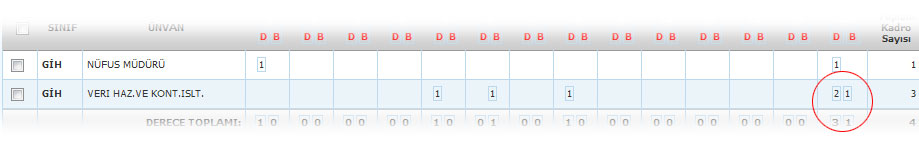
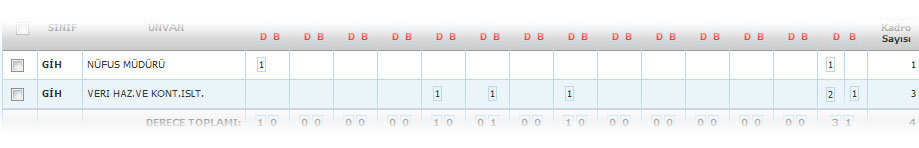
KULLANICI ARAYÜZÜ STANDARTLARI



Bu doküman, ASP.NET ile web tabanlı olarak geliştirilecek olan uygulamalarda uygulanacak ara yüz standartlarını tespit etmek amacı ile hazırlanmıştır. Bu dokümanda yer alan standartların kesin bağlayıcılığı bulunmamakla birlikte öneri niteliğindedir ve takım liderleri ya da proje yöneticileri tarafından, yapılan uygulamanın niteliğine göre değiştirilebilir.

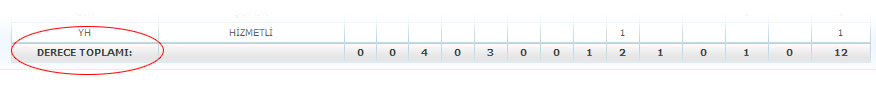
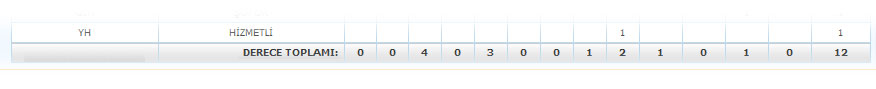
**Listelemeler:**

Verileri listelemek ve gerekli görüldüğünde üzerinde değişiklik yapmak için kullanılır. Değişiklik yapılmasını sağlayan ikon-butonlar ilgili satırın en sağında bulunmalıdır.

1. Aşağıdaki gibi iki ayrı veri içeren sütunlar daha anlaşılır olması için bölünmelidir.  
   Şimdiki haliOlması gereken
2. Sütunların hizalamaları (td align), yazılar sola yanaşık ve sayı-rakamlar ortalanmış şekilde olması için uygulanabilirliği için javascript gerekmektedir. İlgili Css Class ları aşağıdadır.  
     
   td.metin  
   td.tarih  
   td.sayiSolaHizali  
   td.sayiSagaHizali  
   td.sayiOrtali
3. Sayfalama sisteminin bulunduğu bölümlerde, her sayfada kaç kayıt gösterileceğine dair –kullanıcıların seçebileceği- bir drop down seçim alanı olması gerekmektedir (“50 – 100 – 200 …” kayıt göster şeklinde). Sayfalama grid yapılması gerekmektedir. Sona git, başa git butonları eklenmelidir. Listelemenin hem enüstünde hemde en altında bulunması iyi olur. Paging / grid kontolü yapılması gerekiyor.  
     
     
     
