FE-4177P

Hedefimiz

Sürükle-bırak yapmayı destekleyen bir yapılacaklar listesi oluşturalım

Düşünce Yapısı

Bu projede düşünce yapımız, halihazırda elimizde olan kodun üzerinde ufak değişiklikler yaparak olabildiğince kısa sürede istediğimiz özellikleri projemize eklemek üzerine olmalı.

Kodların hepsini oturup teker teker okumaya, anlamaya çalışmaya, baştan yazmaya ihtiyacımız yok. **Sadece değiştirmek istediğimiz kısım neresi ise oraya odaklanarak işi çözmeye çalışmalıyız.**

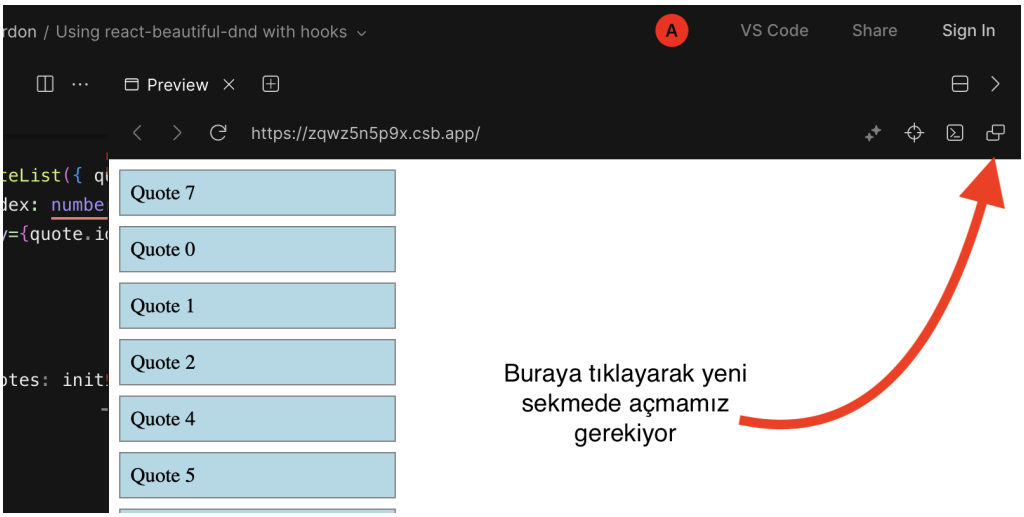
Satır satır her şeyi kendimiz en baştan anlaya anlaya oluşturmak vaktimizi verimli kullanmamız açısından dezavantajımıza olacaktır.

Bu projedeki odağımız başka projelerden kodlar alıp kullanabilmek olacak. İngilizce bilmesek dahi (siz biliyor olabilirsiniz, sadece örnek veriyorum) Google Translate eklentisi gibi araçlar sayesinde İngilizce yazılım sayfalarını parça parça seçip

çevirebiliyorsak, yani İngilizce'yi tam bilmememize rağmen onu kullanmayı biliyorsak aynı şekilde bu düşünce yapısını bu projede de kullanmalıyız. Elimizde elde etmek istediğimiz bir özellik var, parça parça iki farklı projede bulunuyor, bunları alıp birleştirip üstüne kendimizce eklemek istediğimiz ek bir özellik koyacağız. İhtiyacına yakın bir kod bulup değiştirerek kullanmak birçok tecrübeli yazılımcının gerçekleştirdiği bir işlemdir. Çoğu zaman baştan yazmaktan çok daha kısa sürer bu işlem.

Koşullar ve bilgilendirmeler

1. Vite ile bir React-TypeScript projesi oluşturalım
    - Projeyi temizleyelim, kullanmayacağımız kodları silelim. Boş bir proje çıksın karşımıza
2. Projemizde yapılacaklar listesini sürükle-bırak yaparak sıralayacağız. Sürükle bırak yapmak için "react-beautiful-dnd" kütüphanesini kullanacağız
    - Google'dan arama yaparak kütüphanenin GitHub sayfasını bulalım
3. Bulduğumuz kütüphaneyi pnpm ile projemize kuralım
    - GitHub sayfasındaki "Installation" kısmına bakabiliriz (Ctrl + F → "installation")
4. "Examples and samples" kısmından örnekleri inceleyelim
    a. "*Using with function components*" örneğini inceleyelim:
        i. Bu örnekte bir liste var, sürükle bırak yaparak listenin elemanlarının sıralarını değiştirebiliyoruz
        ii. Örnekteki projenin state yapısını; React Devtools eklentisinin bizim tarayıcımızdaki Chrome Devtools sekmeler listesine eklemiş olduğu "Components" sekmesinden inceleyelim
            - Düzgün bir şekilde inceleyebilmek için öncelikle kodun çıktısını yeni sekmede açalım:



- React Devtools'un sağladığı "Components" sekmesinde "QuoteApp" ana component'ini seçtikten sonra sağ tarafta çıkan panelden hooks kısmındaki state'i görebiliriz
    - Bu state yapısını inceleyelim. State'in kod kısmında oluşturulurkenki kullanımını da inceleyelim.
        - Demek ki kod kısmında "initial" değişkeninda yapılan işlemler "React Devtools"ta görüntülediğimiz o state çıktısını oluşturuyormuş

iii. Oradaki liste kodlarını kendi projemize dahil edelim

- js dosyası oluşturup içinde bazı yerlerde TypeScript kodları yazmışlar, muhtemelen dikkatsizlik sonucu olmuş, biraz temizlik yapalım
- Stil vermek için CSS in JS teknolojisi kullanılmış, bu örnekte "styled-components" yerine "emotion" paketi tercih edilmiş. Biz aynı örneği çok ufak değişikliklerle "styled-components" ile yapalım
    - Zaten iksinin de yazılış şekli birbirine çok benziyor, "emotion" import'unu "styled-components" import'uyla değiştirsek muhtemelen yeterli olacaktır

iv. Kodları aldığımız kaynak proje bir JavaScript projesi. Bizim projemiz ise bir TypeScript projesi. Hata yaşamamamız için bazı yerleri kendimiz TypeScript koduna uyarlamamız gerekecek

v. State'i şu an obje içinde array olarak tutuyoruz, bu veri yapısını değiştirip direkt liste elemanlarından oluşan bir array içerisinde tutacak şekilde ayarlayalım

- Yani `{quotes: []}` şeklinde olmasın, direkt quotes kısmının değerini state'te tutalım. Obje altından erişmeyelim, state'e direkt array'in kendisini kayıt edelim

vi. Kendi projemizde o listeyi çalışabilecek bir hale sokalım

b. Ardından "*Simple DnD between a dynamic number of lists (with function components) and ability to delete items*" örneğini açalım:

- İlk önce yine React Devtools ile bu örneğin state'lerini inceleyelim
- Bu örnekte silme butonu var. Silme butonunun tetiklediği fonksiyonu inceleyelim
  - Fonksiyonun çalışma mantığını anlayıp kendi projemize de aynı işe yarayan buton ekleyelim
  - İki proje arasındaki farklılıkları, projeleri yan yana açıp farklı kısımları kopyala yapıştır yaparak, değiştirerek inceleyebiliriz. Dönen işlemleri anlamak için biraz etrafı kurcalayalım, ufak ufak kendi istediğimiz özelliği sağlayacak hale getirmeye çalışalım
  - Böylece yapılacaklar listemizdeki öğelerimizi silebilme özelliği elde ediyoruz

5. Şimdi ise listeye yeni elemanlar ekleme özelliği oluşturmalıyız:

- Listenin üstüne bir input ve bir buton koyalım. Input'a yazdığımız yazıyı butona bastıktan sonra gereken yapıya sokarak listeye (state'imize) ekleyelim
  - Yapıyı zaten React Devtools aracılığıyla inceleyip gördük, aynı türden bir veriye dönüştüreceğiz sadece
  - Yazıyı bir quote objesine çevirip öyle state array'imize eklemeliyiz
  - id'yi "nanoid" gibi paketler aracılığıyla rastgele oluşturmayı da deneyebiliriz, illaki state'teki önceden oluşturulmuş olan verilerdeki gibi sıralı olmasına pek gerek yok

6. Referans projeden aldığımız “reorder” fonksiyonumuzun ve “initial” değişkenimizin kodlarını açıklayalım, işleyişini anlamaya çalışalım
    - Bu konuda “ChatGPT” gibi yapay zeka dil modellerinden de destek alabiliriz
        - Kodu gönderip “şu kısım ne işe yarıyor satır satır detaylı açıklar mısın” tarzı bir soru sorduğumuzda bize yardımcı olacaktır
7. Liste, sayfa açıldığında otomatik olarak oluşturuluyor ve biz de bu listeye ilave yapabiliyoruz. Ancak bu durum şu an bizim için biraz anlamsız
    - Sayfa açıldığında listemiz boş bir şekilde başlasın, biz verileri sıfırdan ekleyelim. Zaten ekleme özelliğini 5. adımda geliştirmiştik
8. Projemizin arayüzünü kendimizce özelleştirelim, gerçek bir projeye dönüştürmeye çalışalım

BONUS ADIM: Liste elemanlarına tıpkı silme butonu gibi bir de düzenleme butonu ekleyelim

- Butona tıklayınca bir dialog/modal açıp (bunun için react-bootstrap kullanılabilir) onun içine koyduğumuz bir input aracılığıyla state’i değiştirtmeyi deneyebiliriz
- Butona tıklandığında aslında bir seçim yapmış oluyoruz: modal içerisinde düzenlemek üzere göstereceğimiz liste elemanını seçiyoruz
    - Yani hangi liste elemanını seçtiğimizi, o elemanın index’ini veya id’sini bir state’e kaydederek (editDialogTodoId-setEditDialogTodoId gibi) eğer bu state null değilse diyalogu ona göre gösterip (`const isEditDialogOpen = editDialogTodoId !== null` gibi) diyalog içindeki input’u da mevcut seçili liste elemanını değiştirecek şekilde ayarlayabiliriz

Yardımcı Kaynaklar

Aşağıdaki konuları araştırmak, uygulamamızı geliştirirken işimize yarayabilir

- 🔑 ”js remove from array by index”
- 🔑 ”react state add item to array”