**Bir CSS Çerçevesinin Araştırması ve Dokümantasyonu**

**Bootstrap Teknolojisi**

İstanbul,Türkiye

Salih Enes ÖZDEL Ön Uç Geliştirici

**Özet**

Teknolojinin gelişmesiyle beraber yazılım sektörü de gelişmiştir. Ön uç web geliştirme alanı da yazılım dünyasında son zamanların popüler alanı haline gelmiştir.Farklı eklentiler, geliştirmeler, kütüphaneler, frameworkler bu sebeple günümüzde hızla web tasarımcılara ulaşıyor ve kullanıma sunuluyor. Teknolojiden faydalanan Bootstrap de yenilenen dünyada yerini almıştır. Bootstrap açık kaynaklı bir ön uç geliştirme framework’ü olup web uygulamaları oluşturmak için kullanılır. Web tasarımda özellikle responsive yani mobil uyumlu siteler, uygulamalar yapılırken kullanılan Bootstrap, oluşturulan iskelet yapıların (HTML etiketlerinin) tasarımında kendine has 'class' yapısıyla CSS kullanımında karşılaşılan zorluğu en aza indirmeyi planlamaktadır. Bu çalışmada Bootstrap teknolojisinin içeriğinden ve ön-uç geliştirmecilere sağladığı yararlardan bahsedeceğiz.

**Giriş**

Gündelik yaşamımızda artık hiçbirimizin görmezden gelemeyeceği bir olgu var: Teknoloji. Teknoloji hayatımıza girdiğinden beri her alanda kendimizi onun getirdiği yeniliklere kaptırıyoruz. İnsanlar olarak hayatımızı kolaylaştıracak makineler, robotlar yapıyoruz ve bunun sonucunda sanayi alanında gün be gün ilerliyoruz. Ürettiğimiz makinaları yönetiyoruz. Böylelikle sanayi sektörü teknolojiyle ilerliyor ve yazılım gelişiyor. Yazılım alanında son zamanların popüler alanı olan ön uç geliştirme teknolojiden oldukça faydalanıyor. Böylece bu alan için farklı eklentiler, geliştirmeler, kütüphaneler, frameworkler geliştiriliyor ve kullanıcıların kullanımına sunuluyor. Bu alanda geliştirilen Bootstrap framework’ü, geliştirmecilere tasarım alanında oldukça kolaylık sağlıyor. Bootstrap kendine has 'class' yapısıyla responsive yani mobil uyumlu siteler, uygulamalar yapmaya yarayan bir teknoloji olarak karşımıza çıkıyor. ''class" yapıları herhangi bir CSS dosyası kullanmadan HTML etiketlerinin içine gömüldüğü için aynı zamanda geliştirmeciyi bir külfetten kurtarıyor diyebiliriz. Bootstrap, şablon tasarımları için bir sözdizimi koleksiyonu sağlayarak, geliştirmecinin farklı modeller oluşturmasına yardım eden bir teknolojiye sahiptir. Bu çalışma, Bootstrap teknolojisinin içeriğinden ve nasıl kullanılacağı hakkında bilgi vermek için hazırlanmıştır. Diğer bir deyişle bu çalışma Bootstrap teknolojisi üzerine yapılan araştırma sonrası dokümantasyon oluşturmak amacını taşıyarak yazılmıştır.

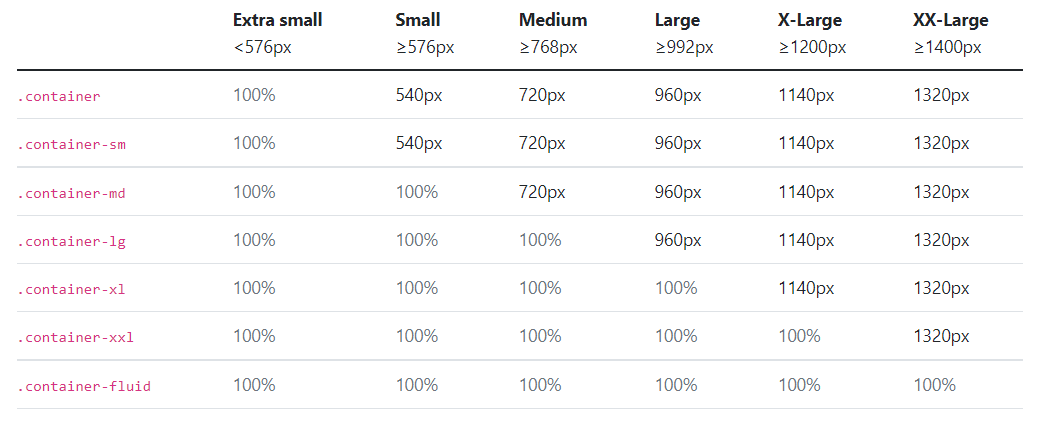
**Materyal ve Metot**

**Class Yapısı**

Class yapısı HTML' de kullanılan etiketleri aslında kümelemeye yarar diyebiliriz. Bootstrap de HTML' deki bu "class" yapısından faydalanarak kullanıldığı HTML etiketine etki eder. Class yapısında temel mantık aslında aynıdır. Örneğin boyut değiştirme, renk değiştirme, bazı animasyon hareketleri gerçekleştirmek için kullanılırlar. Aslında "class" yapılarını birer fonksiyon olarak düşünebiriz. Böylelikle kullanılan "class" lar değişirse gerçekleşen fonksiyon da değişmiş olacaktır.

**Container Yapısı**

Bootstrap responsive siteler tasarlamak için geliştirilmiş bir framework’tür. Responsive sitelerde en önemli noktalardan biri, sayfanın boyutu ile yer değiştiren elemanlardır diyebiliriz. Bu sebeple Bootstrap belirli sayfa genişliğine göre class’lar geliştirmiştir.



**Figür 1: Container Class Yapısı / Ekran Genişliği**

Yukarıda gösterilen figürde sol tarafta kullanılan class’a karşılık gelen sayfa genişlikleri figürün sağ tarafında detaylıca gösterilmiştir. Örnek vermek gerekirse container-sm class’ı ile ekran genişliği 576px’den küçük sayfalarda ekran ölçüsü 540px’e ayarlanmıştır. Diğer class’lar için de aynı şekilde düşünebiliriz. Figürün en alt kısmında bulunan container-fluid class’ı ise kullanılan sayfanın genişliği ne olursa olsun sayfayı tamamen kaplamaktadır.

**Grid Yapısı**

Bootstrap’in en önemli özelliklerinden ve bilinmesi gereken yapı Grid yapısıdır. Bootstrap teknolojisinin <div> etiketlerine class tanımlayarak çalıştığından daha önce bahsetmiştik. Grid yapısında ise tanımlanan bu <container> etiketlerinin içinde row ve column, yani satır ve sütun yapısı bulunmaktadır.”12” sütun sistemi olarak adlandırabileceğimiz Grid sisteminde bir satırda(row) maksimum 12 sütun olabilecek şekilde tasarlanmıştır. Ancak bu sayı oluşturulan sütunların (column) genişliğine göre değişebilir. Bu işlemi de sütunlara tanımladığımız class’lar sayesinde yapıyoruz.

<div class="container ">

<div class="row">

<div class="col">Column</div>

</div>

</div>

Örnekte görüldüğü gibi container div’inin içine önce bir row div’i, daha sonra bu row div’inin içine bir col div’i atanmıştır. Herhangi bir class col div’ine tanımlanmadığı için bu col içerisinde bulunduğu row ve container yapısının genişliğinin 12’de 12’sini alacaktır. Container’ın genişliğini bu sayfa genişliği olarak kabul edersek aşağıdaki gibi bir sonuç elde edeceğiz.

COLUMN

**Figür 2**

Yukarıda bahsedilen 12’li sütun sisteminin daha kolay anlaşılması için aşağıda verilen figürü inceleyiniz.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column1 | Column2 | Column3 | Column4 | Column5 | Column6 |

**Figür 3**

Bu figürde aynı row içinde “col-2” class’ı kullanan 6 adet column görmekteyiz.

Örneğin aynı row içinde 3 adet column oluşturmak istiyoruz. Ancak bu columnlardan birinin genişliği sayfanın yarısı yani 6/12 si , diğeri 4/12 si , sonuncusu ise 2/12 si şeklinde ayarlanmak istenirse şu şekilde bir kod yazılması gerekir:

<div class="container ">

<div class="row">

<div class="col-6">6/12 COLUMN</div>

<div class="col-4">4/12 COLUMN</div>

<div class="col-2">2/12 COLUMN</div>

</div>

</div>

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
| 6/12 COLUMN | | | | | | 4/12 COLUMN | | | | 2/12 COLUMN | |

**Figür 4**

Grid sisteminde iç içe rowlar oluşturmak da mümkündür. Bu sayede columnlar istenilen şekilde konumlandırılabilir.

**Columnların Hizalanması**

Row’un içinde oluşturulan columnlar, kullanıcının istediği şekilde hizalanmaktadır. Dikey ve yatay olarak hizalanabilen bu columnlar için farklı class yapıları kullanılır. Bu hizalama işlemi için kullanılan bazı class yapılarını inceleyelim:

**Dikey Hizalama**

* Aynı row içerisindeki columnları hizalama işlemi için columnların bulunduğu row’a class tanımlarız.

class="row align-items-start"

Tanımlanan class ile row içerisindeki columnlar row’a bağlı olarak, içerisi container’ın en üst kısmında hizalanır.

class="row align-items-center"

Tanımlanan class ile row içerisindeki columnlar row’a bağlı olarak, içerisi container’ın orta(merkez) kısmında hizalanır.

class="row align-items-end"

Tanımlanan class ile row içerisindeki columnlar row’a bağlı olarak, içerisi container’ın en alt kısmında hizalanır.

* Dikeyde sütunları hizalamak için kullanılan bir diğer yöntem ise row’a değil içinde buunan columnlara class tanımlamaktır.

class="col align-self-start"

Tanımlanan class ile row içerisindeki columnlar başlangıç konumuna hizalanırlar.

class="col align-self-center"

Tanımlanan class ile row içerisindeki columnlar merkez konumuna hizalanırlar.

class="col align-self-end"

Tanımlanan class ile row içerisindeki columnlar bitiş konumuna hizalanırlar.

**Yatay Hizalama**

* Sütunları yatayda birleşik yani içerisinde bulunduğu row’la beraber hizalamak istiyorsak, row’a class tanımlarız.

class="row justify-content-start"

Aynı row içerisindeki sütunları başlangıç konumuna (en sola) hizalar.

class="row justify-content-center"

Aynı row içerisindeki sütunları merkez konumuna (ortaya) hizalar.

class="row justify-content-end"

Aynı row içerisindeki sütunları bitiş konumuna (en sağa) hizalar.

* Sütunları beraber değil de, ayrı ayrı hizalamak istiyorsak oluşturduğumuz columnlara class tanımlarız.

class="row justify-content-between"

Row içerisindeki columnlar aralarında eşit boşluk kalacak şekilde başlangıç ve bitiş konumlarına hizalanırlar.

* Bir diğer hizalama yöntemi ise offset class’ı kullanarak yatayda kaydırma işlemi yapmaktır. Bu class ile oluşturulan sütun istenilen sütun sayısı kadar sol boşluk arttırılarak sağa kaydırılabilir.

class="col-md-4 offset-md-4"

Tanımlanan class ile oluşturulan sütun 4 column kadar sağa kaydırılır.

* Sütunların margin özelliğini değiştirerek kardeş sütunların birbiri arasındaki mesafeyi değiştirmek ve böylece hizalama yapmak da diğer bir seçenek olarak karşımıza çıkıyor.

**z-index**

z-indexler bileşenlerin birbiri arasında etkileşimini sağlamak için kullanılan önemli parçalardır. z-index aslında bir sıralama ölçütüdür diyebiliriz. Oluşturulan bileşenlerin hangi sırayla çalışacağını bize gösteren index numaralarıdır.

**Form Yapıları**

Bootstrap ile HTML içerisine kolayca gömebildiğimiz form yapıları mevcuttur. Form yapıları genelde bir bilginin doğruluğunu araştırmak için kullanılır. Bu yapılar sayesinde kullanıcıdan alınan bilgiler değerlendirilir. Bootstrap ile hazır gelen form yapıları sayesinde kişiselleştirebileceğimiz formlar üretebiliriz.

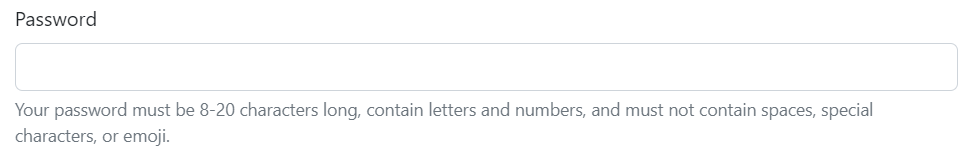
<label for="inputPassword5" class="form-label">Password</label>

<input type="password" id="inputPassword5" class="form-control" aria-describedby="passwordHelpBlock">

<div id="passwordHelpBlock" class="form-text">

Your password must be 8-20 characters long, contain letters and numbers, and must not contain spaces, special characters, or emoji.

</div>



**Figür 5**

Bu örnekte kodları tek tek inceleyelim ve bazı anahtar class’ları vurgulayalım.

Kod satırında en üstte bulunan kod, oluşturulan form yapısı için başlık görevi görüyor diyebiliriz. Burada “form-label” class’ı ile formu “Password” şeklinde etiketlemek istenmiştir.

İkinci satırda kullanıcının bilgiyi paylaşacağı input etiketi verilmiş. Burada ise “form-control” class’ı kullanılarak oluşturulan form yapısının bir kontrol mekanizması olduğu belirtilmiştir. Aynı zamanda “aria-describedby” kullanılarak ilişkilendirileceği yapıya gönderme yapılmıştır ki burada “ aria-decribedby=’passwordhelpBlock’ ” tur. Yani ilişkilendirileceği yapı ‘passwordhelpBlock’ tur diyebiliriz. Üçüncü satırda “form-text” class’a sahip id’si ‘passwordhelpBlock’ olan div’i gördüğümüzde anlıyoruz ki işte oluşturulan bu text yapısı bir üst kod satırındaki input ile ilişkilendirilmiştir.

Basit bir form-control yapısı oluşturmak için bir adet <label> etiketi, bir adet <input> etiketi kullanmamız yeterlidir. <label> etiketi içerisinde “form-label” class’ı ile oluşturulan form yapısının neyi işaterlediğini anlarız. Oluşturduğumuz <input> etiketi içerisine “form-control” class’ı ekleyerek kontrol edilecek yapının <input> etiketi içerisinde olduğunu anlarız. Oluşturulan yazı boşluklarına (input, form-text, textarea vb.) ‘placeholder’ özelliği tanımlayarak kullanıcı etkileşiminden önce istenilen bilgi default olarak gösterilebilir.

Kullanıcının etkileşime girmesini istemediğimiz durumlarda “disabled” (elverişsiz) özelliği form yapılarında kullanılabilir.

<input class="form-control" type="text" placeholder="Disabled input" aria-label="Disabled input example" disabled>



**Figür 6**

Kontrol formları oluştururken sıklıkla kullandığımız diğer bir yapı “button” lardır. Kontrol edilmesi istenen <input> etiketinin içerisindeki bilgi genellikle butonlar aracılığı ile aktarılır.

<button type="submit" class="btn btn-primary mb-3">Confirm identity</button>



**Figür 7**

Bir diğer form yapısı da dosya yükleme (upload) için kullanılan yapıdır. Bu yapıyı oluştururken <input> etiketinin içerisinde “type” özelliğini ‘file’ olarak ayarlamak gerekir. (type = “file“)

<div class="mb-3">

<label for="formFile" class="form-label">Default file input example</label>

<input class="form-control" type="file" id="formFile">

</div>

Örnek incelendğinde <label> etiketi ile formFile kimliğine sahip element işaretlenmiştir. Bir alt kod satırında ise <input> etiketi ile dosya tipinde form-control yapısı oluşturulmuştur.

Bootstrap ile form yapıları oluşturuken veri listesi opsiyonu sayesinde seçim yapmak çok kolay. “datalist” ile önceden belirlenen veriler <input> etiketi içerisine kolayca yazılır ve kullanıcının seçim yapmasını kolaylaştırır.

<label for="exampleDataList" class="form-label">Datalist example</label>

<input class="form-control" list="datalistOptions" id="exampleDataList" placeholder="Type to search...">

<datalist id="datalistOptions">

<option value="İstanbul">

<option value="Ankara">

<option value="İzmir">

<option value="Eskişehir">

</datalist>

Yukarıdaki örneği incelediğimizde <input> etiketi içerisine list özelliği ile “datalistOption” eklenmiştir. Daha sonra <datalist> etiketi kullanarak datalistOptions kimliği liste yapısı oluşturularak (her bir seçim için <option> etiketi ve her bir <option> etiketi için değer) tamamlanmıştır. Bu sayede oluşturulan form-yapısında kullanıcı datalist içerisinden herhangi bir veriyi kolayca seçebilecektir.

Seçim menüleri oluşturmak için <select> etiketi içerisine kolayca form yapısı oluşturabiliriz.

<select class="form-select" aria-label="Default select example">

<option selected>Open this select menu</option>

<option value="1">One</option>

<option value="2">Two</option>

<option value="3">Three</option>

</select>

Seçim menüleri oluşturmak datalist form yapıları oluşturmaya çok benzerdir. Ancak seçim menüleri oluştururken <input> etiketi yerine <select> etiketti kullanılır. İkinci kod satırında yazan <option selected> default olarak ekranda seçilmiş olan opsiyonu belirtir.

Form yapılarında en çok kullandığımız donanımlardan “checkbox” ve “radio” seçenekler listesinden seçenek seçmemize yarar. “checkbox” ile birkaç seçenek seçebilirken “radio” çok seçenek arasından sadece bir seçenek seçmeye olanak sağlar.

<div class="form-check">

<input class="form-check-input" type="checkbox" value="" id="flexCheckDefault">

<label class="form-check-label" for="flexCheckDefault">

Default checkbox

</label>

</div>



**Figür 8**

Örnekte görüldüğü üzere basit bir “check-box” form yapısı oluşturulmuştur. <input> etiketi içerisine “checkbox” tip olarak belirlenmiş ve <label> etiketi ile ‘flexCheckDefault’ kimliği için “Default checkbox” ismi ile işaretlenme yapılmıştır.

<div class="form-check">

<input class="form-check-input" type="radio" name="flexRadioDefault" id="flexRadioDefault1">

<label class="form-check-label" for="flexRadioDefault1">

Default radio

</label>

</div>

<div class="form-check">

<input class="form-check-input" type="radio" name="flexRadioDefault" id="flexRadioDefault2" checked>

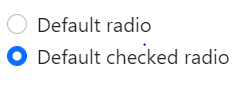
<label class="form-check-label" for="flexRadioDefault2">

Default checked radio

</label>

</div>

Yukarıda “radios” oluşturma örneğine yer verilmiştir. Oluşturulan <input> etiketleri içerisine tip olarak “radio” belirlenmiş <label> etiketleri ile de sırasıyla “flexRadioDefault1”; “flexRadioDefault2” kimlikleri için sırasıyla “Default radio”; “Default checked radio” isimleri ile işaretleme yapılmıştır.



**Figür 9**

İki örneği karşılaştırdığımızda görüyoruz ki: “checkbox” ile birden çok seçenek seçebilirken “radio” ile sadece bir seçenek seçebiliriz.

“checkbox” oluştururken kaydırma – değiştirme rolünde form yapıları ortaya koyabiliriz. “Switches” olarak adlandırılan bu yapılar “form-switch” class’ına sahip div’ler içerisinde oluşturulur. “checkbox” oluştururken kullandığımız <input> etiketine rol olarak “switch” tanımlarız.