   Olması gereken



1. Aşağıdaki gibi “Derece Toplamı” na benzer sonuç bildiren ifadeler sonuç bölümüne yakın sütunda belirtilmelidir.

Olması gereken  


**Veri Giriş Ekranları**

1. Kullanıcılar aynı verileri defalarca girmek zorunda olabilir ve bu kullanıcının hata yapma ihtimalini artırır, bu yüzden de bu tarz verilerin bir yerlerde saklanması ve yeniden kullanımının otomatik sağlanması gerekmektedir.
2. Önemli işlemlerden önce onay ve işlem tamamlandıktan sonra bilgi mesajları görüntülenmelidir (sil – kaydet gibi). Uzman kullanıcılar için onay seçeneğinin bir daha gösterilmemesi şeklinde bir tercih sunulması da faydalı olur. Bununla birlikte işlemle ilgili hata mesajları da kullanıcıya görüntülenmelidir.
3. Kullanıcı her ekranda veri girişi için mümkün olduğunca benzer işlemler yapmalıdır, bu sayfayı ve yapacağı işlemi algılaması açısından önemlidir.
4. Mümkünse veri giriş alanları odaklanmaya göre işaretlenmelidir. Örneğin TC Kimlik no girişi yapılırken ilgili textbox diğerlerinden farklı görünmelidir. Master Page e javascript yazılması gerekiyor.
5. Zorunlu alanlardaki veri başlıklarının (lablel) kırmızı renkte olması gerekiyor.
6. Sekmeli (tab) işlemlerde, tutarlı olabilecek ise kullanıcı önceki adımlardaki verileri tekrar düzenleyebilmelidir. Basit işlemler için gereksiz yere tab kullanılmamalıdır.
7. Sekmeli (tab) işlemlerde veri giriş bölümlerinin en altında “bir sonraki adıma” geç yada “kaydet – ilerle“ şeklinde butonlar olması kullanıcının ne yaptığına ve bir sonraki aşamada ne yapması gerektiğine yardımcı olur, “şimdi ne yapmam gerekiyor?” sorusunu sormasına izin vermez.
8. Tekrarlanan veri girişlerinde, girilen verilerin her işlemin sonunda liste halinde (sil – değiştir işlemleri yapılabilecek bir şekilde) gösterilmesi kullanıcı için faydalı olur. Daha önce de belirtildiği gibi; sıkça tekrar eden veri girişlerinde kullanıcının hata yapması yüksek ihtimaldir. Yukarıda belirtildiği gibi bir listeleme, kullanıcının hatalı girdiği veriyi hemen silmesine/düzeltmesine imkan tanır. Kullanıcının talep etmesi halinde yapılacak.
9. Uzun sayısal veri giriş bölümlerinde ilgili alanın kısa gruplara bölünmesi kullanıcı için çok rahatlatıcı olabilir (Masked textbox). Örnek: 111-111-1111-111  
     
   User kontrol yazılacak, textbox larda max lenght sınırı olacak.
10. Bir kullanıcı örneğin 4 karakter uzunluğunda olması gereken sayısal bir alana iki karakter girmiş ise bunu otomatikman tamamlamak kullanıcı için rahatlatıcı olabilir. Örneğin “56” girmiş ise sistem otomatikman bunu “0056” yapabilmelidir.  
      
    User kontrol yazılacak, textbox larda max lenght sınırı olacak.
11. Textboxlar için kullanıcıya kalan karakter sayısını gösteren sayaç yapılacak.
12. Karmaşık hiyerarşik veri girişinin yapılacağı ekranlarda kullanıcıya yardım amaçlı metinler hatta görseller sunulabilmelidir. Gerekirse sekmeler yardımı ile wizard kullanılmalıdır.
13. Label bölümlerinde mutlaka bir ayraç kullanılmalıdır.   
    Örnek:   
    Olması gereken;  
     İsim : \_\_\_\_\_\_\_\_\_   
    Yanlış kullanım;   
    İsim \_\_\_\_\_\_\_\_\_

