

## ASP.NET 2.0'da Kişiselleştirme Kavramı ve

### Bir kişiselleştirme Uygulaması

Şehra Şen, Ata Önal, Ayşegül Alaybeyoğlu

Ege Üniversitesi, Bilgisayar Mühendisliği Bölümü

sehra\_sen@yahoo.com, ata.onal@ege.edu.tr, aysegul.alaybeyoglu@ege.edu.tr

**Özet:** ASP .NET 2.0 teknolojileri kullanılarak verilerin kişiselleştirmeye izin verecek bağımsız birimler halinde sunulduğu bir web uygulaması gerçekleştirimi yapılmıştır. Gerçekleştirilmiş olan uygulama kapsamında Ege Üniversitesi web sitesi üzerinde yer alan “egedyuru” isimli hizmet web part bileşeni olarak yeniden tasarlanarak kişiselleştirilebilir bağımsız bir birim olarak sunulması sağlanmıştır. Uygulama gerçekleştiriminde kullanılan teknolojiler anlatılmaktadır.

**Abstract:** The goal of the project developed is to design and implement a personalizable web application using ASP .NET 2.0 technologies. In this context the service named “egedyuru” on the web site of Ege University was reimplemented as a web part component. This service is presented as a personalizable, independant module on a portal. Furthermore, a roadmap including the technologies used at implementation is formed.

**Anahtar Kelimeler:** ASP .NET 2.0, Web Part, Kişiselleştirme.

#### 1. Giriş

Günümüzde web siteleri büyük bilgi kaynaklarıdır. Kötü tasarlanmış web siteleri kullanıcılarının bilgi gereksinimlerini yeterince karşılayamayabilmektedir. Bu nedenle günümüzde portallar, genellikle verileri belirli bir derecede kişiselleştirmeyi sağlayacak biçimde bağımsız birimler olarak düzenlenmektedir. Portal kullanıcılarına bu bağımsız birimleri kendi bireysel çalışma biçimlerine uygun olarak düzenleyebilme olanağı sunulmaktadır.

Web kişiselleştirme bir web sitesinin sunumunun, web sitesi tarafından sağlanan bilgi veya servislerin bir veya bir küme kullanıcının açık veya üstü kapalı tercihlerine uygun biçimde değiştirilebilmesidir [9][3]. Web kişiselleştirme bir web kullanıcısının kendi portal web sayfasının içeriğini ve görünümünü uyarlayabilme yeteneğidir.

Günümüzdeki web yapısı ile bilgiye erişim genellikle kullanıcıların siteler üzerinde gezin-

mesini gerektirmektedir. Oysa kişiselleştirilmiş bir sayfa bilgiyi kullanıcılara getirmektedir. Böylece kullanıcı istediği bilgiye istediği zamanda erişebilmektedir. Kişiselleştirilmiş bir sayfa ile web üzerinde dağılmış olan bilginin kullanılarak kullanıcıya özel bilgi alma kaynağının oluşturabilmesi yeteneği oldukça değerlidir [4].

AJAX teknolojisi kullanılarak oluşturulmuş başlangıç sayfaları özelleştirilebilir, kolayca kullanılabilen ve genellikle tarayıcının başlangıç sayfasında olması gereken sayfalardır. AJAX başlangıç sayfaları tarayıcı açıldığında kullanıcının tercihleri doğrultusunda düzenlenmiş olduğu şekilde göstermektedir. Günümüzde birçok başlangıç sayfası vardır. Bu sayfalardan bilinen bazıları şunlardır: Netvibes, Pageflakes, Live.com, iGoogle, My Yahoo.

ASP . NET 2.0 teknolojileri ve AJAX kullanılarak belirli ölçüde kişiselleştirmenin sağlanabildiği sayfalar tasarlamak mümkündür.

## 2. Web Part

Web Part bileşenleri kullanılarak oluşturulmuş bir aspx sayfası, sayfa üzerindeki web part'ların yönetilmesinden sorumlu bir web part manager, web part'ların içerisinde bulunduğu zonlar ve web part'lardan oluşmaktadır.



Şekil 1. Web Part'ları kullanan bir aspx sayfasının mimarisi

### 2.1. Web Part Sınıfının Yapısı

WebPart sınıfı üç arayüzü gerçekleştirilmektedir:

- *IWebPart*: Bir web part'in çekirdek özelliklerini tanımlar. Title, Description, Height ve Width gibi.
- *IWebActionable*: Bir web part'in sağladığı *verb*'leri tanımlar.
- *IWebEditable*: Web part'in bazı özelliklerini yönetmek için özel editör part'ları sağlayan bir web part'i tanımlar.

Tüm web part'lar bu üç arayüzü gerçekleştirdiğinden, portal çatısının her bir parçası tüm web part'lar ile etkileşimde bulunabilmektedir. Örneğin, bir sayfa ilk görüntülediğinde her bir web part'in yetkisi aynı sayfa üzerinde bulunan web part manager'a verilmektedir. Web part manager bir web part'in yetkisini aldığı anda, ilgili web part'in özellikleri hakkında herhangi bir şey bilmek zorunda değildir. Bir web part manager sadece, kontrolün title ve verbs gibi ayırtedici özelliklere sahip olduğunu bilmektedir. Çünkü web part manager, gerektiğinde

bu arayüzleri bir web part'in sahip olduğu yetenekleri belirlemede kullanabilmektedir.

## 3. Web Part Manager

WebPartManager kontrolünün görevi web form üzerindeki WebPart ve WebPartZone'lar için yönetimsel bir altyapı sunmaktır. WebPart kontrolleri içeren her sayfa bir WebPartManager kontrolüne sahip olmalıdır. Bu kontrol sayfanın herhangi bir yerine bırakılabilir çünkü çalışma zamanında görünür bir arayüze sahip değildir.

WebPartManager belirli bir anda bir sayfa üzerinde hangi web part'ların olduğunu, onların hangi zone'lara ilişkin olduğunu ve kullanıcının onları görme yetkisinin olup olmadığını bilir. WebPartManager ayrıca bir web part'in sayfaya eklenmesi veya çıkarılması için gerekli işlemleri gerçekleştirir. Bir web part'in bir zone'dan diğerine taşınması da WebPartManager tarafından gerçekleştirilir. WebPartManager'ın görev ve faaliyet'leri 5 kategori altında sınıflandırılabilir [6].

### 3.1. Web Part'ları İzlemek

WebPartManager belirli bir anda bir sayfa üzerinde hangi web part'ların olduğunu, onların hangi zone'lara ilişkin olduğunu ve kullanıcının onları görme yetkisinin olup olmadığını bilir. WebPartManager ayrıca bir web part'in sayfaya eklenmesi veya çıkarılması için gerekli işlemleri gerçekleştirir. Bir web part'in bir zone'dan diğerine taşınması da WebPartManager tarafından gerçekleştirilmektedir.

Bir kullanıcı kontrolü gibi bir web part'in sayfaya eklenmesi, özel bir web part'in sayfaya eklenmesinden farklıdır. Çünkü WebPartManager öncelikle o anda kullanılan kontrolden bir GenericWebPart yaratmalıdır.

### 3.2. Kişiselleştirme Bilgisini Yönetmek

Sayfanın yaşam döngüsünün başında WebPartManager kişiselleştirme sisteminden verileri

alır, sonra bu verileri ilgili web part'lara dağıtır. Sayfanın yaşam döngüsünün sonunda WebPartManager tüm web part kontrollerinden kişiselleştirme verilerini toplar, verileri paketler ve saklanmak üzere kişiselleştirme sitemine gönderir. Bu süreç, kullanıcı tercihlerinin tarayıcının yeniden başlatılmasından sonra da kalıcı olmasını sağlamaktadır.

### 3.3. Yaşam Döngüsü Olaylarını Kontrol Etmek

Web part sayfa yaşam döngüsü süresince oluşan olaylar WebPartManager tarafından izlenir ve meydana gelir. Örneğin; uygulamada her bir zone'un, en çok 4 web part içerebilecek biçimde sınırlandırılması istenebilir. Kullanıcı 5. web part'ı zone'un içerisine sürüklemeye çalıştığı anda, taşıma işleminin iptali için WebPartMoving olayı kullanılabilir [6].

Özel davranışları sağlamak ve uygulamalar içerisindeki yaşam döngüsünü sağlamak için, toplamda WebPartManager tarafından meydana getirilen 20 yaşam döngüsü olayı vardır.

#### 3.3.1. Sayfa Yaşam Döngüsü:

Ne zaman bir ASP .NET web sayfası isteğinde bulunulsa, web sunucu üzerinde bu isteği karşılayacak bir sınıf yaratılmaktadır. Bu sınıf Page sınıfı olarak atıfta bulunduğumuz sınıftır. Sayfa isteği ele almayı bitirdiğinde, bu sınıf - en azından sayfa için yeniden istekte bulunulana kadar - yok edilir. Sayfanın yaratıldığı ve yok edildiği zaman arasında Sayfa Yaşam Döngüsü olarak bilinen bir dizi olay gerçekleşir. Sayfa Yaşam Döngüsü kontrollerin ne zaman ilkleneceği veya geri dönüş olaylarının ne zaman gerçekleşeceği gibi durumların meydana geldiği bir olay modelidir.

Bir zone ilk yaratıldığında yapılan ilk işlemlerden biri zone'un kendini o andaki sayfanın WebPartManager kontrolüne kaydettirmesidir. Sonra WebPartManager tarafından ilkleme safhasının tamamlanıp tamamlanmadığı kontrol edilir. Safha tamamlanmışsa WebPartManager InvalidOperationException istisnası atar.

Bu nedenle, ilkleme safhasından herhangi bir zaman sonra aynı işlem yapılmaya çalışılırsa hata mesajı alınır. InitComplete safhasından sonra sayfaya bir zone eklenemeyeceği gibi, connection'ların etkinleştirildiği PreRender evresinden önce bir connection üzerinden istenen veriler de okunamaz.

#### 3.3.2. Sayfa Görünüm Modlarını Değiştirmek:

Bir web sayfası beş standart görünüm moduna girebilmektedir: BrowseDisplayMode, CatalogDisplayMode, ConnectDisplayMode, DesignDisplayMode ve EditDisplayMode. Bu modlar WebPartManager tarafından belirlenmekte ve web part'lar üzerinde güçlü işlemler gerçekleştirilebilmesi için bir yöntem sunmaktadır.

Tüm görünüm modları soyut WebPartDisplayMode temel sınıfından türetilmektedir. Bir web sayfasının görünüm modları WebPartManager'ın DisplayMode özelliğinin değiştirilmesi ile belirlenmektedir.

Her bir görünüm modunda kullanıcının belirli bir küme işlemi gerçekleştirmesine izin veren farklı kullanıcı arayüzü elemanları vardır. Sayfa görünüm modları ve onlara ilişkin işlemler şöyledir [6]:

- *BrowseDisplayMode* : Bu bir web part sayfasının varsayılan modudur. Bu modda kullanıcılar sadece sayfanın üzerindeki web part'ları görebilmektedir.
- *CatalogDisplayMode* : Bu modda kullanıcılar web part'ları zone'lara arasında taşıyabilmektedir. Ayrıca kullanıcılara bir galeriden web part seçme ve seçilen web part'ları sayfaya ekleme olanağını tanıyan özel kullanıcı arayüzü elemanları da vardır.
- *ConnectDisplayMode* : Bu modda web part'ları birbirine bağlayabilmeleri için kullanıcılara özel kullanıcı arayüzü elemanları gösterilmektedir. Ayrıca bağlanabilme özelliği taşıyan web part'lar için

tüm web part'lara connect verb'i eklenmektedir.

- *DesignDisplayMode* : Browse moduna benzemekle birlikte, kullanıcılara web part'ları zone'lar arasında taşıma olanağı vermektedir.
- *EditDisplayMode* : Bu mod kullanıcılara web part'ların özelliklerini ve öz niteliklerini düzenleme olanağı tanımaktadır. Bunu yapmak için kullanıcılara özel kullanıcı arayüzü elemanları gösterilmektedir. Bu modda ayrıca kullanıcılar web part'ları zone'lar arasında taşıyabilirler.

WebPartManager sınıfı, ayrıca belirli bir anda sayfanın hangi modlarda olabileceği bilgisini veren SupportedDisplayModes isimli bir özellik sunmaktadır.

Web part kontrolleri çalışma zamanında sayfa içine veya dışına aktarılabilir. Oluşturulmuş olan bir web sitesine yeni bir web part eklemek için tüm uygulamanın yeniden yayımlanması gerekmemektedir, çünkü portal çatısı web part'ların Control Description Files olarak bilinen XML tanım dosyaları aracılığıyla içe ve dışa taşınmasına izin vermektedir. Çalışma zamanında kullanıcıların bu XML tanım dosyalarına göz atmasını ve içe aktarmasını sağlayan ImportCatalogPart olarak bilinen standart bir catalog part bile vardır.

