**Node.js** : Node ,JavaScript kodunun serverda çalıştırılmasından tutun paket yönetimi ve bizim için geliştirme ortamında bır yayın ortamı imkanı sunuyor.

**React Component Mimarisi**

Biz uygulamamızı componentlere bölüyoruz.Yani component demek bileşen demektir.React bir single page application yanı tek sayfa uygulamasıdır.Aslında bizim tek bir sayfamız var.Bu sayfamız aslında bızım için bir component görevi görecek.Ardından ben bu componentin içerisinde örneğin bir menü kısmım var.Oraya onu bir component olarak oluşturuyorum.Sonra sol menüm var,onun içinde bir component oluşturuyorum.Sonra Menü kısmında bir sisteme giriş sistemden çıkış kısmı oluşturmak ıstıyorum oda bır component.Yani bunun sınırı yok. Bu şekilde sayfalarımızı yapılandırıyoruz.Mesela A ve B componentimiz var. Ben A componentinden B componentine veri taşıyorum . Ama B componentinden A ya taşıyamıyorum. Buna **One-Binding** deriz.

**Virtual Dom** :

**Componentlerin kendi Yapısı**

Componentler bir kaç şekilde oluyor.Bunlar Fonksiyon componentler,class componentler birde React Hooks denilen yapı ile oluşturulan componentler var.

**JSX** :

**ReactStrap**: projeme nasıl dahil ederim?

İlk **Yöntem** : package.json dosyasında bağımlılıklara gelip el ile eklemek



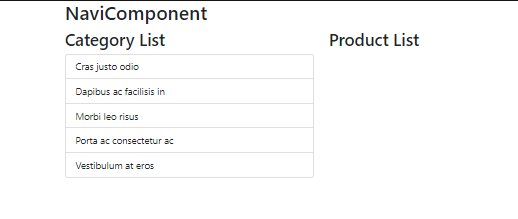
Ve termialden : npm install yazmak.Bu Dependencies e gir ve tüm paketleri güncelle demek.

**İkinci Yöntem** :Terminalden npm install reactstrap yazmak .Package.json a ,bu paket bilgisini ekleyecek.Bu paketler node\_modules altına kuruluyor.Bunları yaptıktan sonra uygulamamın içine nasıl ekleyeceğim?

İndex.js e --- > bootstrap i import ederek ekleyebilirim.

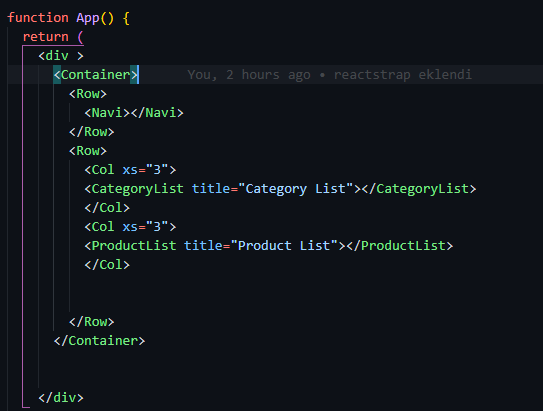
import 'bootstrap/dist/css/bootstrap.min.css';

**Props**:props a neden ihtiyac duyuyoruz?

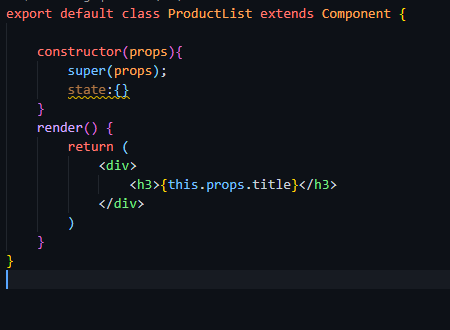


Ben CategoryList ve ProductList in title bilgisini NaviComponentten almak istiyorum.Kısacası amacım bir componentten diğerine data taşımak. Bu durumda karşımıza prop çıkıyor. Props(Propertys).Peki bunu nasıl yapıcağız?

App.js



Sonra ProductList.js e git. Constructor ile props ı ekle.Aslında Constructor a gerek yok . this.props ile de erişilebilir.



**Encapsulation**:Yeni bir eleman geldiği zaman child componente gönderilen değeri bir model gibi düşünüp onun içerisinde herhangi bir yerde değişiklik yapmamak için bu işlemi kullanırız.

**State**:Props gibi veri alışverisinde kullanılıyor.Props bir componentten diğerine taşınan data.Bir componentten diğer componentte bir değişken taşımak istiyorum veya bir fonksiyon props kullanırım.State ise bir componentin datasıdır.Bir coponente özel bir data tutmak istiyorsak onu state ile sağlarız. Örneğin CategoryList in datası Category Componentinin içerisinde lazım.Yanı bu data tekbır data ve tamamen Category componente özel bir data.İşte bu durumda categori datasını app.js (ana component)ten de taşıyabilirim props ile .Ama yok ben bunu categori içeriisnde direkt tutmak istiyorum,herhangi bir yerden taşımak istemiyorum.

**Eventler**:örn onClick() eventi

<ListGroup>

      {

 //map : listenın elemanlarını tek tek döner,döndüğü elemanları yenı bir liste yapar.

  this.state.categories.map(category =>

 (

  <ListGroupItem onClick={this.setState({currentCategory:category.categoryName})} key={category.categoryId}>{category.categoryName}</ListGroupItem>

  ))

 }

</ListGroup>

**setState**:Statede ki herhangi bir nesnenin değiştirilmesini sağlar.

**React Yaşam Döngüsü**

Önce componenetler yerleşiyo(componentDidMount) sonra render lar çalışıyor.

**React Routing**: