Bu kısımda daha önceki 2 kısımda kullanılan example projesi kullanılmaya devam ediliyor. Product listemiz vesaire var. Bunun üzerinden bir uygulama dönecek.

Her şeyi not etmeyeceğim. Bölüm 7 gerekirse açar izlersin. İzleyip uygulayıp, önemli gördüğüm yerler olursa not edeceğim.

Seçili Ürünün Gösterilmesi

Aşağıdaki kullanımla, view üzerinde selectedProduct bir değere karşılık geliyorsa yazdırılsın gelmiyorsa (None) yazdırılsın dedik.

```
    <div class="bg-info text-white p-2">
      Selected Product: {{selectedProduct || '(None)'}}
    </div>
```

Bu selectedProduct değişkeni componentte tanımlanmadı, sadece yine aynı view içerisinde aşağıdaki gibi event ile tanımlandı, yine de başka bir yerde tanımlanmadan kullanılabildi.

```
<tr *ngFor="let product of products" (mouseover)="selectedProduct=product.name
" (mouseleave)="selectedProduct='(None)'">
```

Burada yapılan şu oldu, tablonun herhangi bir satırına mouse geldiğinde, componentte tanımlanmamış selectedProduct değişkeni içerisine product.name'i at, mouseleave olunca da (None) at. Bu atama işlemi olunca da aynı view üzerinde yukarıda Selected Product gösterilecek.

Satıra Active Class Ekleme 1

Şimdi amacımız, mouse bir satırın üstüne geldiğinde bu satıra bir class eklemek. Bu ekleme işlemini class binding ile yapıyorduk.

Component içerisinde selectedProduct tanımlandı, bunun yanında bir getSelected methodu parametre aldığı product ile view içerisinde atanan selectedProduct aynı ise true döndürür.

```
export class AdminProductsComponent {


  products;
  model: ProductRepository;
  selectedProduct: string;

  constructor() {
    this.model = new ProductRepository();
    this.products = this.model.getProducts();
  }

  getSelected(product: Product): boolean {
    return product.name == this.selectedProduct;
  }

}
```

View içine baktığımızda ise:

```
        <tr *ngFor="let product of products"
        (mouseover)="selectedProduct=product.name"
        (mouseleave)="selectedProduct='(None)'"
        [class.bg-info]="getSelected(product)">
```

İlgili tr elemanına mouseover olduğunda o product'ın selectedProduct olduğunu görüyoruz.

Her tr elemanı için getSelected(product) methodu çalıştırılıyor, eğer ilgili satırdaki ürün o andaki selectedProduct ise bir başka deyişle mouse ilgili ürünün üzerindeyse, tr elemanına bg-info class'ı ekleniyor böylece bir hover efektini class ekleyerek vermiş oluyoruz.

Satıra Active Class Ekleme 2

Bu kısımda geçenki olayı bir adım öteye götüreceğiz, ekrana bir input ekleyeceğiz, inputun içine yazılan değer ile herhangi bir ürünün ismi eşleşirse o ürüne bg-info class'ı eklensin diyeceğiz.

Önceki kısımda yapılan işlem özetle şuydu: her satır getSelected(product) methodu ile o satır içinde yazan eleman ile, o anki selectedProduct'ı karşılaştırıyordu eğer true ise bg-info class'ı ekleniyordu. SelectedProduct da mouseover yapılan eleman olarak atanıyordu.

Şimdi ise yine her eleman getSelected(product) methodu ile o satır içinde yazan eleman ile, o anki selectedProduct'ı karşılaştıracak, ancak bu kez selected product sadece mouseover event'i ile değil, input'a yazılar değer ile belirlenecek.

Input elemanında bir değişiklik olursa selectedProduct input elemanın değeri olarak atanacak böylece input'a Samsung S5 yazarsam , selectedProduct Samsung S5 olacak ve ilgili eleman active class'ını alacak.

Yani değişen şey aşağıdaki kod olacak:

```
      <div class="form-group">
        <label>Product Name</label>
        <input (input)="selectedProduct=$event.target.value" type="text" class
="form">
      </div>
```

Template Değişkenleri

Önceki kısımda aşağıdaki gibi bir input event'i tanımlamıştık, bu event input elemanındaki her değişiklikte çalışan bir event.

```
      <div class="form-group">
        <label>Product Name</label>
        <input (input)="selectedProduct=$event.target.value" type="text" class
="form">
      </div>
```

Her değişiklikte bu event'in çalışması biraz gereksiz, bu yüzden input eventi yerine burda keyup eventi ile örneğin enter tuşuna basılınca, selectedProduct ataması yapılsın diyebiliriz:

```
    <div class="col-md-6">
      <div class="form-group">
        <label>Product Name</label>
        <input #pName (keyup.enter)="selectedProduct=pName.value" type="text"
class="form">
      </div>
    </div>
```

Burada hem event'i değiştirdik hem de input'dan bilgiyi $event yerine id.value ile aldık.