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" role="switch" id="flexSwitchCheckChecked" checked>

<label class="form-check-label" for="flexSwitchCheckChecked">Checked switch checkbox input</label>

</div>



**Figür 10**

Örneği incelediğimizde, en üst kod satırında form-switch class yapısını görmekteyiz. İkinci kod satırında <input> etiketi içerisinde “checkbox” tip; “switch” rol olarak belirlenmiştir. <label> etiketi ile “flexSwitchCheckChecked” kimliği için “Checked switch checkbox input” ismi ile işaretleme yapılmıştır.

form-check-inline class’ı ile oluşturulan checkboxları veya radioları yan yana aynı satırda yazabiliriz.

<div class="form-check form-check-inline">

<input class="form-check-input" type="checkbox" id="inlineCheckbox1" value="option1">

<label class="form-check-label" for="inlineCheckbox1">1</label>

</div>

<div class="form-check form-check-inline">

<input class="form-check-input" type="checkbox" id="inlineCheckbox2" value="option2">

<label class="form-check-label" for="inlineCheckbox2">2</label>

</div>



**Figür 11**

Bu örnekte birinci kod satırındaki form-check-inline class’ı sayesinde iki adet checkbox (1, 2) yan yana sıralanmıştır.

Formlarda aralık girdileri kullanarak stil düzenlemesi yapmak mümkün. <input type=”range” class=”form-range”> girdisi ile arka plan çubuğu ve onun üzerinde değer çubuğu bulunan bir form aralığı oluşturulur.

<label for="customRange3" class="form-label">Example range</label>

<input type="range" class="form-range" min="0" max="5" step="0.5" id="customRange3">



**Figür 12**

Yukarıda verilen örnek incelenirse birinci kod satırında <label> etiketi ile “customRange3” kimliği için işaretleme işlemi yapılmıştır. İkinci kod satırında ise aralık girdisi <input> etiketi kullanılarak oluşturulmuştur. Görüldüğü gibi type=”range”, class=”form-range” ile temel düzeyde aralık oluşturma adımı tamamlanmış olup ‘min’, ‘max’, ‘step’ özellikleri ile oluşturulan aralık geliştirilmiştir. Bu örnekte 0’dan başlanıp 0.5 artış hızlı 5’te bitirilen bir aralık tanımlanmıştır.

Girdi grupları oluşturarak form yapılarının başına veya sonuna metin ve/veya buton grupları eklenebilir. Bu yapıyı oluşturmak için formun kapsandığı div yapısının “input-group” class’ına sahip olması gerekir ve eklenilmesi istenen metin vb. <span> etiketi içerisine yazılır. <input> etiketinden önce <span> kullanılırsa eklenilen metin başta, sonra <span> kullanılırsa eklenilen metin satırın sonunda olur.

<div class="input-group mb-3">

<span class="input-group-text">$</span>

<input type="text" class="form-control" aria-label="Amount (to the nearest dollar)">

<span class="input-group-text">.00</span>

</div>



**Figür 13**

Örnekte iki adet <span> etiketi kullanılmıştır. Birisi <input> etiketinden önce olduğu için satır başında gözüken “$” işareti iken, diğeri <input> etiketinden sonra kullanılarak satır sonunda gözüken “.00” dur. İlk kod satırında “input-group” class’ı kullanılmıştır.

Girdi gruplarının başında veya sonunda kullanılan diğer yapılar: checkbox, buton, aşağı açılır buton (dropdown buton) örnekleri aşağıda verilmiştir.

1. Checkbox

<div class="input-group mb-3">

<div class="input-group-text">

<input class="form-check-input mt-0" type="checkbox" value="" aria-label="Checkbox for following text input">

</div>

<input type="text" class="form-control" aria-label="Text input with checkbox">

</div>



**Figür 14**

1. Buton

<div class="input-group mb-3">

<input type="text" class="form-control" placeholder="Recipient's username" aria-label="Recipient's username" aria-describedby="button-addon2">

<button class="btn btn-outline-secondary" type="button" id="button-addon2">Button</button>

</div>



**Figür 15**

1. Dropdown Buton

<div class="input-group mb-3">

<button class="btn btn-outline-secondary dropdown-toggle" type="button" data-bs-toggle="dropdown" aria-expanded="false">Dropdown</button>

<ul class="dropdown-menu">

<li><a class="dropdown-item" href="#">Action</a></li>

<li><a class="dropdown-item" href="#">Another action</a></li>

<li><a class="dropdown-item" href="#">Something else here</a></li>

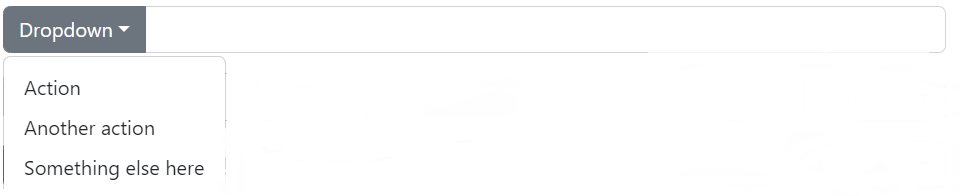
<li><hr class="dropdown-divider"></li>

<li><a class="dropdown-item" href="#">Separated link</a></li>

</ul>

<input type="text" class="form-control" aria-label="Text input with dropdown button">

</div>



**Figür 16**

<div class="input-group">

<select class="form-select" id="inputGroupSelect04" aria-label="Example select with button addon">

<option selected>Choose...</option>

<option value="1">One</option>

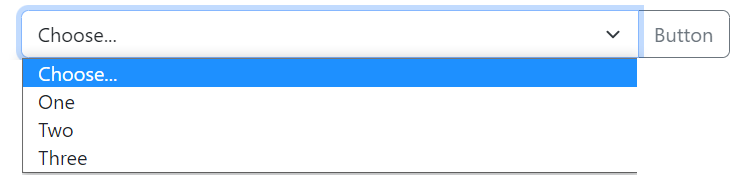
<option value="2">Two</option>

<option value="3">Three</option>

</select>

<button class="btn btn-outline-secondary" type="button">Button</button>

</div>



**Figür 17**

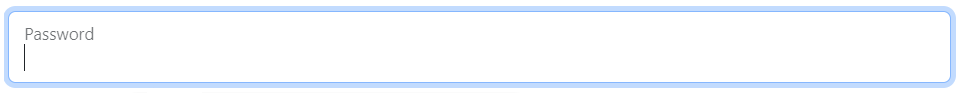
Oluşturulan form yapılarında kayan işaretçilerle özelleştirme gerçekleştirilebilir. Formu kapsayan div’in “form-floating” class’ına sahip olması, <input> etiketi içerisinde placeholder özelliğinin tanımlanması ve <label> etiketinin <input> etiketinden sonra yazılması gerekmektedir.

<div class="form-floating">

<input type="password" class="form-control" id="floatingPassword" placeholder="Password">

<label for="floatingPassword">Password</label>

</div>



**Figür 18-19**

Örnekte “floatingPassword” kimliği için “Password” ismiyle işaretçi tanımlanmıştır. Bu işaretçi form üzerine tıklandığında resimde görüldüğü gibi yukarı doğru kaymaktadır.

Form yapısında kullanılan kayan işaretçiler farklı etiketler altında bulunabilir. (<textarea>, <select>, <span>, vb.)

**Validation**

Form doğrulaması ile kullanıcılara eyleme geçebilen geri bildirim sağlanabilir. Form doğrulaması <input>, <select> ve <textarea> etiketleri için geçerlidir. <form> etiketi içerisine uygulanan .was-validated sınıfına :invalid ve :valid stilleri eklenir. Bootstrap form doğrulama mesajları için "novalidate" niteliğini <form> etketi içerisine eklemeniz gerekir. Böylelikle tarayıcı geri bildirim araç ipuçlarını devre dışı bırakır, ancak yine de JavaScript'teki form doğrulama API'lerine erişim sağlanır.

<form class="row g-3 needs-validation" novalidate>

<div class="col-md-4">

<label for="validationCustom01" class="form-label">First name</label>

<input type="text" class="form-control" id="validationCustom01" value="Mark" required>

<div class="valid-feedback">

Looks good!

</div>

</div>

<div class="col-md-4">

<label for="validationCustom02" class="form-label">Last name</label>

<input type="text" class="form-control" id="validationCustom02" value="Otto" required>

<div class="valid-feedback">

Looks good!

</div>

</div>

<div class="col-md-4">

<label for="validationCustomUsername" class="form-label">Username</label>

<div class="input-group has-validation">

<span class="input-group-text" id="inputGroupPrepend">@</span>

<input type="text" class="form-control" id="validationCustomUsername" aria-describedby="inputGroupPrepend" required>

<div class="invalid-feedback">

Please choose a username.

</div>

</div>

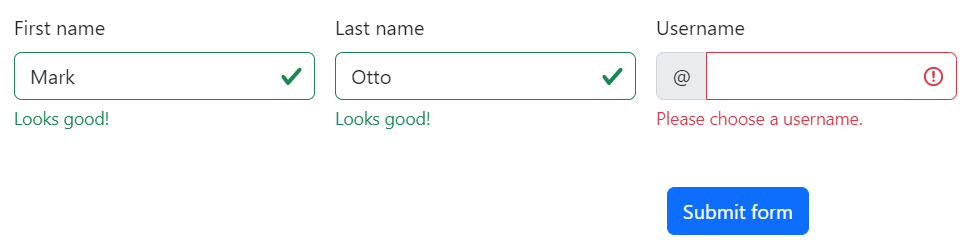
</div>

<div class="col-12">

<button class="btn btn-primary" type="submit">Submit form</button>

</div>

</form>

**Figür 20**

Örnekte görüldüğü üzere “novalidate” niteliği <form> etiketi içerisine tanımlanmıştır. Form yapısının içerisindeki sütun oluşturan <div> ler bireysel olarak incelendiğinde ilk iki sütun geri bildirim ekranında “valid-feedback” sınıfı tanımlanışken sonuncuda “invalid-feedback” sınıfı tanımlanmış ve görseldeki gibi bir sonuç ortaya çıkmıştır.