Bir web part'ın XML formatı onun yüksek derecede taşınabilir olmasını sağlamaktadır. Bir kurumda birden fazla web uygulaması olduğunda, web part'ların paylaşılabilmesi için ayrılması uygun olacaktır. O zaman, her bir uygulamanın sadece o uygulamanın işlevine özel olan web part'ları içermesi gerekecektir. Daha genel olan part'lar tüm uygulamaların erişilebilmesi için merkezi bir depoda tutulmalıdır. Bu web part'lar haberler ve hava durumu gibi part'lar olabilir. Bu genel part'lar XML formatında merkezi servis aracılığıyla bir web servis

veya ortak paylaşımındaki bir dosya kullanılarak oluşturulabilir. Gelişmiş kullanıcıların karmaşık web part'ları özelleştirmesine ve sonrasında bu değişiklikleri bir XML dosyasında saklamasına izin veren uygulamalar yaratılabilir. Bu dosya yapılan değişiklikleri tutabilir ve sonrasında diğer kullanıcılara gönderilebilir veya tüm kullanıcılar tarafından erişilebilmesi ve içe aktarılabilmesi için merkezi bir depoya aktarılabilir.

#### 4. Kişiselleştirme

Geçmişte gelişmiş kişiselleştirme yetenekleri sağlayan her site, sitenin kullanıcılarının kişiselleştirme bilgisini saklamak için kendi veritabanı sistemini kontrol etmek ve kullanıcı web sitesine döndüğünde kişiselleştirme bilgisini veritabanlarından çıkarıp yeniden uygulamak için karmaşık kodlar yazmak zorundaydı [7]. ASP .NET 2.0'da veritabanları yaratmak ve çalışma mantığını oluşturmak için gerekli olan işler bizim için tamamlanmıştır. Bu elemanlar ortaklaşa Personalization olarak bilinen bir küme servis olarak mevcuttur.

Personalization, bir web sayfası üzerindeki kontroller üzerinde yapılan özelleştirmeleri gösteren verileri saklamaktan, onlara erişmekten ve onları yeniden uygulamaktan sorumlu olan bir uygulama servsidir. Personalization servisi bir sayfa için özelleştirme verilerini nasıl saklayacağını ve bir kullanıcı aynı sayfayı yeniden istediğinde onlara nasıl erişeceğini bilir. Kayıtlı özelleştirme verileri olan bir sayfa yeniden istendiğinde, kişiselleştirme servisi verileri alıp getirir, böylece istekte bulunan kullanıcı için yeniden yaratılabilir [5].

Kişiselleştirme (Personalization): Kişiselleştirme verileri bir sayfa üzerindeki kontroller için saklanmaktadır ve özel kullanıcılara özgüdür. Kişiselleştirme verileri sayfaya yapılan ziyaretler arasında korunmaktadır. Profil verilerinden farklı olarak, kişiselleştirme verileri web part kontrollerinde yapılan tercihlere özgüdür

ve belirli bir sayfada yapılan değişikliklere bağlanmıştır [9].

Kişiselleştirme, diğer durum mekanizmalarından farklıdır, çünkü sayfa bazında kullanıcı kimliğine bağlanmıştır, uzun ömürlüdür ve bu nedenle bir tek kullanıcı oturumu dışında da korunmaktadır. Bir kullanıcının kimliğine ve bir sayfanın adresine dayanarak verileri saklamanın yanısıra, kişiselleştirme, kapsam adı verilen bir kavramı da göz önüne almaktadır [8]. Kapsam kişiselleştirme verilerindeki bir değişiklikten sadece değişikliği yapan kullanıcının mı, yoksa tüm portal kullanıcılarının mı etkilendiğini göstermektedir.

#### 4.1.1. Kişiselleştirme Kapsamı:

ASP .NET portallarında iki kapsam bulunmaktadır: paylaşılmış kapsam (shared scope) ve kullanıcı bazlı kapsam (per-user scope). Bir sayfa paylaşılmış kapsam modunda iken verilerde yapılan değişiklikler, portal içerisindeki tüm kullanıcılar tarafından görülebilir olmaktadır. Sayfa paylaşılmış kapsam modundayken bir web part bir zone'dan bir diğerine taşındığında, portal'ın tüm kullanıcıları bu değişiklikten etkilenmektedir. Bir sayfa kullanıcı bazlı kapsam modundayken portal üzerinde yapılacak bir değişiklik sadece değişikliği yapan kişi tarafından görülebilir olmaktadır. Bu nedenle, paylaşılmış kapsam modunda değişiklik yapabilme yeteneği, yönetsel ayrıcalıkları olan portal kullanıcılarına verilmektedir.

WebPartManager'ın Personalization özelliği aslında sayfaya ilişkin WebPartPersonalization sınıfının bir örneğidir. Mantığın çoğunu sağlayan sınıf budur ve portaldaki alt seviye kişiselleştirme işlemlerini gerçekleştirmek üzere gerçekleştirilmiştir.

**WebPartPersonalization Sınıfı:** WebPartPersonalization sınıfı portal uygulamalarındaki web part kontrolleri üzerinde gerçekleştirilen kişiselleştirme işlemleri için gereken mantığı içermektedir. Bazen WebPartManager, kişisel-

leştirme işlemlerini - kontrollerden kişiselleştirme değişikliklerini çıkarmak ve daha önce üzerlerinde kişiselleştirme değişiklikleri yapılmış olan kontrollere kişiselleştirme verilerini yeniden uygulamak gibi - sunucu kontrolleri üzerinde gerçekleştirmek zorunda kalmaktadır. WebPartManager, bu görevleri WebPartPersonalization sınıfı aracılığıyla gerçekleştirmektedir. Bir nesneye, ortak kişiselleştirme işlemlerini gerçekleştirmek için - o andaki kişiselleştirme kapsamını değiştirmek veya bir web sayfası için kişiselleştirme verilerini sıfırlamak gibi- kod içerisinde erişilebilmektedir.'de yer alan tabloda WebPartPersonalization sınıfının önemli genel (public) parçaları gösterilmektedir. Bu genel (public) parçalar kişiselleştirme görevlerini gerçekleştirmek için kullanılabilir. Genel (public) parçalara ek olarak WebPartPersonalization sınıfı, web part kontrollerinden çıkarılan veya onlara yeniden uygulanan kişiselleştirme verileri hakkında daha alt seviye davranışları değiştirmek için ezilebilen korumalı (protected) parçalar içermektedir.

#### Kişiselleştirme Verilerinin Yaşam Döngüsü:

Kişiselleştirme verilerinin yaşam döngüsünü anlamak bize her şeyin nasıl olduğunu anlama olanağını sunmaktadır. Böylece kişiselleştirmeyi alt seviyede etkilemek gereksinimi doğduğunda nereye kod yazılabileceği bilinebilir. Kişiselleştirme aslında iki safhalı bir süreçtir:

- İlk safhada WebPartPersonalization sınıfının web sayfası üzerindeki her bir web part kontrolü için durum verileri toplanmaktadır.
- İkinci safhada ise sayfa üzerindeki kontrollerden toplanan veriler saklanmak üzere bir veritabanına gönderildiğinde meydana gelmektedir.

Şekil 2'de, bir web sayfasına yapılan ziyaretler arasında, sayfa üzerindeki web part kontrolleri için kişiselleştirme verileri bir veritabanında saklanmaktadır. Veritabanı ile haberleşme işle-

mi, kişiselleştirme verilerini bir veritabanından okuma veya veritabanına yazma için gerekli olan metotları tanımlayan PersonalizationProvider tarafından ele alınmaktadır.



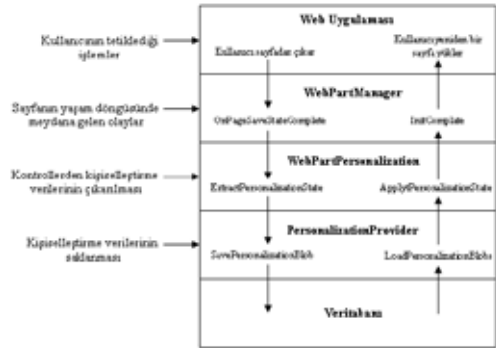
Şekil 2. Kişiselleştirme süreci

### Kişiselleştirme Verilerinin Saklanması:

Çalışma zamanında, kişiselleştirme verilerini saklama ve geri yükleme işlemi WebPartManager tarafından başlatılmaktadır, çünkü sayfanın olay yaşam döngüsü boyunca portalın davranışı WebPartManager tarafından yönetilmektedir. İşlem bir kullanıcının portal içerisindeki bir sayfayı ziyaret etmesiyle başlamaktadır. Bu meydana geldiğinde, WebPartManager sayfa yaşam döngüsünün InitComplete safhası tamamlanana kadar bekler ve sonrasında WebPartPersonalization sınıfının sayfa üzerindeki kontrollere var olan kişiselleştirme verilerini uygulamasını bekler. Bunu yapmak için, WebPartPersonalization sınıfı PersonalizationProvider'dan kişiselleştirme verilerini ister.

Sayfa yaşam döngüsünün sonunda - OnPageSaveStateComplete safhası sırasında - WebPartManager sayfa üzerindeki kontrollerin durumunu saklamak için WebPartPersonalization sınıfındaki Save metodunu çağırır. WebPartPersonalization sınıfı sayfa üzerindeki tüm web part kontrollerinin kişiselleştirme verilerinin bir araya toplar, ve sonrasında saklamak üzere PersonalizationProvider sınıfına iletir.

Şekil 3 kişiselleştirme verilerinin, bir web sayfasındaki kontrollere karşı, nasıl saklandığını ve yüklendiğini gösteren bir özet sağlanmaktadır.



Şekil 3. Kişiselleştirme verilerinin nasıl saklandığı ve yüklendiği [6]

Görüldüğü üzere, yaşam döngüsünde PersonalizationProvider sınıfının rolü kişiselleştirme verilerinin saklanması ve alınması veri erişimi ile sınırlandırılmıştır.

**Personalization Provider Sınıfı:** PersonalizationProvider sınıfı soyut (abstract) olarak işaretlenmiştir. Kişiselleştirme verilerini saklamak ve almak için bir soyut sınıf tanımlanması ASP .NET 2.0'ın yaygın bir genişletilebilirlik desendir.

PersonalizationProvider sınıfından türetme yaparak bir geliştirici, özel bir PersonalizationProvider sınıfı yaratabilir ve geliştiricinin kullanmakta olduğu veritabanındaki veriye erişmek için gerekli olan mantığı gerçekleştirebilir.

Çalışma zamanında veriler saklanacağı veya yükleneceği zaman, portal çatısı PersonalizationProvider base sınıfına çağrıda bulunur ve bu nedenle temeldeki veritabanının tam tipinden haberdar değildir.

## 5. Uygulama

EgeDuyuru servisini veren uygulama bir ascx kontrolü olarak hazırlanmıştır. Böylelikle uygulama (ascx kontrolü), web part manager ve web part zone kontrollerinden oluşturulmuş olan web sayfası içerisindeki zone'lardan bi-

rine sürüklenip bırakıldığında, bu kontrolden GenericWebPart kontrolü oluşturulacaktır. Web sayfası içerisine sürüklenip bırakıldıktan sonra EgeDuyuru servisini veren web part'in görünümü Şekil 4'teki gibidir. Web part'in içerisinde yer aldığı sayfa görünümü Şekil 5'te görülmektedir.



Etkinliğin Adı	Konu	Düzenleme Tarihi	Düzenleme Yeri	Gönderen Birim
Aylık Program	<a href="#">E.Ü. ATATÜRK KÜLTÜR MERKEZİ HAZİRAN 2008 PROGRAMI</a>	30.06.2008	E.Ü. ATATÜRK KÜLTÜR MERKEZİ	Rektörlük-Basin ve Halkla İlişkiler Şube Müdürlüğü
Aylık Program	<a href="#">E.Ü. KAMPUS KÜLTÜR MERKEZİ HAZİRAN 2008 PROGRAMI</a>	30.06.2008	E.Ü. KAMPUS KÜLTÜR MERKEZİ	Rektörlük-Basin ve Halkla İlişkiler Şube Müdürlüğü
Kurslar	<a href="#">I. TIPTA TEZ YAZIMI KURSU</a>	21.06.2008	E.Ü. Tıp Fakültesi Merkez Araştırma Laboratuvarı (AREL) Eğitim Komisyonu	Rektörlük-Basin ve Halkla İlişkiler Şube Müdürlüğü
Kurultay	<a href="#">DOKUZ EYLÜL ÜNİVERSİTESİ V. AKTİF EĞİTİM KURULTAYI</a>	07.06.2008	DOKUZ EYLÜL ÜNİVERSİTESİ	Rektörlük-Basin ve Halkla İlişkiler Şube Müdürlüğü

Şekil 4. EgeDuyuru web servisini kullanan uygulama görünümü

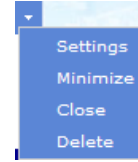


Şekil 5. Admin rolündeki bir kullanıcı için UsersPage.aspx sayfasının görünümü

EgeDuyuru servisi Ege Üniversitesi'nin sisteminden yapılan duyuruları, belirli bir formatta kullanıcılarına sunmak için kullanılmaktadır. Ancak üniversitenin web sayfası üzerinde yer alan mevcut uygulamada kullanıcılar, görmek istedikleri duyurulara her defasında sayfa üzerinde arama yaparak ulaşabilmektedirler. Yeniden gerçekleştirimi yapılan uygulamada ise ASP .NET 2.0'daki kişiselleştirme olanaklarından faydalanılarak kullanıcının kendi sayfasında görmek istediği duyuru seçeneklerini be-