Diğer event'ler içinde de pName.value kullanılabilir:

```
          <tr *ngFor="let product of products"
          (mouseover)="pName.value=product.name"
          (mouseleave)="pName.value='(None)'"
          [class.bg-info]="pName.value==product.name">
```

Yani mouseover veya mouseleave event'leri input'un value'sunu değiştiriyor, bg-info class'ı da input'un value'su ile product name'i kıyaslıyor, true dönerse bg-info class'ı ekleniyor.

Ürün Detaylarının Gösterilmesi

Edit butonuna basılınca ilgili ürünün bilgileri form içerisinde gösterilsin:

```
    <div class="col-md-6">

      <div class="form-group">
        <label>Product Name</label>
        <br>
        <input #pName [value]="selectedProduct.name" type="text" class="form">
      </div>

      <div class="form-group">
        <label>Product Price</label>
        <br>
        <input #pPrice [value]="selectedProduct.price" type="text" class="form
">
      </div>

      <div class="form-group">
        <label>Product Url</label>
        <br>
        <input #pImageUrl [value]="selectedProduct.imageUrl" type="text" class
="form">
      </div>

      <div class="form-group">
        <label>Product Description</label>
        <br>
        <input #pDescription [value]="selectedProduct.description" type="text"
 class="form">
      </div>

    </div>
```

Görüldüğü gibi selectedProduct'ın bilgileri property binding ile input formlarına doldurulуyor.

Peki selected product nasıl belirleniyor? Hangi product için edit butonuna basılırsa o product selected product olarak atanıyor.

```
          <tr *ngFor="let product of products"
          [class.bg-info]="getSelected(product)">
            <td>{{product.id}}</td>
            <td><img src = "/assets/img/{{product.imageUrl}}" width="80"></td>
            <td>{{product.name}}</td>
            <td>{{product.price}}</td>
            <td>
              <button class="btn btn-danger btn-sm">Delete</button>
              <button (click)="editProduct(product)"  class="btn btn-
primary btn-sm ml-2">Edit</button>
            </td>
          </tr>
```

Yukarıda görüldüğü gibi eğer edit butonuna basılırsa editProduct() methodu yardımıyla ilgili edit butonu hangi prudct'a ait ise o product selectedProduct olarak atanır.

Ayrıca eğer ilgili product selected product ise bg-info class'ına sahip olur, yani edit butonuna basılan product selected olur ve bg-info class'ını alır.

```
  getSelected(product: Product): boolean {
    return product == this.selectedProduct;
  }

  editProduct(product: Product){
    this.selectedProduct = product;
  }
```

Değişikliklerin Kayıt Edilmesi

Önceki kısımda edit butonuna basınca ürün bilgilerini bir form içerisinde getirdik.

Şimdi ise bu form içeriklerini değiştirdikten sonra yeni ürünü nasıl kaydedebiliriz ona bakacağız.

Form'un altına bir save butonu ekleriz, ve save butonuna tıklandığında form inputları saveProduct() methoduna aktarılır.

```
<button (click)="saveProduct(pName.value,pDescription.value,pImageUrl.value,pP
rice.value)"    class="btn btn-primary btn-sm ml-2">Save</button>
```

Component içindeki saveProduct methoduna bakarsak:

```
  saveProduct(name,description,imageUrl,price){
    this.product = this.model.getProductsById(this.selectedProduct.id);
    this.product.name = name;
    this.product.description = description;
    this.product.imageUrl = imageUrl;
    this.product.price = price;
    this.selectedProduct = null;
  }
```

Burada da yapılan selectedProduct'ı yani input form'da bilgileri gösterilen product'ı model içerisinden bulmak ve bu product'ın parametrelerini yukarıdaki gibi update etmek.

Değişikliklerin Two Way Binding ile Kayıt Edilmesi

Önceki kısımda form'dan alınan bilgileri save butonu ile componente taşımıştık, bunu yapmak yerine input form içeriklerini ngModel ile component içeriğine bağlarsak anlık olarak update işlemi sağlayabiliriz:

Artık form elemanları aşağıdaki gibi çift yönlü olarak selectedProduct'ın özellikleri ile bağlandı.

```
<div class="col-md-6" *ngIf="selectedProduct">

      <div class="form-group">
        <label>Product Name</label>
        <br>
        <input [(ngModel)] = "selectedProduct.name" type="text" class="form">
      </div>
```

Uygulama ilk yüklendiğinde selectedProduct olmadığı için form da gösterilmeyecektir.

Edit'e basılınca selectedProduct belirlenecek, forma bilgiler düşecektir, düşen bilgiler değiştirilirse anlık olarak selectedProduct'ın içeriği de değişecektir, zaten bu selectedProduct da referans ile atandığı için orijinal product da değişmiş olacaktır.

Bana kalırsa bu savaeProduct methoduna gerek yok ama yine de koyduk.

```
<button (click)="saveProduct()"   class="btn btn-primary btn-sm ml-
2">Save</button>
```

```
  saveProduct(){
    this.product = this.model.getProductsById(this.selectedProduct.id);
    this.product = this.selectedProduct;
    this.selectedProduct = null;
}
```

Eğer ngIf'ı koyamasaydık problem çıkardı neden?

- selectedProduct ilk başta olmadığı için problem çıkıyor.