1. Rakam/sayı kullanılan alanlarda mümkünse hane sayısını belirtir veri giriş alanı kullanılması gerekir.

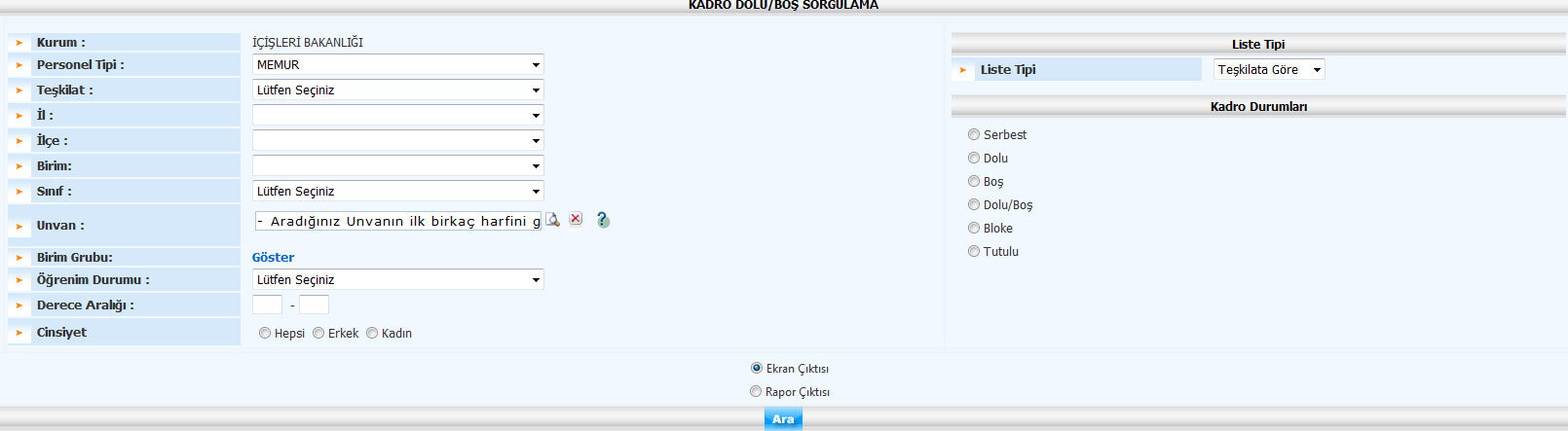
Doğru kullanım;

Numara :\_ \_ \_ \_ \_

Yanlış kullanım;

Numara:

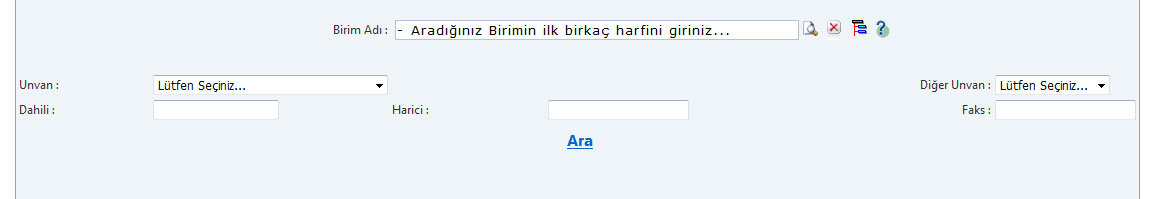
Numara (5 karakter):

1. Çoklu seçim yada birden fazla nesne içinden tekli seçim yapılan alanlarda seçimin/seçimlerin diğer seçilmemiş nesnelerden daha farklı görünmesi yada yapılan seçim/seçimlerin başka bir bölümde kullanıcıya bildirilmesi gerekmektedir.
2. Uzun metin ile yapılan veri girişlerinde kullanıcılar işlem sonunda ön izleme şeklinde görüntüleme yapabilmeli ve ön izlemeden sonra işlemi sonlandırmalıdır. Ayrıca silmek kopyalamak gibi işlemleri de imleç(mouse) ile yapabilmeleri de büyük bir rahatlık sağlayacaktır. Bu tarz ekranlarda kullanıcının imleci(mouse) daha etkin kullanabilmesi tercih edilmelidir.
3. Ara – Kaydet gibi sayfanın altında bulunan butonların scrollbar çıkmadan görünebilmesi için mümkün olduğunda yerden tasarruf etmek amacıyla satırların eşit şekilde iki sütuna bölünerek konumlandırılması gerekmektedir.  
     
   Çok uzun sayfalarda butonlar hem altta hem de üstte olmalıdır.  
     
   

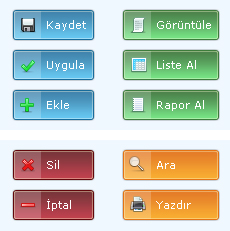


1. Ara – Kaydet gibi sayfanın altında bulunması gereken butonlar th içerisinde değil, td içerisinde olması gerekmektedir.

**Görsel Hatalar**

1. “Textbox – combobox – textarea” gibi component bölümlerinin sağındaki “label” bölümlerinin arka planları aşağıdaki örnekte belirtildiği gibi olmalıdır.  
     
   Yanlış yerleşim  
     
   Doğru yerleşim  
   
2. “Ekle – tarihe göre” gibi linkler **bilgi başlıkları** bölümündeki “koyu mavi” alanda değil, **veri giriş** bölümündeki “açık mavi” alanda bulunmalıdır.
3. Yardım konusunun incelenmesi gerekiyor.

**Butonlar**

****

Butonlar sayfanın en altında bulunmalıdır ve 4 ana gruba ayrılmıştır. **Button tagı verilmelidir**. İlgili butonlar için ileride yazılacak skinlerden skin idlere ulaşabilirsiniz. Buton isimleri/işlemleri için örnekler aşağıdadır. İsmi/işlevi bulunmayan butonları onay – sonuç – iptal – genel kategorilerine göre sınıflandırabilirsiniz. Bu bölümde anlatılanlar aynı işlemi yapan ve component bölümünde bulunan linkbutton yada imagebuttonlar için geçerli değildir.

Butonların bulunduğu satır th yerine td olmalıdır.

* Onay Butonları
  + Kaydet (SkinID="buttonKaydet")
  + Uygula (SkinID="buttonUygula")
  + Ekle (SkinID="buttonEkle")
* Sonuç
  + Görüntüle (SkinID="buttonGoruntule")
  + Liste Al (SkinID="buttonListeal")
  + Rapor Al (SkinID="buttonRaporal")
* İptal
  + Sil (SkinID="buttonSil")
  + İptal (SkinID="buttonİptal")
* Genel
  + Ara (SkinID="buttonAra")
  + Yazdır (SkinID="buttonYazdir")

**Linkbuttonlar**

Component bölümünde textbox – dropdown vb. yanında bulunan linklerdir. Linkbutton tagı verilmelidir.

**İkonlar**

Bazı componentlerin sağında ve özellikle liste şeklinde sonuç görünümlerinin en sağındaki sütunda yer alan imagebuttonlardır. Imagebutton tagı verilmelidir. İlgili ikonlara aşağıdaki skinid lerden ulaşabilrisiniz.



 SkinID="ButtonAciklama"

SkinID="ButtonAnasayfa"

 SkinID="ButtonAra"

 SkinID="ButtonDagitim"

 SkinID="ButtonEkle"

SkinID="ButtonEkliDosya"

 SkinID="ButtonGeri"

 SkinID="ButtonGeriGonder"

 SkinID="ButtonGuncelle"

 SkinID="ButtonIleri"

 SkinID="ButtonIptal"

SkinID="ButtonIsBitir"

SkinID="ButtonIslem"

 SkinID="ButtonKapat"

SkinID="ButtonListele"

SkinID="ButtonOnayla"

 SkinID="ButtonRapor"

 SkinID="ButtonSaat"

SkinID="ButtonTakvim"

SkinID="ButtonYardim"

SkinID="ButtonYenile"

SkinID="ButtonYonlendir"

**Genel Açıklamalar**

1. Her şey, uygulama hakkında hiçbir bilgisi olmayan bir kişinin herhangi bir sayfaya baktığında “Evet, bu bir ….. butonu/linki ” diyebileceği kadar açık olmalıdır. Nesneler kullanıcıyı fazla düşündürmemelidir, “Tamam, …. orada ve bu da bir …. Ve benim istediğim şey de burada” diyebilmelidir.
2. Kısacası sayfaya bakarken düşünce baloncukları çok az olmalı ve hiç birinin içinde soru işareti olmamalıdır. İşimiz soru işaretlerini yok etmek olmalıdır.
3. Kullanıcıların web sitelerini nasıl kullandıklarını düşünmemiz ile onların nasıl kullandıkları arasında çok büyük farklar vardır. Özenerek günlerce uğraşarak yazılan uzun uzun metinleri/açıklamaları okumazlar. Kullanıcı gerçeği; 100 km hızla yanından geçtiğiniz bir ilan tahtasını ne kadar detaylı okuyabildiğinizle aynıdır. Bizim gerçeğimiz; o ilan panosunun mükemmel bir broşür/metin olduğu şeklindedir.
4. İşin gerçeği böyle iken mümkün olduğunca kısa ve çarpıcı açıklamalar, uyarılar yazmamız gerekir.

