Mpa:(multiple page app): با تغییر در path یکبار صفحه به طور کامل ریلود میشود.از نظر پرفورمنس ضعیف هستند.

Spa(single page app): کل صفخه یکبار لود میشود و با تغییر path فقط محتوای صفحه بدون ریلود شدن عوض میشود.

نکته: به هر مسیر یک route گفته میشود و برای مدیریت این ها از react router استفاده میکنیم.

React router: نصب npm install react-router-dom

از Routes برای تعریف روت ها و برای هر روت از Route استفاده میکنیم که به صورت prop برای آن مقادیر path برای تعیین کردن مسیر مثلا /users و element برای مشخص کردن چیزی که قرار است نمایش داده شود که معمولا یک کامپوننت است.کل آن کامپوننت که درآن Routes را تعریف کردیم باید داخل BrowserRouter قرار بدهیم.

Link: لینک هایی که با تگ a میسازیم باعث ریلود شدن صفحه میشوند ولی برای spa بودن سایت نباید این اتفاق بیوفتد پس از Link که از reat-routert-dom ایمپورت میشود استفاده میکنیم.به عنوان prop با to مسیر را مشخص میکنیم.با children المنت یا متنی که قرار است داخل لینک نمایش بدهیم مشخصی میکنیم.با state میتوانیم یک مقدار را به صفحه بعدی که قرار است باز شود منتقل کنیم.

NavLink: تمام موارد بالا را دارد علاوه بر آن یک prop به نام end دارد که مشخص میکند الان داخل مسیر همان لینک هستیم برای استایل دهی لینک فعال استفاده میشود.در className به یک مقدار به نام Link دسترسی داریم که یک ابجکت است و مقدار isActive را دارد که به صورت شرط میتوانیم استایل مد نظرمان را مشخص کنیم.

className={{isActive} => isActive ? active : deactive }

useParams: در تعریف مسیر میتوانیم با : یک پارامتر تعریف کنیم مثلا => /posts/:id برای استفاده از این پارامتر در کامپوننت مقصد از هوک useParams استفاده میکنیم که یک آبجکت از تمام پارامتر ها به ما میدهد.

Navigate: از این کامپوننت تگ برای redirect کردن کاربردر کد های jsx استفاده میکنیم که بدون ریلود شدن این کار را میکند و به to مسیر را مشخص میکنیم.

404: یک Route با مقدار \* اضافه میکنیم و اگر کاربر وارد هر مسیر دیگری که در روت های پروژه نباشد وارد آن کامپوننت میشود که میتوانیم بسازیم.

subRoutes: اگر مسیرمان به این باشد مثلا /users/profile به قسمت پروفایل یک subRoute از users میگویند.به این صورت که ابتدا کامپونتت users رندر میشود و پس از آن کامپوننت مربوط به profile رندر میشود.برای این کار ابتدا در Routes اصلی یک مسیر به صورت /users/\* مینویسیم که برای هر ساب روتی کامپوننت user اجرا بشود سپس داخل خود کامپوننت users یک Routes جدا مینویسیم و Route ها را مشخص میکنیم. برای نوشتن path در این حالت نباشد در ابتدا / بذاریم چون از root پروژه حساب میشود.

Outlet: در روش قبلی نیاز هست در جاهای مختلف پروژه Routes تعریف کنیم که از خوانایی پروژه می کاهد.به جای آن میتوانیم در Routes اصلی پروژه از تگ Route به صورت باز و بسته استفاده کنیم و subRoutes را داخل آن بنویسیم. در این حالت نباید از \* در Route اصلی استفاده کنیم و ساب روت ها را باید بدون / در ابتدا بنویسیم.در جایی که میخواهیم sub رندر بشه باید از Outlet استفاده کنیم.

useRoutes: باید از react-router-dom ایمپورت بشود و به عنوان ورودی یک لیست از آبجکت ها میگیرد که دو مقدار path ,element مانند قبل باید برایش ارسال کنیم subRoute ها را هم با کلید children به صورت یک آرایه مینویسیم.در jsx هم هر جا که میخواستیم رندر شدن روتر ها انجام شود از متغیری که ساختیم استفاده میکنیم با نام دلخواه router.برای خوانایی بیشتر تمام route ها را داخل فایلی به نام routes.js در src مینویسیم.