Tarayıcı varsayılanı olarak JS yazmakla ilgilenmeyen kullanıcılar için "novalidate" niteliği tanımlanmak zorunda değildir. Daha basit bir kullanıcı etkileşimi içeren tarayıcı varsayılan doğrulama formu aşağıda örneklendirilmiştir.

<form class="row g-3">

<div class="col-md-4">

<label for="validationDefault01" class="form-label">First name</label>

<input type="text" class="form-control" id="validationDefault01" value="Mark" required>

</div>

<div class="col-md-4">

<label for="validationDefault02" class="form-label">Last name</label>

<input type="text" class="form-control" id="validationDefault02" value="Otto" required>

</div>

<div class="col-md-4">

<label for="validationDefaultUsername" class="form-label">Username</label>

<div class="input-group">

<span class="input-group-text" id="inputGroupPrepend2">@</span>

<input type="text" class="form-control" id="validationDefaultUsername" aria-describedby="inputGroupPrepend2" required>

</div>

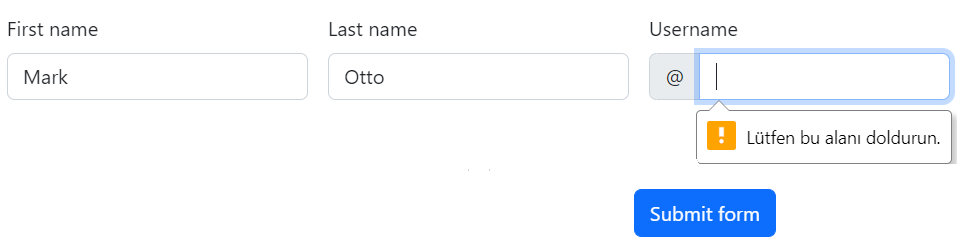
</div>

<div class="col-12">

<button class="btn btn-primary" type="submit">Submit form</button>

</div>

</form>

 **Figür 21**

Örneği incelediğimizde “novalidate” niteliğinin olmadığını görmekteyiz. Ayrıca “valid” veya “invalid” olarak bir geri bildirim de almamaktayız. Bunun yerine “required” class’ının <input> etiketinin içerisine tanımlanması sayesinde sadece gerekli alanların doldurulması gerektiğine dair bir mesaj almaktayız.

**Bileşenler - Komponentler**

**Accordion - Akordeon**

Dikey olarak katlanabilen akerdoanlar oluşturmak Bootsrap teknolojisi ile çok kolay. Akerdeon oluşturmak için "collapse" class'ını kapsayıcı <div> e tanımlamak yeterlidir.

<div class="accordion" id="accordionExample">

<div class="accordion-item">

<h2 class="accordion-header" id="headingOne">

<button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-target="#collapseOne" aria-expanded="true" aria-controls="collapseOne">

Accordion Item #1

</button>

</h2>

<div id="collapseOne" class="accordion-collapse collapse " aria-labelledby="headingOne" data-bs-parent="#accordionExample">

<div class="accordion-body">

<strong>1.</strong>

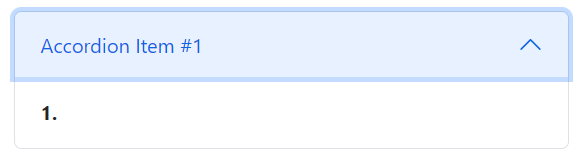
</div>

</div>

</div>

</div>



**Figür 22**

Örnekte görüldüğü gibi “collapse” sınıfı akordeon oluşturmak için kullanılmıştır. Ayrıca akordeonun sürekli açık kalmasını istersek örnekteki gibi “accordion-collapse” sınıfı da tanımlanabilir.

**Alerts - Uyarılar**

Kullanıcı eylemleri için uyarı mesajı sağlanması "alert" ile geri bildirim şeklinde olur. İsteğe bağlı bir kapatma butonu bulundurur.

<div class="alert alert-primary" role="alert">

A simple primary alert—check it out!

</div>



**Figür 23**

Örnekte görüldüğü gibi basit bir şekilde uyarı mesajı oluşturmak mümkündür. Burada uyarı almak için “alert” class’ı tanımlanmıştır. “alert-primary” uyarının rengini belirler.

Uyarılarda ayrıca farklı etiketler (<p> <h2> ) veya ikonlar da kullanılabilir.

<div class="alert alert-primary d-flex align-items-center" role="alert">

<svg xmlns="http://www.w3.org/2000/svg" class="bi bi-exclamation-triangle-fill flex-shrink-0 me-2" viewBox="0 0 16 16" role="img" aria-label="Warning:">

<path d="M8.982 1.566a1.13 1.13 0 0 0-1.96 0L.165 13.233c-.457.778.091 1.767.98 1.767h13.713c.889 0 1.438-.99.98-1.767L8.982 1.566zM8 5c.535 0 .954.462.9.995l-.35 3.507a.552.552 0 0 1-1.1 0L7.1 5.995A.905.905 0 0 1 8 5zm.002 6a1 1 0 1 1 0 2 1 1 0 0 1 0-2z"/>

</svg>

<h6 class="alert-heading">Well done!</h6>

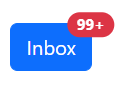
</div>



**Figür 24**

**Badge – Nişan**

Bileşenleri etiketlemek için kullanılan nişan yapısı farklı etiketler içerisinde kullanılabilir. Farklı boyutlarda , farklı pozisyonlarda ve farklı renklerde nişanlar kullanılabilir.



**Figür 25**

<button type="button" class="btn btn-primary position-relative">

Inbox

<span class="position-absolute top-0 start-100 translate-middle badge rounded-pill bg-danger"> 99+

<span class="visually-hidden">unread messages</span>

</span>

</button>

Bu örnekte buton yapısı içerisinde kullanılan “badge” class’ı ile pozisyon, renk ve kenar şekli belirlenmiştir.

**Breadcrumb**

Geçerli sayfanın konumunu hiyerarşik olarak belirten yapılara "breadcrumb" denir.

<nav aria-label="breadcrumb">

<ol class="breadcrumb">

<li class="breadcrumb-item"><a href="#">Home</a></li>

<li class="breadcrumb-item active" aria-current="page">Library</li>

</ol>

</nav>



**Figür 26**

Bu örnekte breadcrumb sınıfı kullanılarak kullanıcının bulunduğu dizinin “Library” sayfası olduğu belirtilmek istenmiştir. <a> etiketi ile link verilmiş olup bulunulan sayfanın “Home” sayfasının içerisinde olduğunu görmek mümkündür.

**Buttons - Butonlar**

Formlarda aksiyonu kontrol etmek için kullanılan yapılardır. Farklı boyutlarda ve farklı renklerde kullanılabilirler.

<button type="button" class="btn btn-primary">Primary</button>

<button type="button" class="btn btn-secondary">Secondary</button>



**Figür 27**

Butonlar “btn” class’ları için geliştirilmiş olsa da <input>, <a> etiketleri içerisinde de kullanılabilirler. Ancak <a> etiketi içerisinde kullanım için <a> etiketi içerine “role=button” tanımlanmalıdır. <input> etiketi ile oluşturulmuş butonlar farklı amaçlar için kullanıldığından burada “type= “ niteliği önem taşır. Örneğin “type=submit” olarak ayarlanan bir <input> artık bir “submit” butonudur.

<a class="btn btn-primary" href="#" role="button">Link</a> <input class="btn btn-primary" type="submit" value="Submit">



**Figür 28**

Sadece kenarlık gözükecek şekilde butonlar oluşturulabilir. Üzerlerine gelindikleri zaman ise arka plan rengi alırlar.

<button type="button" class="btn btn-outline-primary">Primary</button>

<button type="button" class="btn btn-outline-secondary">Secondary</button>



**Figür 29**

Buton grupları oluşturmak istiyorsak kapsayıcı div’in “btn-group” class’ına sahip olması gerekir. Yardımcı teknolojilerin butonların gruplandığını iletebilmesi için "role" niteliğinin sağlanması gerekir. Buton grupları için bu role="group" olur, araç çubuklarında ise role="toolbar" olmalıdır.

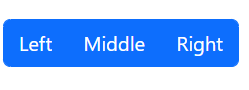
<div class="btn-group" aria-label="Basic example">

<button type="button" class="btn btn-primary">Left</button>

<button type="button" class="btn btn-primary">Middle</button>

<button type="button" class="btn btn-primary">Right</button>

</div>



**Figür 30**

Ayrıca buton grupları ile checkbox veya radiolar da gruplandırılabilir.

<div class="btn-group" role="group" >

<input type="radio" class="btn-check" name="vbtn-radio" id="vbtn-radio1" checked>

<label class="btn btn-outline-danger" for="vbtn-radio1">Radio 1</label>

<input type="radio" class="btn-check" name="vbtn-radio" id="vbtn-radio2" >

<label class="btn btn-outline-danger" for="vbtn-radio2">Radio 2</label>

<input type="radio" class="btn-check" name="vbtn-radio" id="vbtn-radio3" >

<label class="btn btn-outline-danger" for="vbtn-radio3">Radio 3</label>

</div>



**Figür 31**

Yukarıdaki örnekte radiolar gruplandırılmıştır. btn-group class’ı ile gruplanan radiolara ‘group’ rolü tanımlanmıştır. “btn-outline-danger” ile renkleri belirlenmiştir. En soldaki radionun seçili olmasının sebebi ise ikinci kod satırındaki ‘checked’ niteliğidir.

**Cards - Kartlar**

Kartlar farklı tipte birçok içeriği barındıran konteynırlardır. Web sayfalarında bulunan komponentleri oluştururlar. Birkaç farklı etiketi bünyesinde bulundurarak düzeni sağlar.