* Her sayfada net bir şekilde görülebilir bir hiyerarşi sağlamaya çalışmalıyız,
* Sayfaları net bir şekilde parsellemeliyiz (tanımlanmış bölgelere ayırmalıyız),
* Neyin tıklanabilir olduğunu/olmadığını belirginleştirmeliyiz,
* Gürültüyü/karmaşayı en aza indirmeliyiz

1. Kullanıcılar kendilerini düşündürmeyen seçenekleri severler. Bu noktada da kaç kere tıklayarak sonuca ulaştıkları önemsizleşir, yeter ki her tıklama düşündürmeyen bir seçenek olsun.

Doğru kullanım:

Ev

Ofis

Yanlış kullanım:

Adres1

Adres2

1. Web de yazmanın temel şartı, “her sayfada bulunan kelimelerin yarısını atın, sonra da geri kalan yarısını atın” dan geçer.

* Sayfadaki gürültüyü/karmaşayı azaltır,
* İçeriğin daha belirgin olmasını sağlar,
* Sayfaları daha kısa yapar ve kullanıcı için sıkıcı olan scrollbar ı çoğu zaman ortadan kaldırır.
* Kullanıcılar talimatları asla okumazlar, sadece şaşkınca defalarca tekrarlanmış başarısız girişimler sonucunda okurlar. Dolayısıyla kendiliğinden anlaşılır sayfalar yapmaya çalışmamız gerekir.

1. 15. yy. da Leonardo da Vinci’ nin bulduğu sekmeler (tab) hala kullanılmaktadır ve fiziksel bir alan izlenimi uyandırırlar. Çünkü bunları fiziksel dosyalarda da hala kullanmaktayız. Kendiliğinden anlaşılırlar, kullanıcı ilk defa bilgisayar kullanıyor olsa bile asla “şunlar ne işe yarıyor acaba?” sorusunu sormaz. Kolay kolay gözden kaçmazlar ve şıktırlar.
2. Bir web site de “herkes ….. yı sever” her sayfanın solunda bulunan menüleri seviyoruz çünkü alıştığımız bildiğimiz yerindeler ve onlardan nefret ederiz çünkü çok sıkıcılar. Her sayfada flashlara bayılıyoruz çünkü eğlenceli ve şıklar ve onlardan nefret ediyoruz çünkü yavaş yükleniyorlar. Her sayfada ….. ları çok seviyoruz yada sevmiyoruz. Genelde çözümsüzlük bir web sitesinde, neyin iyi durduğuna dair farklı fikirleri olan, bir toplantı odası dolusu kişinin saatlerce tartışmasından kaynaklanmaktadır. Tüm web kullanıcılarının bizimle aynı şeyi sevdiklerini düşünmekle başlayan hatalar silsilesinden kurtulmamız gerekir.   
     
   Tasarımcı, kullanıcıların görsel sitelerden hoşlandığını düşünür. Yazılımcılar, kullanıcıların hoş özelliklere sahip sitelerden hoşlandıklarını düşünürler. Bu iki grup bir araya geldiğinde tartışmaya ve beklide gücenmeye yol açar. Tartışmaların sonunda ise varılan kanı; sorunu çözebilecek “ortalama bir kullanıcı/kullanıcıların” bulunmasıdır. İşin aslına baktığımızda ise tüm web kullanıcıları özeldir ve özgündür, birbirlerine asla benzemezler. Çözüm ise çok basittir; dikkatlice düşünülmüş, uygulanmış ve denenmiş ve sadece ihtiyacı karşılayan bir tasarım/projedir.