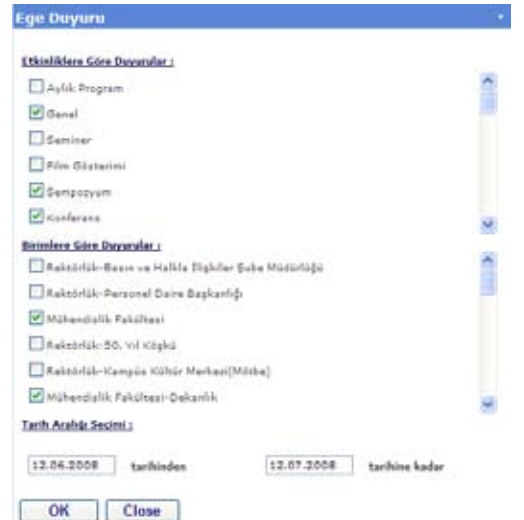
lirleyebilmektedir. Bu seçenekler, kullanıcıya ait sayfada korunmaktadır.

EgeDuyuru uygulamasının kullanıcı tarafından tercihlerine uygun şekilde kişiselleştirilebilmesini sağlamak için verbs menüsüne Settings isimli yeni bir verb eklenmiştir (Bkz. Şekil 6). Bu verb'e tıkladığında kullanıcının kişisel tercihlerine uygun ayarlamaları yapabileceği bir ekran görüntülenmektedir (Bkz. Şekil 7). Bu ekranda yapılan seçimlere göre sonuçlar filtrelenmektedir.



Şekil 6. Settings isimli verb

Kullanıcı bu listeler üzerinde sayfayı her açtığında görüntülenmesini istediği seçimleri yapabilmektedir. Ayrıca kullanıcı belirli tarihler arasındaki etkinlikleri görüntülemeyi seçebilmektedir. Kullanıcı tercihlerine uygun seçimler yapıldıktan sonra "OK" butonuna tıklanarak saklanmaktadır.



Şekil 7. Kullanıcı tercihlerinin belirlendiği ekran görünümü

Kullanıcıya, veritabanında yer alan etkinlik ve birimler, yetkileri kapsamında CheckBox kontrolleri listesi olarak gösterilmektedir. Kullanıcı yetkileri, yönetici sayfasında yer alan “rol” ve “grup” bilgileri işaretlenerek, sistem yöneticisi tarafından belirlenmektedir. EgeDuyuru uygulamasının kullandığı veritabanına yeni bir duyuru kaydı girilirken o duyuruyu hangi “rol” ve “grup” yetkisine sahip olan kullanıcıların görebileceği bilgisi “rol” ve “grup” alanlarına giriş yapılarak belirlenmektedir.

Şekil 7’de kullanıcı, “12.06.2008” ve “12.07.2008” tarihlerinde gerçekleşecek olan, “Mühendislik Fakültesi” ve “Mühendislik Fakültesi-Dekanlık” birimlerine ait “Genel”, “Sempozyum”, “Konferans” etkinlik türlerindeki duyuruları sayfasını her ziyaret edişinde görebilecektir. “OK” butonu yapılan seçimleri kaydetmek için kullanılırken “Close” butonu ile seçim penceresi kapatılmaktadır.

Kod içerisinde EgeDuyuru sınıfının UserControl, IWebActionable arayüzlerini gerçekleştirmesi sağlanmıştır. Etkinlik türleri, Birim isimleri, Başlangıç Tarihi ve Son Tarih seçimlerinin kişiselleştirilmesi için gerekli bildirimler yapılmıştır (Bkz. Çizelge 1).

**Çizelge 1.** EgeDuyuru uygulaması için Etkinlik, Birim, Başlangıç Tarihi ve Son Tarih seçimlerinin kişiselleştirilmesi:

```
private string startDate = "";  
[WebBrowsable(true)]  
[WebDescription("StartDate")]  
[WebDisplayName("StartDate")]  
[Personalizable(PersonalizationS  
cope.User, false)]  
public string StartDate  
{  
    get { return startDate; }  
    set { startDate = value; }  
}  
  
private string endDate = "";
```

```
[WebBrowsable(true)]  
[WebDescription("EndDate")]  
[WebDisplayName("EndDate")]  
[Personalizable(PersonalizationS  
cope.User, false)]  
public string EndDate  
{  
    get { return endDate; }  
    set { endDate = value; }  
}  
  
private string activity;  
[WebBrowsable(true)]  
[WebDescription("Activity")]  
[WebDisplayName("Activity")]  
[Personalizable(PersonalizationS  
cope.User, false)]  
public string Activity  
{  
    get { return activity; }  
    set { activity = value; }  
}  
  
private string department;  
[WebBrowsable(true)]  
[WebDescription("Department")]  
[WebDisplayName("Department")]  
[Personalizable(PersonalizationS  
cope.User, false)]  
public string Department  
{  
    get { return department; }  
    set { department = value; }  
}
```

**Çizelge 2.** Verb menüsüne Settings başlıklı verb’in eklenmesi:

```
WebPartVerbCollection IWebActionable.Verbs  
{  
    get  
    {  
        WebPartVerb Settings;  
  
        Settings = new  
WebPartVerb("Settings",
```

```
        "showSettings(\'" +  
PanelSettings.ClientID + "\')");  
        Settings.Text = "Settings";  
  
        WebPartManager wpm =  
        WebPartManager.  
GetCurrentWebPartManager(Page);  
        if (wpm != null && wpm.  
DisplayMode !=  
        WebPartManager.BrowseDisplayMode)  
        {  
            WebPartVerb[] verbs =  
new WebPartVerb[1];  
            verbs[0] = Settings;  
            WebPartVerbCollection  
verbCollection =  
            new WebPartVerbCollection(verbs);  
            return verbCollection;  
        }  
        return null;  
    }  
}
```

## 6. Sonuç

ASP.NET 2.0 diğer geliştirme ortamlarından farklı olarak kolay bir biçimde kişiselleştirilebilen portal uygulamaları geliştirmek için hazır kontroller ve ara birimler sunmaktadır. ASP.NET 2.0'da portal web part'lardan oluşur. Web part kontrolleri kullanılarak oluşturulan bir portal üzerinde yapılan değişiklikler kullanıcı bazlı kaydedilmektedir.

Uygulama kapsamında ASP.NET 2.0 ve AJAX teknolojileri kullanılarak kullanıcılarının bilgi gereksinimlerini daha etkin bir biçimde karşılayabilmeleri için site üzerinde yer alan bir takım hizmetlerin ayrı birimler olarak sunulduğu, belirli bir oranda kişiselleştirmenin sağlanabildiği bir web sitesi tasarlanmış ve oluşturulmuştur. Böylece web sitesinde sunulan bilgi içeriğine alternatif bir erişim yöntemi sunulmuştur.

Gerçekleştirilen çalışmada web sitesi kullanıcılarının bilgi gereksinimlerinin daha etkin bir

biçimde sağlanabilmesi hedeflenmiştir. Proje-nin geliştirilmesi sırasında izlenen adımlar bel-gelenerek ortaya bir yol haritası çıkarılmıştır.

## Kaynaklar

- [1] Ayers, D., Bruchez E., Fawcwt J., Vernet A., Vlist E. V. D., Web 2.0 Teknolojileri, Alfa Yayınları, 2007.
- [2] Demirkol, Z., C# ile ASP .NET 2, Pusula Yayıncılık, 2007.
- [3] Eirnaki, M., New Approaches to Web Personalization, PhD Thesis, Athens University of Economics and Business, Dept. of Informatics, May 2006.
- [4] Kravatz, H., Designing Web Personalization Features, STC 2000, Orlando, Florida.
- [5] Lee, C. C., Xu, W., Category-based Web Personalization System, Computer Software and Applications Conference, 2001. COMP-SAC 2001, 25th Annual International, 8-12 Oct. 2001, pp. 621 – 625.
- [6] Neimke, D., ASP .NET 2.0 Web Parts in Action, Manning Publications, October, 2006.
- [7] Ramakrishnan, N., PIPE: Web Personalization by Partial Evaluation, Internet Computing, IEEE, November-December 2000, Volume: 4, Issue: 6, pp. 21-31.
- [8] Tam, K. Y., Ho, S. Y., Web Personalization: is it Effective?, Res. Center for Electron. Commerce, Hong Kong Univ. of Sci. & Technol., China, IT Professional, September-October 2003, Volume: 5, Issue: 5, pp. 53 – 57.
- [9] Thomson, L., A Standard Framework for Web Personalization, 1st International Workshop on Innovations In Web Infrastructure (IWI 2005), May 2005.