<div class="card" style="width: 18rem;">

<img src="..." class="card-img-top" alt="...">

<div class="card-body">

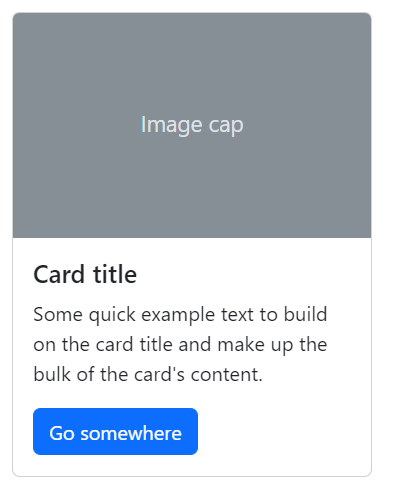
<h5 class="card-title">Card title</h5>

<p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's content.</p>

<a href="#" class="btn btn-primary">Go somewhere</a>

</div>

</div>



**Figür 32**

Yukarıda verilen örnekte 4 farklı yapı bir kart içerisinde birleştirilmiştir. Birinci kod satırında “card” class’ı sayesinde kapsayıcı <div> e card özelliği tanımlanmıştır. “card-body” sınıfı genellikle içerik yazmak içi kullandığımız alan olarak tanımlanmaktadır.<h3>,<p>,<a> gibi etiketleri bu sınıfı bulunduran<div> içerisine yazarız.

Oluşturulan “card” yapılarına üst bilgi ve alt bilgi de eklenebilir.

<div class="card text-center">

<div class="card-header">

Featured

</div>

<div class="card-body">

<h5 class="card-title">Special title treatment</h5>

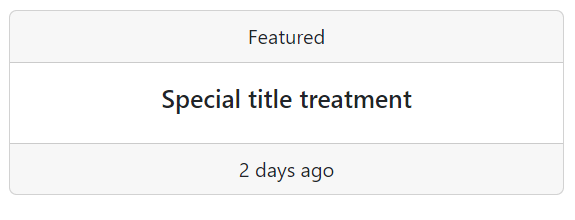
</div>

<div class="card-footer ">

2 days ago

</div>

</div>



**Figür 33**

**Carousel**

Metinler ve resimler arasında geçiş yapmak için kullanılan slayt komponentidir. Slayt döngüsünün gözükmesi için "active" class'ının tanımlanması gerekmektedir. Oluşturulan döngülerde kontrol butonları, işaretçiler ve açıklamalar ile özelleştirmeler yapılabilir.

<div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="false">

<div class="carousel-indicators">

<button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="0" class="active" aria-current="true" aria-label="Slide 1"></button>

<button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="1" aria-label="Slide 2"></button>

<button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="2" aria-label="Slide 3"></button>

</div>

<div class="carousel-inner">

<div class="carousel-item active">

<img src="..." class="d-block w-100" alt="...">

<div class="carousel-caption d-none d-md-block">

<h5>First slide label</h5>

<p>Some representative placeholder content for the first slide.</p>

</div>

</div>

<div class="carousel-item">

<img src="..." class="d-block w-100" alt="...">

<div class="carousel-caption d-none d-md-block">

<h5>Second slide label</h5>

<p>Some representative placeholder content for the second slide.</p>

</div>

</div>

<div class="carousel-item">

<img src="..." class="d-block w-100" alt="...">

<div class="carousel-caption d-none d-md-block">

<h5>Third slide label</h5>

<p>Some representative placeholder content for the third slide.</p>

</div>

</div>

</div>

<button class="carousel-control-prev" type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide="prev">

<span class="carousel-control-prev-icon" aria-hidden="true"></span>

<span class="visually-hidden">Previous</span>

</button>

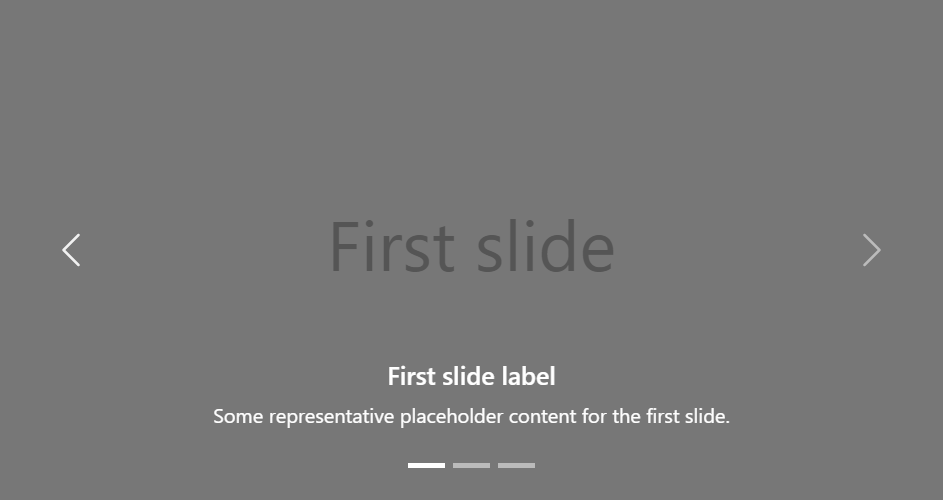
<button class="carousel-control-next" type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide="next">

<span class="carousel-control-next-icon" aria-hidden="true"></span>

<span class="visually-hidden">Next</span>

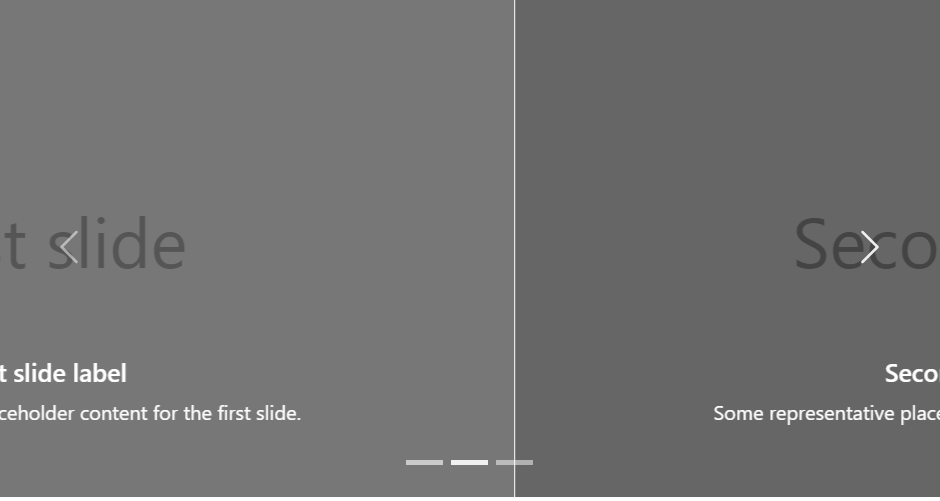
</button>

</div>



**Figür 34**

Verilen örnekte kontrol çubukları, işaretçiler ve açıklamalar kullanılan slayt yapısı oluşturulmuştur. Yukarıda verilen figürün alt kısmında gözüken çizgiler “carousel-indicators” class’ı kullanılarak <button> tipinde oluşturulmuştur. Açıklamaların yazıldığı <div>> in “carousel-caption” class’ına sahip olması gerekir. Aynı zamanda farklı sayfa büyüklüklerinde görünüp görünmemesini ayarlamak için “d-none d-md-block” class’ları kullanılmıştır. Böylelikle açıklamalar “medium” ve daha üzeri sayfa büyüklüklerinde görünür kılınmıştır. Kontrol çubukları ise kod satırının en altında bulunan “carousel-control-next” ve “carousel-control-prev” class’ları kullanan <button> yapıları ile oluşturulmuştur. ‘data-bs-ride="carousel" ‘ niteliği verilen slayt otomatik olarak çalışacaktır.



**Figür 35**

Bu figürde ise birinci slayttan ikinci slayta geçiş gösterilmiştir.

Ek olarak “carosuel-fade” class’ı kullanılarak geçişlerin slayt şeklinde değil de “fade” (solarak değişen) şeklinde gösterilebilir. Geçişler arası bekleme süresi vermek için ‘data-bs-interval="" ‘ niteliği de eklenebilir.

**Close Button – Kapatma Düğmesi**

Uyarı pencerelerine kapatmak için kullanılan kapatma butonlarıdır. "btn-close" class'ı tanımlanarak komponentleri kapatma seçeneği sağlanır.

<button type="button" class="btn-close" aria-label="Close"></button>



**Figür 36**

**Collapse - Daraltma**

Üretilen içeriklerin gösterilmesi veya gizlenmesi için kullanılır. 'data-bs-toggle="collapse" ' niteliği emiri veren etiket için kullanılması gerekirken gizlenilmesi istenilen içeriğin "collapse" class' ına sahip olması gerekir. <button> ve <a> etiketleri için kullanılabilir. Gizlenmek istenilen içeriğin konumu <a> etiketindeki kullanımında ' href" " ' ile verilirken <button> etiketindeki kullanımda 'data-bs-target=" " ' niteliği içerisinde verilir. Görünürlüğü kontrol eden yapı eğer <a> etiketi ise role=”button” tanımlanması gerekir. Ayrıca varsayılan olarak içerik gösterilmek istenmiyorsa kontrol elamanındaki nitelik ‘ aria-expanded="false" ‘ şeklinde tanımlanmalıdır.

<p>

<a class="btn btn-primary" data-bs-toggle="collapse" href="#collapseExample" role="button" aria-expanded="false" aria-controls="collapseExample">

Link with href

</a>

<button class="btn btn-primary" type="button" data-bs-toggle="collapse" data-bs-target="#collapseExample" aria-expanded="false" aria-controls="collapseExample">

Button with data-bs-target

</button>

</p>

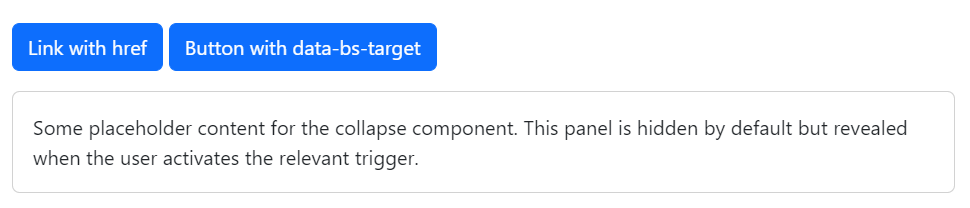
<div class="collapse" id="collapseExample">

<div class="card card-body">

Some placeholder content for the collapse component. This panel is hidden by default but revealed when the user activates the relevant trigger.

