoplay: true,

controls: false,

// responsive: true,

// fluid: true,

width: windowSize.width,

height: windowSize.height,

sources: [

{

// src: videoData?.video,

// src: "https://d2zihajmogu5jn.cloudfront.net/bipbop-advanced/bipbop\_16x9\_variant.m3u8",

src: "https://bitmovin-a.akamaihd.net/content/sintel/hls/playlist.m3u8",

type: "application/x-mpegurl",

},

],

};

// eslint-disable-next-line react-hooks/exhaustive-deps

}, [windowSize]);

const handlePlayerReady = function (player: Player): void {

player.on("pause", () => {

setPlayerState((draft) => {

return { ...draft, paused: true };

});

});

player.on("play", () => {

setPlayerState((draft) => {

return { ...draft, paused: false };

});

});

player.on("timeupdate", () => {

setPlayerState((prev) => ({ ...prev, playedSeconds: Number(player.currentTime() ?? 0) }));

});

player.one("durationchange", () => {

setPlayerInitialized(true);

setPlayerState((prev) => ({ ...prev, duration: Number(player.duration() ?? 0) }));

});

playerRef.current = player;

setPlayerState((prev) => ({ ...prev, paused: player.paused() }));

};

const handleVolumeChange = (\_: unknown, value: number | number[]) => {

const v = Array.isArray(value) ? value[0] : value;

playerRef.current?.volume(v / 100);

setPlayerState((prev) => ({ ...prev, volume: v / 100 }));

};

const handleSeekTo = (v: number) => {

playerRef.current?.currentTime(v);

};

const handleGoBack = () => {

router.push("/browse");

};

if (!!videoJsOptions.width) {

return (

sx={{

position: "relative",

}}

>

{playerRef.current && playerInitialized && (

sx={{

top: 0,

left: 0,

right: 0,

bottom: 0,

position: "absolute",

}}

>

px={2}

sx={{

position: "absolute",

top: { xs: "40%", sm: "55%", md: "60%" },

left: 0,

}}

>

variant="h3"

sx={{

fontWeight: 700,

color: "white",

}}

>

Title

px={{ xs: 0, sm: 1, md: 2 }}

sx={{

position: "absolute",

top: { xs: "50%", sm: "60%", md: "70%" },

right: 0,

}}

>

variant="subtitle2"

sx={{

px: 1,

py: 0.5,

fontWeight: 700,

color: "white",

bgcolor: "red",

borderRadius: "12px 0px 0px 12px",

}}

>

12+

px={{ xs: 1, sm: 2 }}

sx={{ position: "absolute", bottom: 20, left: 0, right: 0 }}

>

{/\* Seekbar \*/}

playedSeconds={playerState.playedSeconds}

duration={playerState.duration}

seekTo={handleSeekTo}

/>

{/\* end Seekbar \*/}

{/\* Controller \*/}

{/\* left controller \*/}

direction="row"

spacing={{ xs: 0.5, sm: 1.5, md: 2 }}

alignItems="center"

>

{!playerState.paused ? (

onClick={() => {

playerRef.current?.pause();

}}

>

) : (

onClick={() => {

playerRef.current?.play();

}}

>

)}

muted={playerState.muted}

handleVolumeToggle={() => {

playerRef.current?.muted(!playerState.muted);

setPlayerState((draft) => {

return { ...draft, muted: !draft.muted };

});

}}

value={playerState.volume}

handleVolume={handleVolumeChange}

/>

{`${formatTime(playerState.playedSeconds)} / ${formatTime(

playerState.duration

)}`}

{/\* end left controller \*/}

{/\* middle time \*/}

maxLine={1}

variant="subtitle1"

textAlign="center"

sx={{ maxWidth: 300, mx: "auto", color: "white" }}

>

Description

{/\* end middle time \*/}

{/\* right controller \*/}

direction="row"

alignItems="center"

spacing={{ xs: 0.5, sm: 1.5, md: 2 }}

>

{/\* end right controller \*/}

{/\* end Controller \*/}

)}

);

}

return null;

