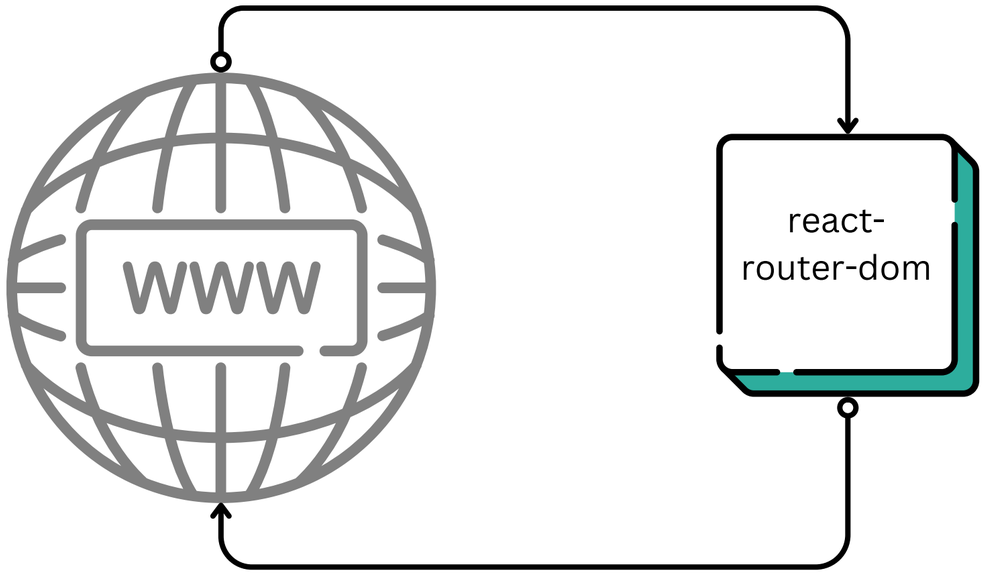
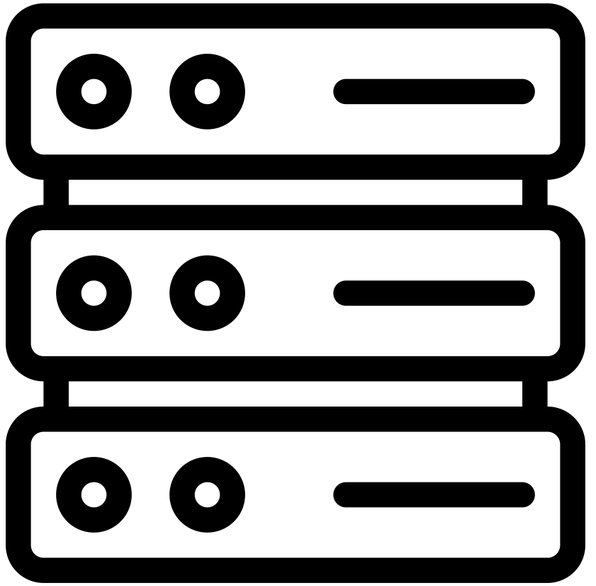
BROWSER ROUTER

single page application: this feature is achieved in react in this react wont send request to server unwantedly instead of requesting server we use “react-router-dom” so here when request is made by client react-router-dom intercepts the req and checks the url and swaps the page which is needed

State





server

OLDER WAY ROUTING

1. BrowserRouter:
   BrowserRouter is a component which routes our entire component and allows us to use Routes and Route components within it
2. Routes:
   thise is the important component that used for routing
   Routes component is the parent component for multiple Route components
3. Route:
   Route should be wrapped withing the Routes every time
   every route in the application is defined using Route component
   it has two props 1. path='our desired path'
   2. element={component to show when visited that route}

BrowserRouter

code:

```
<BrowserRouter>
    component
<BrowserRouter/>
```

any component that is wrapped in browser component can use routing features of react-router-dom

in this code the component we wrapped inside BrowserRouter can use react-router-dom features

Routes & Route

code:

```
<Routes>
    <Route path='/' element={component}/>
<Routes/>
```

instead of path='/' we can just use index

Routes component is used within the main component and its children that is wraped inside BrowserRouter so here

Routes is a parent component to Route which is used to store all children components of Route

Route component defines the path and which element to render when requested

Link & NavLink

code:
<Link to='somepath'>any element<Link/>
<NavLink to='somepath'>any element<NavLink/>

when using Link component it takes props as to='route' it navigates or redirects our url to that path and if theres any Route defined to that path then the element to that route will be displayed

for NavLink diffrence is it adds a special class "active" when clicked so that we can use that to style the active clicked link
in case of Link its not like that

NEWER WAY ROUTING

1. in case of older way we used readymade BrowserRouter but in this newer way we will create the BrowserRouter ourself as custom and use it
2. Important components used here are:

- createBrowserRouter
- createRoutesFromElements
- RouterProvider
- Route
- Routes
- Link
- NavLink

createBrowserRouter

code:

```
const router = createBrowserRouter(
        createRoutesFromElements(
                //just give Routes and Route here
                <Route> // same as Routes
<Route path='ourPath' element={compnt}/>
<Route path='nxtPath' element={component}
                <Route/>
        )
)
```

we create BrowserRouter using createBrowserRouter and store it in router and in createBrowserRouter we design this routes tree using Routes and Route components

we will just use Route as usual in this newer version too
instead of wrapping Route components in Routes we just use Route here also

RouterProvider

code:

```
 <RouterProvider
router={customMadeBrowserRouter}
/>
```

we use RouterProvider component to include BrowserRouter we created
to include custom BrowserRouter we pass it to RouterProvider component using props named “router”

for NavLink diffrence is it adds a special class “active” when clicked so that we can use that to style the active clicked link
in case of Link its not like that

navbar

code:RootLayOut.js

```
return (
        <NavBar/> // navbar component
         <Outlet/> // this is the place where Route
places its element requested using route
)
const router = createBrowserRouter(
 createRoutesFromElements(
 <Route path='/' element = {<RootLayOut/>}>
<Route path='ourPath' element={compnt}/>
<Route path='nxtPath' element={component}
 <Route/>) )
```

in olderVersion we included navbar inside BrowserRouter but here we can't do that coz we are creating that soo

SOLUTION:

we use layouts

layout is also a component which tells Route to where to place the element retrived from its path

in this layout we include nabar at top and tell Route to place element retrived from its path below using "Outlet"

Outlet

code:

```
import {Outlet} from 'react-router-dom'
return (
        <NavBar/>
        <Outlet/>

)
```

Outlet is a predefined component imported from react-router-dom which is used to help Route where to place its element in the layout given to parent Route

nestedroutes

code:

```
const router = createBrowserRouter(
 createRoutesFromElements(
 <Route path='/' element = {<RootLayOut/>}>
<Route Index element={<Home/>}/>
<Route path='about' element={<About/>}>
        <Route path='faq' element={<Faq/>}/>
<Route/> <Route/>) )
```

for Faq component to send into rootOutLet
it should match path /about/faq
it automatically adds its parent routes to its route

in the beside example we nested Route inside Route

1. Route with path '/' is main parent
2. Route sends and places <Home/> component in place of Outlet inside RootLayOut
3. Route sends and places <About/> component in place of outlet in RootLayOut here it itself appends routes we just gave path='about' but it appends / before about and sends About component only that path matches

REFERENCE:https://www.youtube.com/playlist?list=PL4cUxeGkcC9iVKmtNuCeIswnQ97in2GGf

BY: HEMA VARDHAN

2024