</div>

</div>



**Figür 37**

Verile örnekte yukarıdaki kontrol bileşenleri ile etkileşime geçince alt taraftaki açıklama penceresi görünür hale geliyor.

**Dropdowns – Açılır Menüler**

Açılır menüler link listelerini göstermek için kullanılan bir bileşendir. Bootstrapte açılır menüler çeşitli durumlara ve biçimlendirme yapılarına uygulanabilecek şekilde tasarlanmıştır. Örneğin, arama alanları veya oturum açma formları gibi ek girişler ve form kontrolleri içeren açılır menüler oluşturmak mümkündür. Klavye ile menü etkileşimi sağlayan Bootstrap, menü elemanları arasında tek tek geçiş yapmayı destekler.

“dropdown” class’ına sahip bir <div> içerisine basit bir <button> veya <a> etiketi ile açılır menü rahatlıkla kontrol edilebilir. Açılır menü içerisindeki liste elemanları <ul> veya <ol> etiketleri altına yazılabilir.

<div class="dropdown">

<button class="btn btn-secondary dropdown-toggle" type="button" data-bs-toggle="dropdown" aria-expanded="false">

Dropdown button

</button>

<ul class="dropdown-menu">

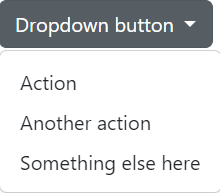
<li><a class="dropdown-item" href="#">Action</a></li>

<li><a class="dropdown-item" href="#">Another action</a></li>

<li><a class="dropdown-item" href="#">Something else here</a></li>

</ul>

</div>



**Figür 38**

Örneği incelediğimizde ilk kod satırında “dropdown” class’ının <div> içerisine tanımlandığı gözükmektedir. <button> bileşeni data-bs-toggle="dropdown" niteliği ile menüyü kontrol etmektedir. Menü içeresindeki liste elemanları ise “dropdown-item” class’ı ile tanımlanmıştır.

Menüyü kontrol eden yapıların görünümünü “dropdown-toggle-split” class’ı kullanarak özelleştirebiliriz.

<div class="btn-group">

<button type="button" class="btn btn-danger">Action</button>

<button type="button" class="btn btn-danger dropdown-toggle dropdown-toggle-split" data-bs-toggle="dropdown" aria-expanded="false">

<span class="visually-hidden">Toggle Dropdown</span>

</button>

<ul class="dropdown-menu">

<li><a class="dropdown-item" href="#">Action</a></li>

<li><a class="dropdown-item" href="#">Another action</a></li>

<li><a class="dropdown-item" href="#">Something else here</a></li>

<li><hr class="dropdown-divider"></li>

<li><a class="dropdown-item" href="#">Separated link</a></li>

</ul>

</div>



**Figür 39**

Bu örnekte açılır menü, kontrol eden yapı olan butonun sağ tarafındaki imleç altında açılmıştır. İşte bu özellik <button> a “dropdown-toggle-split" class’ı tanımlanarak sağlanmıştır. Açılır menüdeki dördüncü liste elementi <hr> etiketi altında “dropdown-divider” class’ı tanımlanmış ayırıcıdır. Bu element sayesinde liste elementleri birbirinden ayrılabilir.

Açılır menüler ile aynı zamanda form yapıları da oluşturmak mümkündür. Dropdown <div> inin içerisine <form> tanımlayarak bu yapılar oluşturulur. Form oluşturma detayları için Form alt başlığını inceleyiniz.

Ek olarak, farklı yönlere açılan menüler ve menü altındaki liste elemanlarının farklı şekillerde hizalanmış olduğu menüler oluşturulabilir.

**List Groups – Liste Grupları**

Liste grupları bir dizi içerik üretmek için kullanılan güçlü bir komponenttir. Liste grupları Bootstrap teknolojisi ile özelleştirilebilir:

1. "active" class'ı tanımlanan liste elemanı seçili olarak gözükür.
2. "disabled" class'ı tanımlanan liste elemanı üzerinde işlem yapılamaz.
3. Kullanıcı etkileşimi oluşturmak için "list-group-item-action" tanımlanan liste elamanı üzerinde kullanılır.
4. “list-group-flush” class'ı tanımlanan listenin kenarlıkları kalkar.
5. “list-group-horizontal” class'ı tanımlanan listenin elemanları dikey olarak hizalanır.
6. Farklı renklerde liste elemanları oluşturmak için list-group-item-" " class'ı liste elemanlarına tanımlanır.
7. Liste elamanlarına nişan (badge) eklenerek özelleştirilebilir.

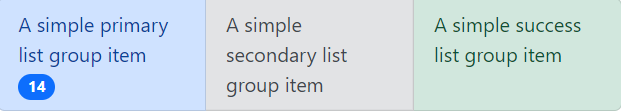
<ul class="list-group list-group-horizontal ">

<li class="list-group-item list-group-item-action list-group-item-primary">A simple primary list group item <span class="badge bg-primary rounded-pill">14</span></li>

<li class="list-group-item list-group-item-action list-group-item-secondary">A simple secondary list group item</li>

<li class="list-group-item list-group-item-action list-group-item-success">A simple success list group item</li>

</ul>



**Figür 40**

Verilen örnekte yukarıdaki bahsedilen özelliklerin çoğu liste özelleştirmesinde kullanılmıştır. Farklı renklerde arka plana sahip dikey olarak oluşturulan bu liste yapısındaki figürde görüldüğü üzere birinci liste elamanında nişan (badge) bulunmaktadır. Aynı zamanda bu liste yapısı kullanıcı etkileşimlidir.

**Modal**

Bootstrap'in JavaScript modal eklentisi kullanıcı dostu web sitelerinde iletişim kutuları oluşturur. Modal yapılarının HTML'deki <body> içerisindeki içerikten önce görüntülenebilmesi için 'tabindex=" " ' değerinin <body> nin sahip olduğu "tabindex" niteliğinin değerinden düşük olması gerekmektedir. Oluşturulan modal yapıları sabit bir pozisyonda bulundukları için "postion:fixed" niteliğini alırlar. Unutulmamalıdır ki modal eklentisi bir JS eklentisi olduğundan dolayı öncelikle JS yazılmalıdır.

Temel olarak modal yapısı 3 ana kısımdan oluşmaktadır. İlk olarak gösterilecek olan içeriğin başlığının yazıldığı modal-header yapısı oluşturulur. Daha sonra gösterilmek istenilen içeriğin bulunduğu kısım modal-body içerisine yazılır. Son olarak ise kullanıcının modal ile etkileşime girmesi için genellikle kapatma veya ileri gibi düğmelerin bulunduğu modal-footer yapısı oluşturulur.

<div class="modal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title">Modal title</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-

label="Close"></button>

</div>

<div class="modal-body">

<p>Modal body text goes here.</p>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

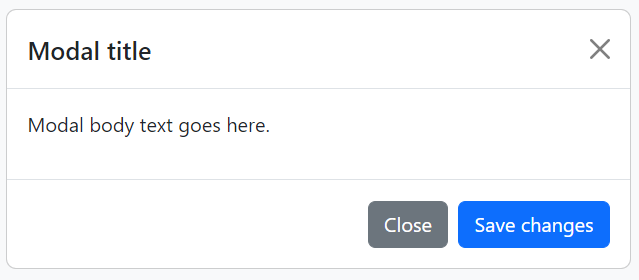
<button type="button" class="btn btn-primary">Save changes</button>

</div>

</div>

</div>

</div>



**Figür 41**

Örnekte verilen yapı ile temel düzeyde bir modal yapısı oluşturulmuştur. “modal” class’ı tanımlanan <div> içerisine “modal-dialog” class’ına sahip bir <div> oluşturularak kullanıcıya gösterilmek istenilen işaret kutusu hazırlanmıştır. Sırasıyla “modal-header” “modal-body” ve “modal-footer” class’larına sahip diğer kapsayıcı elemanlar ile modal yapısı son halini almıştır.

Modal yapısını özelleştirmek için kullanılan farklı class yapıları ve ne işe yaradıkları aşağıda belirtilmiştir:

1. ‘data-bs-backdrop="static"’ niteliği eklenen modal kapsayıcısı, modal dışındaki herhangi bir kullanıcı etkileşimi ile kapatılamaz. Diğer bir deyişle modalı kapatmak için kapatma düğmelerini kullanmak gereklidir.
2. “modal-dialog-scrollable" class’ı “modal-dialog" class’ı ile birlikte kullanılırsa kaydırılabilir modallar oluşturulur.
3. Kullanıcıdan bilgi almak için form yapıları modal içerisinde kullanılabilir.
4. “data-bs-target” ve “data-bs-toggle” nitelikleri kullanılarak modallar arasında geçiş yapabiliriz. Ancak aynı anda iki veya daha fazla modal gösterilemez.

<div class="modal fade" id="exampleModalToggle" aria-hidden="true" aria-labelledby="exampleModalToggleLabel" tabindex="-1">

<div class="modal-dialog modal-dialog-centered">

<div class="modal-content">

<div class="modal-header">

<h1 class="modal-title fs-5" id="exampleModalToggleLabel">Modal 1</h1>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

Show a second modal and hide this one with the button below.