}

Component.displayName = "WatchPage";

src/providers/DetailModalProvider.tsx

"use client";

import { ReactNode, useEffect, useState, useCallback } from "react";

import { usePathname } from "next/navigation";

import { INITIAL\_DETAIL\_STATE } from "src/constant";

import { createContext, useContext } from "react";

import { useLazyGetAppendedVideosQuery } from "src/store/slices/discover";

import { MEDIA\_TYPE } from "src/types/Common";

import { MovieDetail } from "src/types/Movie";

interface DetailType {

id?: number;

mediaType?: MEDIA\_TYPE;

}

export interface DetailModalConsumerProps {

detail: { mediaDetail?: MovieDetail } & DetailType;

setDetailType: (newDetailType: DetailType) => void;

}

const DetailModalContext = createContext(undefined);

export function useDetailModal() {

const ctx = useContext(DetailModalContext);

if (!ctx) throw new Error("DetailModalProvider is missing in the tree.");

return ctx;

}

const Provider = DetailModalContext.Provider;

export default function DetailModalProvider({

children,

}: {

children: ReactNode;

}) {

const pathname = usePathname();

const [detail, setDetail] = useState<

{ mediaDetail?: MovieDetail } & DetailType

>(INITIAL\_DETAIL\_STATE);

const [getAppendedVideos] = useLazyGetAppendedVideosQuery();

const handleChangeDetail = useCallback(

async (newDetailType: { mediaType?: MEDIA\_TYPE; id?: number }) => {

if (!!newDetailType.id && newDetailType.mediaType) {

const response = await getAppendedVideos({

mediaType: newDetailType.mediaType,

id: newDetailType.id as number,

}).unwrap();

setDetail({ ...newDetailType, mediaDetail: response });

} else {

setDetail(INITIAL\_DETAIL\_STATE);

}

},

[]

);

useEffect(() => {

setDetail(INITIAL\_DETAIL\_STATE);

}, [pathname, setDetail]);

return (

{children}

);

}

src/providers/PortalProvider.tsx

"use client";

import { ReactNode, useState, useCallback } from "react";

import { Movie } from "src/types/Movie";

import createSafeContext from "src/lib/createSafeContext";

export interface PortalConsumerProps {

setPortal: (anchor: HTMLElement | null, vidoe: Movie | null) => void;

}

export interface PortalDataConsumerProps {

anchorElement: HTMLElement | null;

miniModalMediaData: Movie | null;

}

export const [usePortal, Provider] =

createSafeContext();

export const [usePortalData, PortalDataProvider] =

createSafeContext();

export default function PortalProvider({ children }: { children: ReactNode }) {

const [anchorElement, setAnchorElement] = useState(null);

const [miniModalMediaData, setMiniModalMediaData] = useState(

null

);

const handleChangePortal = useCallback(

(anchor: HTMLElement | null, video: Movie | null) => {

setAnchorElement(anchor);

setMiniModalMediaData(video);

},

[]

);

return (

value={handleChangePortal}

>

value={{

anchorElement,

miniModalMediaData,

}}

>

{children}

);

}

src/routes/index.tsx

export {};

src/store/slices/apiSlice.ts

import { API\_ENDPOINT\_URL } from "src/constant";

import { createApi, fetchBaseQuery } from "@reduxjs/toolkit/query/react";

export const tmdbApi = createApi({

reducerPath: "tmdbApi",

baseQuery: fetchBaseQuery({ baseUrl: API\_ENDPOINT\_URL }),

endpoints: (build) => ({}),

});

src/store/slices/configuration.ts

import { TMDB\_V3\_API\_KEY } from "src/constant";

import { tmdbApi } from "./apiSlice";

type ConfigurationType = {

images: {

base\_url: string;

secure\_base\_url: string;

backdrop\_sizes: string[];

logo\_sizes: string[];

poster\_sizes: string[];

profile\_sizes: string[];

still\_sizes: string[];

};

change\_keys: string[];

