## 315. JSON server ile Fetch

//Kategorilerin api den getirilmesi

    getCategories = () =>{

        fetch("http://localhost:3000/categories")

        .then(response=>response.json())

        .then(data=>this.setState({categories : data}))

    }

## 316.Ürünlerin Listelenmesi (productList)

<https://reactstrap.github.io/components/tables/> kopyalıyoruz.

Burda listelenen ürünler, seçilen kategoriye göre listelenmiş olarak gelmeli.

Seçilen kategoride, app.js de mevcut, ozamn productlist datalarını, app.js te filtreleyip gönderiyoruz.

Redux’a geçince bu saçmalıklar sona ericek.

## 318. Şartlı ifadeler ile çalışmak

Reactstrap te *<ListGroupItem> active key ile actif olanın* üzerine işlem uygulamamıza olanak sağlar.

## 319. Sepete Ürün Eklemek

Burda çok havalı şeyler oldu. Bir kodu incele derim.

1. Şimdi ilk mesele productlist ten sepete ürün eklenecek. Event
2. productListten eklenen data => navi ye iletilmeli
3. Bu yüzden event ve data akışını yanyana componentlerde sağlayaadıgımız için event ‘ii productList’e props aracılığı ile gönderiyoruz.
4. Producttan Navi ye gönderilecek olan datayı da App.js içinde state olarak tanımlıyoruz.
5. Ve her bir AddToCart event i için şimdilik bu şekilde tanımladık
6. Ve bir de state olarak cart :[] tanımlaması yapıldı.

addToCart = (product) => {

1. let newCart = this.state.cart;
2. var addedItem = newCart.find(i=>i.product.id === product.id)
3. if(addedItem){
4. addedItem.quantity+=1
5. }
6. else{
7. newCart.push({product:product,quantity:1});
8. }
10. this.setState({cart : newCart}) // oluşan bu state i de navi ye göndermemiz gerekiyor.
11. }

## 320. Component Drilling

Sepete eklediğimiz ürünlerin isimlerini göstermek istiyoruz.

 if(addedItem){

      addedItem.quantity+=1 // addedItem daki bu değişiklik state içindeki quantityi nasıl etkiliyor anlamıyorum

    }

## 321. Component Drilling ile event taşıma

### Sepetteki elemanların başına koyduğumuz buton ile eleman silme

Şimdi sepetin içinden eleman silmek istediğimizde işlem cartSmmary’den tetiklenecek ama biz bu datayı zaten app.js te eklemiştik, silme işleminide orda yapalım. Silme işlemini ise taaa App=>navi=>cartSummary’den tetiklemeliyiz.

App.js tanımı

 removeFromCart = (product) =>{

    let filterCart = this.state.cart.filter(c=>c.product.id !== product.id);

    this.setState({ cart : filterCart})

  }

Şimdi event bir altına değil 2 altına nasıl gidecek onu gösterelim.

App.js=>

removeFromCart={this.removeFromCart}

NAvi => CartSumamry

<CardSummary cart={this.props.cart} removeFromCart={this.props.removeFromCart}/>

CartSummary=>

<Badge color="danger" onClick={()=>this.props.removeFromCart(cartItem.product)}>X</Badge>

Artık bu componenet aracılığı ile event taşıma olayı baya oturmuştur.

## 322.Alertify İmplementayonu

Kur

İndex’e ekle

Kullanacağın component’ ekle.

Notify.success(göstermek istediğin not)

## 323. Routing

1. npm install react-router-dom ile routing kurulur

2. index’ e gel

3. import et

import {BrowserRouter} from 'react-router-dom'

4. sarmalla

 <BrowserRouter><App /></BrowserRouter>,

Artık, App imizde routing kullanabiliriz.

5. routing işlemini nerde yapıcaz. App içinde productList yerine isteiğimize göre CartList gelsin istiyoruz. Ozman componenetlerin route’a göre değiştiği yere >switch> <Route> konulur.

Kullanımına örnek için aşağıdaki datayı kullanabilirsin.

<Switch>

                <Route exact path="/" render={props => (

                  <Products

                    selectedCategory={this.state.selectedCategory}

                    products={this.state.products}

                    addToCart={this.addToCart} />

                )} />

                <Route path="/cart" component={CartList}/>

                <Route component={NotFound}/>

              </Switch>

## 324. Sepet detay sayfasının Gösterilmesi