</div>

<div class="modal-footer">

<button class="btn btn-primary" data-bs-target="#exampleModalToggle2" data-bs-toggle="modal">Open second modal</button>

</div>

</div>

</div>

</div>

<div class="modal fade" id="exampleModalToggle2" aria-hidden="true" aria-labelledby="exampleModalToggleLabel2" tabindex="-1">

<div class="modal-dialog modal-dialog-centered">

<div class="modal-content">

<div class="modal-header">

<h1 class="modal-title fs-5" id="exampleModalToggleLabel2">Modal 2</h1>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

Hide this modal and show the first with the button below.

</div>

<div class="modal-footer">

<button class="btn btn-primary" data-bs-target="#exampleModalToggle" data-bs-toggle="modal">Back to first</button>

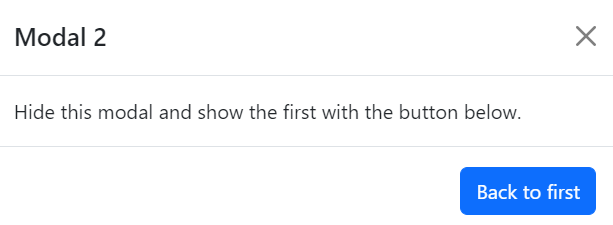
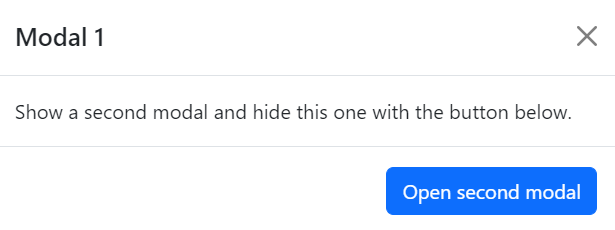
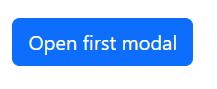
</div>

</div>

</div>

</div>

<a class="btn btn-primary" data-bs-toggle="modal" href="#exampleModalToggle" role="button">Open first modal</a>



**Figür 42**

Bu örnekte iki adet modal arasındaki geçiş “data-bs-target” ve “data-bs-toggle” nitelikleri kullanılarak sağlanmıştır. Böylelikle birinci butona basan kullanıcı ilk modalı açar ve eğer ikinci modalı açmak için buton ile etkileşime geçerse ikinci modal ekranda gösterilir. İkinci modalın “back to first” adlı butonu ile etkileşime girecek olan kullanıcı tekrar birinci modala dönmüş olur.

**Navbar – Navigasyon Çubuğu**

Sitelerin navigasyonunu oluşturan navbar sayesinde web sayfasında gezinmek çok kolay hale gelir. Bootstrap ile özelleşmiş navbar komponenti kullanıcılar için özel bir deneyim sağlıyor. Navbar oluşturmak için genel olarak “navbar” class’ı tanımlanan <nav> etiketi kullanılır. Ekran genişliğine göre değişen (responsive) navbar yapısını geniş ekranlarda tam boyutta görmek için “navbar-expand-‘ ‘ “ class’ı <nav> etiketi içerisine tanımlanabilir. Ekran genişliği daraltıldığında ise navbar yapısının görünmemesi için “collapse navbar-collapse” class’ları kullanılabilir. Navbar içerisindeki her bir eleman (item) “navbar-item” class’ı almaktadır. Ayrıca navbar içerisine form yapıları, butonlar, dropdown menüler yerleştirilebilir.

<nav class="navbar navbar-expand-lg bg-light">

<div class="container-fluid">

<a class="navbar-brand" href="#">Navbar</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav me-auto mb-2 mb-lg-0">

<li class="nav-item">

<a class="nav-link active" aria-current="page" href="#">Home</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#">Link</a>

</li>

<li class="nav-item dropdown">

<a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">

Dropdown

</a>

<ul class="dropdown-menu">

<li><a class="dropdown-item" href="#">Action</a></li>

<li><a class="dropdown-item" href="#">Another action</a></li>

<li><hr class="dropdown-divider"></li>

<li><a class="dropdown-item" href="#">Something else here</a></li>

</ul>

</li>

<li class="nav-item">

<a class="nav-link disabled">Disabled</a>

</li>

</ul>

<form class="d-flex" role="search">

<input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">

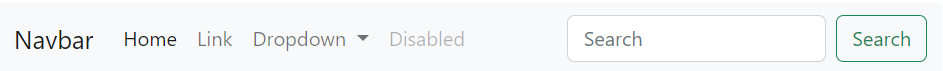
<button class="btn btn-outline-success" type="submit">Search</button>

</form>

</div>

</div>

</nav>



**Figür 43**

Verilen örnekte yukarıdaki açıklamada bahsedilen gibi “navbar-expand-lg” class’ı <nav> etiketi içerisine tanımlanarak “large” ve daha geniş ekranlarda navbar’ın tam boyutu gösterilir. Bu örnekte navbar elemanları liste elementleri olarak düzenlenmiştir. Üçüncü navbar iteminde “dropdown” class’ı ile bir adet dropdown menü oluşturulmuştur. Navbarın son kısımda ise bir adet form yapısı kullanılarak navbar tamamlanmıştır.

Farklı sınıflar yardımı ile özelleştirmeler yapılabilir:

“navbar-brand” class’ı kullanılarak marka, proje adı veya ürünler için isimlendirme yapılabilir.

“navbar-toggler” class’ı kullanılarak Navbar’ın daraltılması(collapse) sağlanılabilir.

“active” class’ı tanımlanan navbar itemi işaretlenir.

**Offcanvas**

Offcanvas kullanıcı deneyimini en üst seviyeye taşımak için JavaScript aracılığıyla değiştirilebilen özelleştirilmiş sidebarlardır. Proje içerisinde navigasyon, alışveriş sepetleri ve daha fazlasını oluşturmak için bu gizli sidebarları kullanabiliriz. JavaScript, "offcanvas" class'ına sahip bir öğenin görünür veya gizli olduğunu kontrol eder. Kontrol elemanı olarak ise genellikle <buton> lar tercih edilir.

<button class="btn btn-primary" type="button" data-bs-toggle="offcanvas" data-bs-target="#offcanvasExample" aria-controls="offcanvasExample">

Button with data-bs-target

</button>

<div class="offcanvas offcanvas-start" tabindex="-1" id="offcanvasExample" aria-labelledby="offcanvasExampleLabel">

<div class="offcanvas-header">

<h5 class="offcanvas-title" id="offcanvasExampleLabel">Offcanvas</h5>

<button type="button" class="btn-close" data-bs-dismiss="offcanvas" aria-label="Close"></button>

</div>

<div class="offcanvas-body">

<div class="">

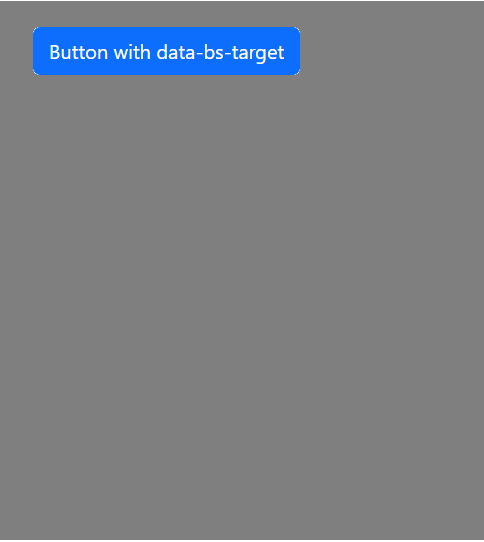
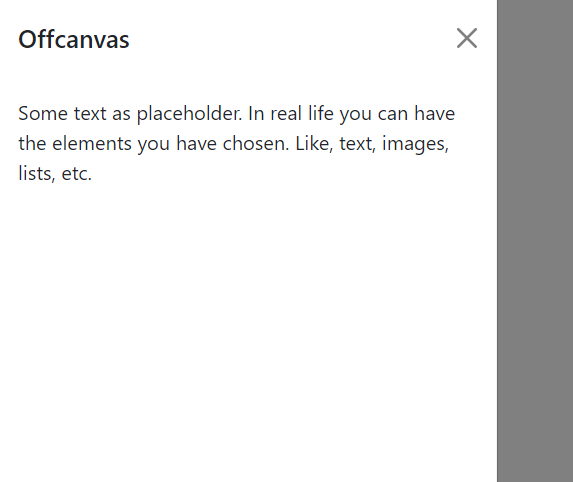
Some text as placeholder. In real life you can have the elements you have chosen. Like, text, images, lists, etc.

</div>

</div>

</div>

</div>

**Figür 44-45**

Yukarıda basit bir offcanvas örneği verilmiştir. Kullanıcı etkileşimi öncesi ekranda sadece buton gözükürken etkileşim sonrası ekranın sol kısmında offcanvas görünür hale gelmektedir. Kontrol butonu kodlarını incelediğimizde data-bs-toggle="offcanvas" niteliği sayesinde kontrol edilecek yapının offcanvas olduğunu anlamaktayız. Bu örnekte JS offcanvas <div>i içerisine girerek “show” class’ını aktif hale getirmiştir ve böylelikle offcanvas ekranda görünmüştür. Offcanvas yapısının ekranın sol tarafında görünmesini sağlayan sınıf ise “offcanvas-start” class’ıdır. Ayrıca çoğu Bootstrap komponentinde olduğu gibi offcanvas da farklı bölümler altına ayrılarak (header, body, footer) özelleştirilebilir.

1. data-bs-scroll=”true” niteliği şeklinde ayarlanırsa offcanvas ekranda olsa bile kullanıcı offcanvas dışındaki ekranda da işlem yapabilir.
2. data-bs-backdrop="static" niteliği şeklinde ayarlanırsa offcanvas ekranda iken kullanıcı offcamvas dışındaki ekranda işlem yapamaz.

**Pagination - Sayfalandırma**

Bir dizi ilgili içeriğin birden çok sayfada bulunduğunu belirtmek için kullanılır. <nav> etiketi içerisine gömülürse ekran okuyuculara ve diğer yardımcı teknolojilere yönelik bir gezinme bölümü olarak tanımlanması sağlanır. Ayrıca sayfalarda aynı türden birden fazla gezinme (navigation) bölümleri olabileceğinden karışıklığı önlemek için spesifik navigasyon belirlemek için aria-label niteliği <nav> etiketi içerisinde kullanılmalıdır. Pagination liste tipinde kullanılan bir komponenttir.