};

export const extendedApi = tmdbApi.injectEndpoints({

endpoints: (build) => ({

getConfiguration: build.query({

query: () => ({

url: "/configuration",

params: { api\_key: TMDB\_V3\_API\_KEY },

}),

}),

}),

});

export const { useGetConfigurationQuery } = extendedApi;

src/store/slices/discover.ts

import { TMDB\_V3\_API\_KEY } from "src/constant";

import { tmdbApi } from "./apiSlice";

import { MEDIA\_TYPE, PaginatedMovieResult } from "src/types/Common";

import { MovieDetail } from "src/types/Movie";

import { createSlice, isAnyOf } from "@reduxjs/toolkit";

const initialState: Record> = {};

export const initialItemState: PaginatedMovieResult = {

page: 0,

results: [],

total\_pages: 0,

total\_results: 0,

};

const discoverSlice = createSlice({

name: "discover",

initialState,

reducers: {

setNextPage: (state, action) => {

const { mediaType, itemKey } = action.payload;

state[mediaType][itemKey].page += 1;

},

initiateItem: (state, action) => {

const { mediaType, itemKey } = action.payload;

if (!state[mediaType]) {

state[mediaType] = {};

}

if (!state[mediaType][itemKey]) {

state[mediaType][itemKey] = initialItemState;

}

},

},

extraReducers(builder) {

builder.addMatcher(

isAnyOf(

extendedApi.endpoints.getVideosByMediaTypeAndCustomGenre.matchFulfilled,

extendedApi.endpoints.getVideosByMediaTypeAndGenreId.matchFulfilled

),

(state, action) => {

const {

page,

results,

total\_pages,

total\_results,

mediaType,

itemKey,

} = action.payload;

state[mediaType][itemKey].page = page;

state[mediaType][itemKey].results.push(...results);

state[mediaType][itemKey].total\_pages = total\_pages;

state[mediaType][itemKey].total\_results = total\_results;

}

);

},

});

export const { setNextPage, initiateItem } = discoverSlice.actions;

export default discoverSlice.reducer;

const extendedApi = tmdbApi.injectEndpoints({

endpoints: (build) => ({

getVideosByMediaTypeAndGenreId: build.query<

PaginatedMovieResult & {

mediaType: MEDIA\_TYPE;

itemKey: number | string;

},

{ mediaType: MEDIA\_TYPE; genreId: number; page: number }

>({

query: ({ mediaType, genreId, page }) => ({

url: `/discover/${mediaType}`,

params: { api\_key: TMDB\_V3\_API\_KEY, with\_genres: genreId, page },

}),

transformResponse: (

response: PaginatedMovieResult,

\_,

{ mediaType, genreId }

) => ({

...response,

mediaType,

itemKey: genreId,

}),

}),

getVideosByMediaTypeAndCustomGenre: build.query<

PaginatedMovieResult & {

mediaType: MEDIA\_TYPE;

itemKey: number | string;

},

{ mediaType: MEDIA\_TYPE; apiString: string; page: number }

>({

query: ({ mediaType, apiString, page }) => ({

url: `/${mediaType}/${apiString}`,

params: { api\_key: TMDB\_V3\_API\_KEY, page },

}),

transformResponse: (

response: PaginatedMovieResult,

\_,

{ mediaType, apiString }

) => {

return {

...response,

mediaType,

itemKey: apiString,

};

},

}),

getAppendedVideos: build.query<

MovieDetail,

{ mediaType: MEDIA\_TYPE; id: number }

>({

query: ({ mediaType, id }) => ({

url: `/${mediaType}/${id}`,

params: { api\_key: TMDB\_V3\_API\_KEY, append\_to\_response: "videos" },

}),

}),

getSimilarVideos: build.query<

PaginatedMovieResult,

{ mediaType: MEDIA\_TYPE; id: number }

>({

query: ({ mediaType, id }) => ({

url: `/${mediaType}/${id}/similar`,

params: { api\_key: TMDB\_V3\_API\_KEY },

}),

}),

}),

});

export const {

useGetVideosByMediaTypeAndGenreIdQuery,

useLazyGetVideosByMediaTypeAndGenreIdQuery,

useGetVideosByMediaTypeAndCustomGenreQuery,

useLazyGetVideosByMediaTypeAndCustomGenreQuery,

useGetAppendedVideosQuery,

useLazyGetAppendedVideosQuery,

useGetSimilarVideosQuery,

useLazyGetSimilarVideosQuery,

} = extendedApi;

src/store/slices/genre.ts

import { TMDB\_V3\_API\_KEY } from "src/constant";

import { Genre } from "src/types/Genre";

import { tmdbApi } from "./apiSlice";

const extendedApi = tmdbApi.injectEndpoints({

endpoints: (build) => ({

getGenres: build.query({

query: (mediaType) => ({

url: `/genre/${mediaType}/list`,

params: { api\_key: TMDB\_V3\_API\_KEY },

}),

transformResponse: (response: { genres: Genre[] }) => {

return response.genres;

},

}),

}),

});

export const { useGetGenresQuery, endpoints: genreSliceEndpoints } = extendedApi;

src/store/index.ts

"use client";

import { configureStore } from "@reduxjs/toolkit";

import { tmdbApi } from "./slices/apiSlice";

import discoverReducer from "./slices/discover";

import type { Middleware } from "redux";

const store = configureStore({

reducer: {

discover: discoverReducer,

[tmdbApi.reducerPath]: tmdbApi.reducer,

},

middleware: (getDefaultMiddleware: any) =>

getDefaultMiddleware().concat(tmdbApi.middleware),

});

export type RootState = ReturnType;

export type AppDispatch = typeof store.dispatch;

export default store;

src/theme/MUIRegistry.tsx

"use client";

import \* as React from "react";

import createCache from "@emotion/cache";

import { CacheProvider } from "@emotion/react";

import { useServerInsertedHTML } from "next/navigation";

import { ThemeProvider, createTheme } from "@mui/material/styles";

import CssBaseline from "@mui/material/CssBaseline";

import palette from "./palette"; // (dosya bu klasördeyse)

function createEmotionCache() {

const cache = createCache({ key: "css", prepend: true });

cache.compat = true;

return cache;

}

const theme = createTheme({ palette });

export default function MUIRegistry({ children }: { children: React.ReactNode }) {

const [cache] = React.useState(() => {

const c = createEmotionCache();

const prevInsert = c.insert;

let inserted: string[] = [];

c.insert = (...args) => {

const serialized = args[1];

if (!c.inserted[serialized.name]) inserted.push(serialized.name);

// @ts-ignore

return prevInsert(...args);

};

// @ts-ignore

c.flush = () => {

const prev = inserted;

inserted = [];

return prev;

};

return c as typeof c & { flush?: () => string[] };

});

useServerInsertedHTML(() => {

// @ts-ignore

const names = cache.flush?.() ?? [];

if (names.length === 0) return null;

let styles = "";

for (const name of names) {

styles += cache.inserted[name];

}

return (

data-emotion={`${cache.key} ${names.join(" ")}`}

dangerouslySetInnerHTML={{ \_\_html: styles }}

/>

);

});

return (

{children}

);

}

src/theme/palette.ts

import type { PaletteMode } from "@mui/material";

const PRIMARY = {

light: "#B8B8B8",

main: "#141414",

dark: "#0E0A0A",

};

const GREY = {

100: "#F9FAFB",

200: "#F4F6F8",

300: "#DFE3E8",

400: "#C4CDD5",

500: "#919EAB",

600: "#637381",

700: "#454F5B",

800: "#212B36",

900: "#161C24",

};

const COMMON = {

common: { black: "#000", white: "#fff" },

primary: { ...PRIMARY, contrastText: "#fff" },

grey: GREY,

action: {

active: GREY[500],

hoverOpacity: 0.08,

disabledOpacity: 0.48,

},

};

const palette = {

...COMMON,

text: { primary: "#fff", secondary: GREY[500], disabled: GREY[600] },

background: { default: PRIMARY.main, paper: PRIMARY.main },

mode: "dark" as PaletteMode,

};

export default palette;

src/types/Common.ts

"use client";

import { Movie } from "src/types/Movie";

export enum MEDIA\_TYPE {

Movie = "movie",

Tv = "tv",

}

export type Company = {

description: string;

headquarters: string;

homepage: string;

id: number;

logo\_path: string;

name: string;

origin\_country: string;

parent\_company: null | object;

};

export type Country = {

iso\_3166\_1: string;

english\_name: string;

};

export type Language = {

iso\_639\_1: string;

english\_name: string;

name: string;

};

export type PaginatedResult = {

page: number;

total\_pages: number;

total\_results: number;

};

export type PaginatedMovieResult = PaginatedResult & { results: Movie[] };

src/types/Genre.ts

"use client";

export type Genre = {

id: number;

name: string;

};

export type CustomGenre = {

id?: number;

name: string;

apiString: string;

};

src/types/Movie.ts

"use client";

import { Company, Country, Language } from './Common';

import { Genre } from './Genre';

export type Appended\_Video = {

id: string;

iso\_639\_1: string;

iso\_3166\_1: string;

key: string;

name: string;

official: boolean;

published\_at: string;

site: string;

size: number;

type: string;

};

export type MovieDetail = {

adult: boolean;

backdrop\_path: string | null;

belongs\_to\_collection: null;

budget: number;

genres: Genre[];

homepage: string;

id: number;

imdb\_id: string;

original\_language: string;

original\_title: string;

overview: string;

popularity: number;

poster\_path: string | null;

production\_companies: Company[];

production\_countries: Country[];

release\_date: string;

revenue: number;

runtime: number;

spoken\_languages: Language[];

status: string;

tagline: string;

title: string;

video: boolean;

videos: { results: Appended\_Video[] };

vote\_average: number;

vote\_count: number;

};

export type Movie = {

poster\_path: string | null;

adult: boolean;

overview: string;

release\_date: string;

genre\_ids: number[];

id: number;

original\_title: string;

original\_language: string;

title: string;

backdrop\_path: string | null;

popularity: number;

vote\_count: number;

video: boolean;

vote\_average: number;

};

src/utils/common.ts

"use client";

export const getRandomNumber = (maxNumber: number) =>

Math.floor(Math.random() \* maxNumber);

export const formatMinuteToReadable = (minutes: number) => {

const h = Math.floor(minutes / 60);

const m = minutes - h \* 60;

if (h > 0) {

return `${h}h ${m}m`;

} else {

return `${m}m`;

}

};

export const formatBytes = (bytes: number, decimals: number = 2) => {

if (!+bytes) return "0 Bytes";

const k = 1024;

const dm = decimals < 0 ? 0 : decimals;

const sizes = [

"Bytes",

"KiB",

"MiB",

"GiB",

"TiB",

"PiB",

"EiB",

"ZiB",

"YiB",

];

const i = Math.floor(Math.log(bytes) / Math.log(k));

return `${parseFloat((bytes / Math.pow(k, i)).toFixed(dm))} ${sizes[i]}`;

};

export const formatTime = (current: number) => {

const h = Math.floor(current / 3600);

const m = Math.floor((current - h \* 3600) / 60);

const s = Math.floor(current % 60);

const sString = s < 10 ? "0" + s.toString() : s.toString();

const mString = m < 10 ? "0" + m.toString() : m.toString();

if (h > 0) {

const hString = h < 10 ? "0" + h.toString() : h.toString();

return `${hString}:${mString}:${sString}`;

} else {

return `${mString}:${sString}`;

}

};

src/utils/index.ts

"use client";

export { formatMinuteToReadable, getRandomNumber } from "./common";

function buildThresholdList() {

let thresholds = [];

const numSteps = 20;

for (let i = 1.0; i <= numSteps; i++) {

let ratio = i / numSteps;

thresholds.push(ratio);

}

thresholds.push(0);

return thresholds;

}

export { buildThresholdList };

src/next-env.d.ts

///

///

///

///

// NOTE: This file should not be edited

// see https://nextjs.org/docs/app/api-reference/config/typescript for more information.

src/next.config.ts

import type { NextConfig } from "next";

const nextConfig: NextConfig = {

/\* config options here \*/

};

export default nextConfig;

src/vite.config.ts

import { defineConfig } from 'vite'

import react from '@vitejs/plugin-react'

import tsconfigPaths from 'vite-tsconfig-paths'

// https://vitejs.dev/config/

export default defineConfig({

plugins: [react(), tsconfigPaths()]

})