FE-9733P

Hedefimiz

JSON Placeholder API'nin gelişmiş bir arayüzünü React'ta yapalım

https://jsonplaceholder.typicode.com

Koşullar ve bilgilendirmeler

- Bu projede birden fazla sayfamız olacak
- Global state, yerel depolama alanı gibi yapıları kullanmamız gerekecek

Adımlar

1. İlk önce projemize react-router kurup "/users" endpoint'ini ana sayfamıza istediğimiz şekilde bir tasarım ile bağlayalım
    a. Bu endpoint'te kullanıcılar listelenecek

    b. API'den verileri çekmek için react-router'ın data loader'ını kullanalım, normal useEffect yöntemi ile yapmayalım

2. Az önce bağladığımız endpoint, bir "index" endpoint'iydi yani birden fazla verinin listelendiği bir endpoint'ti. Bu endpoint'i kullanarak kullanıcıların listelendiği bir sayfa oluşturmuştuk. Şimdi ise tek bir kullanıcının gösterildiği bir sayfa oluşturalım
    a. Bu sayfa için projemizde gireceğimiz local adresimiz "/users/:userId" olsun yani ":userId" kısmı dinamik olsun; oraya 1, 2, 3, 4 gibi kullanıcı id'leri girelim
    b. Adresimizdeki ":userId" kısmına göre JSON Placeholder'dan dinamik bir şekilde veriyi çekelim
        i. JSON Placeholder API'ye "/users/1", "/users/2" şeklinde istek atacağız
3. Oluşturduğumuz tekli user gösterme sayfasına bir tablist koyup bu tablist'in içerisinde her sekmede ayrı ayrı o kullanıcıya ait post'ları, album'leri ve todo'ları göstertelim
    a. Sadece o kullanıcıya ait olan post, album ve todo'ları gösterelim
    b. Bunlara ilişkin verileri sayfa yüklenirken react-router'ın loader'ı ile atmamıza gerek yok, useEffect ile o sekmeye girince istek attırabiliriz
        i. Böylece sayfa açılışı sırasında tüm sekmelerin verilerini çekmeyip, sayfanın içerisindeyken sadece sekme aktif olduğunda aktif sekmenin verilerini çektiğimiz için her sekmenin verilerini beklemeyiz, böylece sayfa açılışı daha hızlı olur
    c. Sekme verileri yüklenirken "Yükleniyor..." tarzı bir uyarı veya ikon vs. gösterelim
4. Projemizin tekli user sayfasındaki herhangi bir post'a tıkladığımızda, kendi projemiz içerisinde "/users/:userId/posts/:postId" sayfasına yönlendirsin
    a. Bu sayfayı oluşturup rotalarımıza ekleyelim. Burada ilk başta post'u sayfanın başında gösterelim, hemen altında ise o post'un yorumları (comments) olsun
        i. Yorumları çekmek için örnek JSON Placeholder API endpoint'i: "/posts/1/comments"

b. Post'un tasarımının içine, post'un sahibinin (user) kullanıcı adını da koyalım. Tıklayınca, post'un sahibinin profiline gitsin

c. user, post ve comments verilerini react-router'ın dataloader'ında API'dan çekelim

5. Tekli user sayfasındaki herhangi bir album'e tıkladığımızda, kendi projemiz içerisinde "/users/:userId/albums/:albumId" sayfasına yönlendirsin

   a. Bu sayfayı oluşturup rotalarımıza ekleyelim. Burada ilk başta album'u sayfanın başında gösterelim, hemen altında ise o album'un fotoğrafları olsun

      i. Fotoğrafları çekmek için örnek JSON Placeholder API endpoint'i: "/albums/1/photos"

   b. Album'ün tasarımının içine, album sahibinin (user) kullanıcı adını da koyalım. Tıklayınca, album'un sahibinin profiline gitsin

   c. Bu sayfada yüklenmesi gereken verileri react-router'ın dataloader'ı aracılığıyla çekelim

6. Tekli user sayfasındaki todo'lara özel bir sayfa yapmamıza gerek yok çünkü onlara bağlı bir şey yok

   a. Post'a comments bağlıydı, album'e ise photos bağlıydı ama todo'ya bağlı bir şey yok, o yüzden sayfayı ayırma ihtiyacımız da yok

7. Şimdi, photo'ları favorilere ekleme yapısı ekleyelim

   a. Bir album'ün içinde listelenen photo'ların her birine bir favorilere ekleme butonu koyalım

      i. Kalp ikonu tarzı bir şey olabilir, tıkladığımızda favoriler listesine ekleteceğiz

   b. Favorilere ekleme butonuna tıkladığımızda, global bir "favorites" store'unun içerisindeki "photos" array'ine; favorilere eklenen photo'nun bilgilerini, album sahibi kullanıcının userId'si ile birlikte kaydedelim. Örnek:

```
{
        "userId": 1,
    "albumId": 1,
```

```
    "id": 1,
    "title": "accusamus beatae ad facilis cum similique
qui sunt",
    "url": "https://via.placeholder.com/600/92c952",
    "thumbnailUrl": "https://via.placeholder.com/150/92
c952"
}
```


   c. Bir fotoğrafı favoriye eklediysek, o fotoğraf render edilirken favorilere ekleme butonu farklı bir şekilde gözüksün. Yani favoriye eklendiği belli olsun
      i. Mesela kalp ikonu ile belirttiysek normalde favoriye ekli değilken kalp ikonu gri iken favoriye eklediğimizde kalp ikonu kırmızıya dönüşebilir. E-ticaret sitelerindeki gibi

8. Sitemizde her sayfada gözükecek olan bir navbar ekleyelim
   a. Bunun için layout oluşturup her sayfayı o layout'un içine almamız gerekecek
   b. Navbar'da favorites kısmı olsun ve orada da bir sayaç gözüksün. O sayaç, kaç fotoğrafı favorilerimize eklediğimizi sayıyor olsun ve global state olarak tuttuğumuz favorites store'unun altındaki photos verisine göre sayım yapsın
9. Favorilere eklenen fotoğrafları göstermek için bir sayfa oluşturalım.
   a. Orada, bu fotoğrafları listeleyelim
   b. Fotoğrafa tıkladığımızda, tıkladığımız fotoğrafın album sayfasına gidelim
   c. Favorilerden kaldırmak için buton tarzı bir şey koyalım
10. favorites store'unu localstorage'a kayıt edelim (persist)
   a. Bu sayede uygulamamızı kapatıp açsak dahi verilerimiz kaybolmayacak
   b. Bunun için zustand'ın localstorage entegrasyonunu kullanalım
11. Post'ları da favorilere eklenebilir yapalım

a. Post’u, favorites store’u altında “posts” array’ine ekleyecek şekilde ayarlamalar yapalım
b. Arayüz’de, post’u favoriye ekleme ve favoriden kaldırma butonu koymamız gerekecek

Yardımcı Kaynaklar

- https://reactrouter.com/en/main/start/tutorial
- https://react-bootstrap.netlify.app/docs/components/tabs
- https://github.com/pmndrs/zustand/blob/main/docs/guides/typescript.md