<nav aria-label="Page navigation example">

<ul class="pagination">

<li class="page-item">

<a class="page-link" href="#" aria-label="Previous">

<span aria-hidden="true">«</span>

</a>

</li>

<li class="page-item active"><a class="page-link" href="#">1</a></li>

<li class="page-item"><a class="page-link" href="#">2</a></li>

<li class="page-item"><a class="page-link" href="#">3</a></li>

<li class="page-item">

<a class="page-link" href="#" aria-label="Next">

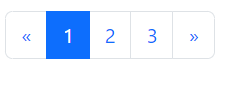
<span aria-hidden="true">»</span>

</a>

</li>

</ul>

</nav>



**Figür 46**

Verilen örnekte <nav> etiketi içerisinde “pagination” class’ına sahip bir sırasız liste görmekteyiz. <nav> etiketi altında aria-label niteliği sayesinde spesifik navigasyon olarak “Page navigation example” belirlenmiştir. Liste elamanları “page-item” class’ı ile tanımlanmakta olup <a> etiketi içerisinde href=” “ niteliği ile gezinmek istediğimiz bölüme link tanımlanmıştır. Ekranda aktif olarak gösterilmek istenen liste elamanına “active” class’ı tanımlanmıştır. Ayrıca ikonları da bu komponent içerisinde kullanabileceğimiz örnekte belirtilmiştir.

Bootstrap ile genişlik, renk, hizalama, aktiflik ve engelleme gibi özelleştirmeler yapılabilir.

**Placeholders – Yer Tutucular**

Bir şeyin hâlâ yüklenmekte olduğunu belirtmek için kullanılırlar. Genişliğini ayarlamak için grid sisteminden yararlanılır. Görünümleri, renkleri ve boyutları, Bootstrap ile kolayca özelleştirilebilir. "placeholder-glow" ve "placeholder-wave" class'ları sayesinde animasyon eklenebilir. Tutulmak istenilen etiket içerisine genellikle <span> etiketi ile "placeholder" ve "col-' ' " class'ları eklenir.

<p class="placeholder-glow">

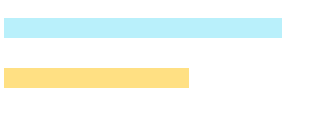
<span class="placeholder col-6 bg-info"></span>

</p>

<p class="placeholder-wave">

<span class="placeholder col-4 bg-warning"></span>

</p>



**Figür 47**

Bu örnekte oluşturulan placeholder komponentleri <p> etiketlerinin yerini tutmaktadır. Eğer yukarıdaki kodu canlı olarak çalıştırırsak fare imlecinin şeklin üzerine geldiğinde sanki bir şeyi yüklüyor gibi dairesel bir imlece dönüştüğünü tecrübe edebiliriz. Ayrıca <p> etiketleri içerisinde bulunan “placeholder-wave” ve “placeholder-glow” class’ları sayesinde yer tutuculara animasyon eklenmiştir. “bg-warning bg-info” class’ları ise Bootstrap’in renk sınıflarıdır.

**Progress – İlerleme Çubukları**

Sitedeki bir öğeninin ilerleme durumunu göstermek için kullanılan komponentlerdir. "progress" class'ı kullanılarak dış kapsayıcı oluşturulur. Oluşturulan dış kapsayıcı içerisine ilerleme çubuğunu eklemek için "progress-bar" class'ı kullanılırken, ilerleme çubuğu üzerinde işlemin ne kadar ilerlediğini göstermek için farklı nitelikler kullanılır:

1. style="width: " niteliği ile ilerleme çubuğu üzerinde işlemin ne kadar ilerlediği hakkında görsel bir sonuç elde edilir.
2. aria-valuenow="" niteliği ile işlemin anlık olarak ne durumda olduğu hakkında görsel olmayan ancak bilgiyi taşıyan bir sonuç elde edilir.
3. aria-valuemin="" niteliği ile işlemin minimum ilerleme değeri atanır.
4. aria-valuemax="" niteliği ile işlemin maximum ilerleme değeri atanır.

Özelleştirme class'ları:

1. “progress-bar-striped” eklenmesi ile ilerleme çubuğu çizgili görünüme sahip olur.
2. “progress-bar-animated” eklenmesi ile ilerleme çubuğu görseline animasyon eklenir.

<div class="progress">

<div class="progress-bar" role="progressbar" style="width: 25%" aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>

</div>

<div class="progress">

<div class="progress-bar progress-bar-striped bg-danger" role="progressbar " style="width: 75%" aria-valuenow="75" aria-valuemin="0" aria-valuemax="100">%75</div>

</div>



**Figür 48**

Örnekte görüldüğü üzere iki adet ilerleme çubuğu vardır. “progress” class’ına sahip dış kapsayıcı içerisine “progress-bar” class’lı ilerme çubuğu oluşturulmuştur. Farklı “aria” nitelikleri ile özelleştirilmişlerdir. Özel olarak ikinci çubuk, “progress-bar-stripped bg-danger” class’larını almış ve sayısal olarak da işaretlenmiştir.

**Spinners – Dönen İşaretler**

Bu komponent ile bir öğenin veya sayfanın yüklenme durumu gösterilir. Erişilebilirlik için oluşturulan yapıda role="status" niteliğine ihtiyaç varken aynı zamanda yapıyı <span class="visually-hidden">Loading...</span>. şeklinde germek gereklidir. Spinner’in genel olarak iki görsel çeşidi vardır. “spinner-border” class’ı ile kenarları dönen bir yapı oluşturulurken, “spinner-grow” class’ı ile yanıp sönen spinner elde edilir.

<div class="spinner-border text-primary" role="status">

<span class="visually-hidden">Loading...</span>

</div><br>

<div class="spinner-grow text-warning" role="status">

<span class="visually-hidden">Loading...</span>

</div><br>

<div class="spinner-border text-success" role="status">

<span class="visually-hidden">Loading...</span>

</div>



**Figür 49**

Örnekte iki farklı görsel yapısındaki spinner komponentleri gösterilmiştir. Bootstrap renk class’ları ile özelleştirilmiş olan bu komponentler bir öğenin ya da sayfanın yüklenme durumunu simgelemek için kullanılır.

**Toasts**

Toast, flexbox ile üretilen mobil ve masaüstü işletim sistemleri tarafından popüler olan tasarlanmış hafif bildirimlerdir. Uyarı mesajı olarak kullanıcılara bildirim sağlar. Toasts kullanıcı etkileşimli bir komponent olduğundan dolayı JS kullanımı gereklidir. Toasts oluştururken “toast” class’ına sahip bir kapsayıcının içerisinde yapıyı özelleştirmek ve daha kolay kontrol etmek için “toast-header”, “toast-body” class’larına sahip kapsayıcılar oluşturulabilir. Toasts kullanıcılar tarafından çalıştırılır. Bunun için de toasts’ı kontrol eden bir butona ihtiyacımız vardır.

Toast ile verilmek istenilen içerik değiştikçe, gecikme zaman aşımını güncelleyerek, kullanıcılara toast'ı okumak için yeterli zaman verilmelidir. Ayrıca içeriğe göre "role" ve "aria-live" nitelikleri de güncellenmelidir. aria-atomic="true" niteliği sayesinde eğer toast içerisinde bir değişiklik yapıldıysa ayrı ayrı parçaları göstermektense direkt tek parça şeklinde uyarı bildiriminin görüntülenmesi sağlanır.

<div role="alert" aria-live="assertive" aria-atomic="true" class="toast" data-bs-autohide="false">

<div class="toast-header">

<img src="..." class="rounded me-2" alt="...">

<strong class="me-auto">Bootstrap</strong>

<small>11 mins ago</small>

<button type="button" class="btn-close" data-bs-dismiss="toast" aria-label="Close"></button>

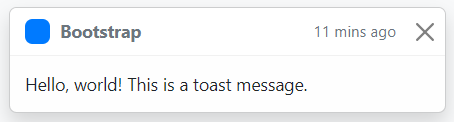
</div>

<div class="toast-body">

Hello, world! This is a toast message.

</div>

</div>



**Figür 50**

Bu örnekte verilen toast iki parçadan oluşmuştur.”toast-header” ile başlık kısmı ve kapatma düğmesi gösterilirken “toast-body” ile kullanıcıya verilmek istenilen mesaj gösterilmiştir. Bbu toast role=”alert” olarak ayarlandığı için kullanıcıya uyarı bildirimi niteliğinde mesaj verilmek istenilmiştir. Aynı zamanda aria-live=”assertive” niteliği kullanılarak iddialı(ciddi) bir uyarı oluşturulmuştur.

**Sonuç**

Yapılan çalışma ile ön uç geliştirmecilerin sıklıkla kullandığı CSS framework’ü olan Bootstrap teknolojisinin dokümantasyonu oluşturulmuştur. Bootstrap’in sağladığı her eklenti örneklerle açıklanmış olup geliştirmecilerin anlayacağı şekilde yorumlanmıştır. Oluşturulan çalışma sayesinde ön uç geliştirmecilerin Bootstrap ortamını tanıması ve bu ortama ayak uydurması amaçlanmıştır. Mevcut teknolojide oluşturulan bu dokümantasyon teknoloji ilerledikçe değişebilir.

Yapılan bu çalışma Bootstrap’i anlama ve yorumlama çalışmasıdır. Detaylı bilgi için [getbootstrap.com](https://getbootstrap.com/docs/5.2) adresini ziyaret ediniz.

**Referans**

Tüm örneklerde (kodlar-figürler) [***https://getbootstrap.com/docs/5.2***](https://getbootstrap.com/docs/5.2)sayfasından yararlanılmıştır.

Detaylı bilgi için bakınız:

[***https://getbootstrap.com***](https://getbootstrap.com)