Public route: به روت هایی گفته میشود که برای همه ی کاربران قابل دسترسی هستند.

Private route: به روت هایی گفته میشود که برای کاربرانی با شرط خاص قابل نمایش است و اگر کاربر ان شرط را نداشته باشد redirect میشود به یک صفحه ی دیگر.به این صورت عمل میکنیم که ابتدا یک تابع برای بررسی یک شرط مثلا لاگین بودن کاربر یا ادمین بودن کاربر مینویسیم و با شرط های تک خطی بررسی میکنیم اگر مقدار true از تابع برگشت محتوا را نشان میدهیم و در غیر اینصورت با استفاده از Navigate کاربر را به صفحه ی دیگری میفرستیم یا پیامی را به کاربر نشان میدهیم.

روش بهینه تر این است که یک کامپوننت با اسم دلخواه مثلا PrivateRoute بسازیم که در prop هایش children را دریافت کنیم که میشود همان محتوای بین کامپوننت تگ حالا همان شرط لاگین بودن را بررسی میکنیم و اگر true بود محتوای children را نمایش میدهیم و در غیر اینصورت redirect میکنیم حالا در فایلی که تمام روت ها را نوشتیم هر روتی که خواستیم Private باشد را بین تگ های باز و بسته ی PrivateRoute مینویسیم.

روش بهتر این است که کامپوننت privateRoute را به عنوان path \* بنویسیم و تمام روت های private را به صورت subRoute در children براش تعریف کنیم.در کامپوننت PrivateRoute هم به جای محتوای children باید Outlet را نمایش بدیم.

useNavigate: برای شرایطی که بخواهیم در داخل تابع با استفاده از شرط ها کاربر را redirect کنیم میتوانیم از این هوک استفاده کنیم.ایمپورت از react-router-dom یک متغیر با استفاده از این هوک تعریف میکنیم و سپس هر جا خواستیم روت مورد نظر را به آن میدهیم تا redirect کند.اگر ورودی را -1 بذاریم به صفحه قبل از خودش هر چی که باشد برمیگردد.هر عدد منفی که بذاریم به همان تعداد به روت قبلی میرود.ورودی دوم هم به صورت یک ابجکت میتواند تنظیماتی بگیرد مثلا اگر replace: true بذاریم وقتی ریدایرکت میوشد از history مرورگر هم حذف میشود.

useLocation: برای دسترسی به اطلاعات لوکیشن مانند pathname, search, hash و غیره بکار میرود.

Jsx format: در فایل هایی که کد jsx مینویسیم بهتر است از فرمت jsx استفاده کنیم مثلا test.jsx

Recharts: برای استفاده از نمودارها در react استفاده میشود که نمودار های متنوعی دارد. npm install recharts

Custome hook: بعضی وقتا توی پروژه نیاز به هوک هایی داریم که در ری اکت وجود ندارد در این شرایط میتوانیم هوک اختصاصی خودمان را بنویسیم که از تکرار کد جلوگیری کنیم یک تابع ساده با نام use\_\_\_ مینویسیم و عملکرد مد نظر مثل اعتبار سنجی ورودی کاربر را انجام میدهیم و نتیجه را return میکنیم. و در کدمان مانند بقیه هوک ها استفاده میکنیم.تمام هوک ها را در یک دایرکتوری با نام hooks مینویسیم و اگر نیاز بود در فایل های jsجدا مینویسیم و ایمپورت میکنیم. هوک باید مستقیماً در سطح کامپوننت اجرا شود (نه داخل تابع یا اکشن).