

Mobil Programlamaya Giriş

Orçun Madran

www.madran.net

orcunmadran@gmail.com

Bu eser Creative Commons Atıf 4.0 Uluslararası Lisansı ile lisanslanmıştır.

İstenildiği gibi çoğaltılabilir, paylaşılabilir, üzerinde değişiklik yapılabilir ve ticari amaçla kullanılabilir.

Kasım 2017

İçindekiler

İçindekiler

2.3 Mobil Programlama

2.3.1. Mobil Uygulama Geliştirmeye Giriş

2.3.1.1. Mobil Programlamadaki Temel Kavramlar

2.3.1.2. Uygulama Geliştirirken Kullanılan Tasarım Yapıları

2.3.2. Mobil Donanım

2.3.2.1. Donanım Bileşenleri

2.3.2.2. Donanım Bileşenlerinin Çalışma Mantıkları

2.3.2.3. Donanım Bileşenlerinin Programlanması

2.3.2.4. Mobil Uygulama Geliştirme Araçları

2.3.2.5. Mobil İşletim Sistemleri

2.3.2.6. Günümüzün Mobil İşletim Sistemleri

2.3.2.7. Mobil Cihazlarda Yer Alan Sensörler

2.3.2.8. Mobil Cihazlarda Yer Alan Sensörlerin Çalışma Mantıkları

2.3.2.9. Emülatörlerin Çalışma Mantıkları.

2.3.2.10. Geliştirilen Mobil Uygulamaların Emülatörlerde Çalıştırılması

2.3.3. Uygulama Geliştirme

2.3.3.1. Uygulamaların Yaşam Döngüleri.

2.3.3.2 Tamamen Web Tarayıcıda Çalışan Uygulamalar için Yaşam Döngüsü

2.3.3.3 Yerel (Native) Uygulamalar için Yaşam Döngüsü

2.3.3.4 Web ve Yerel Bileşenleri İçinde Barındıran Karma (hibrid) Uygulamalar için Yaşam Döngüsü

2.3.4. Mobil Uygulama Geliştirme

2.3.5. Web Tabanlı Mobil Uygulama Geliştirme

2.3.5.1 Web Tabanlı Mobil Uygulama Geliştirme Ortamı

2.3.5.2 Web Tabanlı Örnek Bir Mobil Uygulama Geliştirme

2.3.5.3 Mobil Uygulamanın Kişisel Bilgisayar Üzerinde Yer Alan İnternet Tarayıcı ile Test Edilmesi

2.3.5.4 Mobil Uygulamanın Mobil Cihaz Üzerinde Yer Alan İnternet Tarayıcı ile Test Edilmesi

2.3.5.5 Mobil Uygulamada Test Aşamasında Yaşanabilecek Sorunlar ve Çözümleri

2.3.5.6 Mobil Uygulamada Kullanılan Harici Kütüphanelerin İnternet Bağımlılıklarının Sona Erdirilmesi

2.3.6 Web Tabanlı Mobil Uygulamanın Yerel (Native) Bir Uygulama Olarak Paketlenmesi

2.3.6.1 Mobil Uygulamanın Paketlenmesi için Kullanılacak Bulut Platformları

2.3.6.2 Mobil Uygulamanın GitHub Kod Deposuna Yüklenmesi

2.3.6.3 Paketleme İşleminin Gerçekleştirilmesi için PhoneGap Build Platformunun Kullanılması.

2.3.6.4 APK Uzantılı Kurulum Dosyasının Mobil Cihaza Aktarılması ve Kurulumun Gerçekleştirilmesi

2.3 Mobil Programlama

2.3.1. Mobil Uygulama Geliştirmeye Giriş

İnternet bilgiye erişim süreçleri üzerinde çok önemli değişikliklere yol açmıştır. Bilgi kaynaklarına erişim ağ üzerinden gerçekleşmeye başladığı andan itibaren mekandan bağımsız olarak bilgiye erişim de mümkün hale gelmiştir. İnternet'e bağlı bir bilgisayardan çok farklı hizmetlere (e-devlet, e-finans, e-ticaret, vb.) ve kaynaklara erişilmesi ve bunun sağlamış olduğu avantajlar, kullanıcıların bu sürece mobil ortamda da devam etme taleplerini beraberinde getirmiştir. Bu talepleri mobil platformlarda karşılayabilmek için farklı yazılım çözümleri hayata geçirilmiştir. Bu çözümlerden mobil cihazlardaki işletim sistemleri için özel olarak geliştirilen yazılımlar Mobil Uygulama olarak adlandırılır.

2.3.1.1. Mobil Programlamadaki Temel Kavramlar

Mobil Programlama, uygulamanın çalışacağı platformun özelliklerine uygun olarak özellikle çalışma performansı açısından dikkatli bir geliştirme sürecine ihtiyaç duymaktadır. Mobil cihazların sınırlı donanım özellikleri ve kısıtlı enerji kaynakları nedeniyle, geleneksel geliştirme yaklaşımının dışına çıkılması gerekmektedir.

2.3.1.2. Uygulama Geliştirirken Kullanılan Tasarım Yapıları

Mobil uygulama geliştirirken kullanılabilecek farklı tasarım yapıları bulunmaktadır. Bu tasarım yapılarının seçiminde uygulamanın kullanım alanı, hedefi, mobil cihazların donanım özellikleri, proje süresi ve maliyeti gibi çok farklı değişkenlerin devreye girdiği rahatlıkla söylenebilir. Mobil programlamada yerel geliştirme (native) olarak ifade edilen mobil işletim sistemine özel olarak kodlanan uygulamalar olduğu gibi, tamamen Web tabanlı olarak geliştirilen uygulamalar ya da hibrid olarak adlandırılan yerel ve Web tabanlı uygulamaların iç içe kullanıldığı yapılar da bulunmaktadır.

2.3.2. Mobil Donanım

Mobil donanım, standart bilgisayar mimarisi ile paralellik içerir ve daha fazlası olarak çeşitli sensörlere sahiptir.

2.3.2.1. Donanım Bileşenleri

Mobil donanımlarda bulunan temel bileşenler aşağıda listelenmiştir:

a. İşlemci
b. Anakart
c. Bellek
d. Ekran
e. Depolama alanı

Bu temel bileşenlere ek olarak çeşitli çoklu ortam bileşenleri de mobil cihazların olmazsa olmaz donanımları arasına girmiştir. Bu bileşenlerden bazıları (kameralar, mikrofonlar vb.) aynı zamanda algılayıcı (sensör) olarak da kullanılmaktadır:

a. Ön kamera
b. Arka Kamera
c. Hoparlör
d. Mikrofon

Mobil cihazların dış dünya ile iletişim kurabilmesi açısından gerekli olan bağlantı modülleri de önemli mobil donanım bileşenleri olarak öne çıkmaktadır:

a. Kablosuz ağ bağlantısı
b. Bluetooth
c. GPS
d. NFC
e. USB
f. USB Type-C

Tüm bunlara ek olarak mobil donanımı belki de diğer bilgisayar sistemlerinde ayıran en önemli özellik sahip oldukları algılayıcılardır (sensörler). Bu algılayıcıları

insanlarda bulunan duyu organlarına benzetmek mümkündür. Bir mobil cihazın kendi bulunduğu konumu, yeryüzüne göre aldığı açıyı, ışık koşullarını ve benzeri diğer birçok bilgiyi toplayabilmesini algılayıcılar sağlamaktadır.

2.3.2.2. Donanım Bileşenlerinin Çalışma Mantıkları

Mobil cihazlardaki donanım bileşenlerinin çalışma mantığı, temel bilgisayar çalışma mantığı ile aynıdır. Sadece mobil cihazların güç yönetimi, ısı dağılımı ve diğer sınırlılıklarını dikkate almak gerekir. Tüm elektronik sistemlerde olduğu gibi, tüm işlemlerin gerçekleştirildiği ve genel denetim işlemlerinin yapıldığı bir merkezi işlem birimi bulunmaktadır. Bu merkezi işlem birimi günümüz teknolojisinin izin verdiği sınırlar çerçevesinde çoğunlukla çoklu çekirdeğe sahiptir ve ek olarak görüntü işleyen ek bir işlem birimi de artık çoğu mobil cihaz için standart hale gelmiştir. İşlemlerin geçici olarak işlendiği bellek, yapılan çalışmaların kaydedildiği depolama alanı ve tüm donanımın birbiri ile bağlantısının kurulduğu anakart genel yapıyı oluşturur.

2.3.2.3. Donanım Bileşenlerinin Programlanması

Donanım bileşenlerinin programlanması, genel olarak ilgili donanım bileşeni için oluşturulmuş kütüphanenin uygulama içerisinden çağrılarak kullanılması şeklinde gerçekleştirilir. Mobil cihazlarda çalışacak işletim sistemlerinin geliştirme ortamlarında donanıma özel olarak hazırlanmış fonksiyonlar donanım kütüphanesi olarak adlandırılır.

2.3.2.4. Mobil Uygulama Geliştirme Araçları

Mobil uygulama geliştirirken kullanılacak araçları üç kategoride değerlendirmek mümkün olabilir. Bu kategoriler:

1. Yazılım geliştirme ortamı
2. Uygulama test ortamı
3. Uygulama yayınlama platformu

Her kategori içerisinde yer alacak araç öncelikle mobil uygulamanın çalışacağı işletim sistemi açısından farklılık gösterebilmektedir. Yazılım geliştiriciler, geliştirme aracı seçiminde hedefledikleri işletim sistemini veya uygulama platformunu dikkate alarak araç seçiminde bulunurlar.

2.3.2.5. Mobil İşletim Sistemleri

Mobil işletim sistemi, diğer tüm bilgisayar sistemlerinde olduğu gibi cihazın en etkin şekilde işletilebilmesi için donanıma özel olarak geliştirilmiş ana yazılımdır. Mobil cihazların donanım yapısındaki farklılıklar, onların geleneksel bilgisayarlarda kullanılan işletim sistemlerini kullanmalarına olanak vermez. Bu açıdan gerek algılayıcıların kontrolü açısından, gerekse de işlemci gücü vb. gibi sınırlılıklar açısından mobil cihazlara özel olarak geliştirilen işletim sistemleri kullanılmaktadır.

2.3.2.6. Günümüzün Mobil İşletim Sistemleri

Mobil cihazların gelişim süreci içerisinde ana akım platformlara bağlı olarak birçok işletim sistemi kullanılmıştır. Bu işletim sistemleri içerisinde bazılar gelişimlerine devam ediyorken bazıları ise artık kullanılmamaktadır. Uzun yıllar mobil telefon pazarında lider konumda bulunan Nokia firmasının cihazlarında kullanılan Symbian işletim sistemi, artık kullanılmayan mobil işletim sistemleri arasında en bilinenidir.

Günümüzde en çok tercih edilen mobil işletim sistemleri aşağıda listelenmiştir:

- Android
- iOS
- Windows Phone

Bu işletim sistemleri aynı zamanda mobil cihazlar için ana kategorileri de oluşturmaktadır. Google tarafından birçok farklı donanım üreticisinin kullanımına sunulan Android, çok çeşitli cihazlarda kullanılması açısından yaygınlığı en fazla olan mobil işletim sistemidir. Bunu sadece Apple tarafından üretilen mobil cihazlarda kullanılabilen iOS işletim sistemi takip etmektedir. Pazar payı günümüzde en düşük

olan işletim sistemi ise Microsoft tarafından farklı donanım üreticilerinin de kullanımına sunulan Windows Phone işletim sistemidir.

2.3.2.7. Mobil Cihazlarda Yer Alan Sensörler

Mobil donanımı belki de diğer bilgisayar sistemlerinde ayıran en önemli özellik sahip oldukları algılayıcılardır (sensörler). Bu algılayıcıları insanlarda bulunan duyu organlarına benzetmek mümkündür. Bir mobil cihazın kendi bulunduğu konumu, yeryüzüne göre aldığı açıyı, ışık koşullarını ve benzeri diğer birçok bilgiyi toplayabilmesini algılayıcılar sağlamaktadır. Mobil donanımlar içerisinde en çok yer verilen ve artık neredeyse standart hale dönüşen algılayıcılar aşağıda listelenmiştir:

a. Parmak izi sensörü
b. Üç eksenli jiroskop
c. İvmeölçer
d. Yakınlık sensörü
e. Ortam ışığı sensörü
f. GPS

2.3.2.8. Mobil Cihazlarda Yer Alan Sensörlerin Çalışma Mantıkları

Mobil cihazlarda yer alana sensörlerin çalışma mantıkları, donanım bileşenlerinin çalışma mantıkları ile paralellik gösterir. Mobil işletim sistemlerinin geliştirme paketleri içerisinde sensörlere özel olarak geliştirilmiş kütüphaneler bulunur. Bu kütüphanelerin uygulama içerisinden çağırılması ile birlikte ilgili sensöre erişim sağlanır ve kontrol edilebilir bir yapı ortaya çıkar.

2.3.2.9. Emülatörlerin Çalışma Mantıkları.

Emülatörler, fiziki donanıma sahip olan cihazların işlevlerini taklit edebilen yazılımlar olarak ifade edilebilirler. Geliştirilen uygulamaların test edilebilmesi için farklı donanım özelliklerine sahip birçok cihaz yerine bu şekilde yapılandırılabilen yazılımların kullanılması hem zaman hem de bütçeleme açısından önemli avantajlar sağlar. Buna ek olarak fiziki donanımlarda rahatlıkla oluşturamadığınız birçok anlık

durumu emülatörlerde oluşturabiliriz. Örneğin bir mobil cihazın pili bitmek üzereyken yapacağı işlemleri test edebilmek için emülatörü buna göre yapılandırmak mümkün iken fiziki donanıma sahip bir cihazda pilin bitmesini beklememiz gerekir.

Emülatörlerin sağlamış olduğu birçok avantaja rağmen geliştirme ortamı olarak kullanılan bilgisayar sistemine ek yük getiriyor olması en önemli dezavantajları arasında yer alır. Buna ek olarak her zaman için emülatörde sorunsuz çalışan bir uygulamanın fiziki donanım üzerinde sorun yaşaması mümkündür. Bu açıdan özellikle ticari projelerde geliştirme aşamalarında sıkça kullanılan emülatörler uygulama hayata geçirilmeye yakın yerlerini fiziki donanımlara bırakırlar.

2.3.2.10. Geliştirilen Mobil Uygulamaların Emülatörlerde Çalıştırılması

Uygulama geliştirme araçlarının içinde entegre olarak yer alan emülatörlerin kullanımı oldukça basittir. Uygulama geliştirildikten sonra derlenip emülatöre otomatik olarak gönderilir ve emülatör, geliştirme için kullanılan bilgisayar içerisinde sanki gerçek bir mobil cihaz gibi kullanılabilir.

2.3.3. Uygulama Geliştirme

Uygulama geliştirme, belirli kuramsal temeller üzerine oturtulmuş yol haritaları ile gerçekleştirilebileceği gibi yazılım dünyasında yer almak isteyen her geliştiricinin kendine göre oluşturduğu farklı metodolojilerden de oluşabilir. Birçok yazılımcının birlikte çalıştığı yazılım projeleri ile tek başınıza geliştirdiğiniz küçük uygulamaların geliştirilme süreçleri arasında önemli farklılıklar da bulunur.

Yukarıda belirtilenlere ek olarak aynı hedef platform için geliştirilen bir uygulamanın mimarisi de çok farklı şekillerde kurgulanabilir. Günümüzde bir mobil uygulama geliştirirken kullanabileceğiniz farklı uygulama geliştirme mimarileri bulunmaktadır. Bunlar içerisinde en çok kabul görenler aşağıda sıralanmıştır:

- Tamamen Web tarayıcıda çalışanlar
- Yerel (native) uygulama olarak geliştirilenler

- Web ve yerel bileşenleri içinde barındıran karma (hibrid) uygulamalar

2.3.3.1. Uygulamaların Yaşam Döngüleri.

Uygulama geliştirme yaşam döngüleri, uygulama geliştirme mimarisinde tercih edilen yönteme göre şekillenir ve bu tercih mobil uygulama geliştirme araçlarının seçimini de etkiler.

2.3.3.2 Tamamen Web Tarayıcıda Çalışan Uygulamalar için Yaşam Döngüsü

Bu uygulamaların yaşam döngüsü, standart bir Web sitesi geliştirme yaşam döngüsü ile paralellik gösterir ve buna ek olarak mobil ortamlar için sunulmuş ek yazılım kütüphanelerinin kullanımlarını da içerir.

HTML5 (HTML, CSS3, JavaScript) kullanılarak geliştirilen bu mobil uygulamalar mobil cihazlar için geliştirilen tarayıcılar tarafından yorumlanır ve kullanıcılara sunulur. Geliştirme ortamı olarak herhangi bir metin editörü kullanılabilir. Geliştirilen uygulamanın test edilmesi için ise güncel bir tarayıcı yeterli olacaktır.

2.3.3.3 Yerel (Native) Uygulamalar için Yaşam Döngüsü

Bu uygulamaların yaşam döngüsü, tamamen hedef platforma yönelik olarak uygulamanın geliştirmesi süreçlerini içerir. Seçilen hedef platforma (ör: Androd, iOS, Windows Phone) has geliştirme araçları, yazılım dili ve test ortamları kullanılır.

Hedef platformun tüm özelliklerini kullanabilmesi açısından önemli avantajlar sağlamasına rağmen her bir platforma özel olarak uygulamanın geliştirilmesi (kodlanması) gerekmektedir. Geliştirme ortamları çoğunlukla mobil cihazın işletim sistemine özel olarak geliştirilmiş ortamlardır ve test ortamları da yine işletim sistemine özel olarak geliştirilmiş emülatörlerden oluşur.

Bu ders kapsamında kullanılacak yerel geliştirme platformu Android olarak seçilmiştir. Bunun nedeni, Android platformunun geliştirme ortamı olarak kişisel bilgisayarlardaki işletim sistemleri konusunda esnek oluşudur. Mobil cihazlar için geliştirilecek bir Android uygulaması, Linux, Windows ve MacOS işletim sistemleri

kullanılarak geliştirilebilir. Bu durum maalesef iOS ve Windows Phone için geçerli değildir. Bu platformlarda yerel geliştirme yapılabilmesi ancak iOS için MacOS, Windows Phone için ise Windows işletim sisteminin kullanılması ile mümkündür.

2.3.3.4 Web ve Yerel Bileşenleri İçinde Barındıran Karma (hibrid) Uygulamalar için Yaşam Döngüsü

Bu uygulamaların yaşam döngüsü içerisinde hem Web tarayıcılar için geliştirilen uygulamaların yaşam döngüsü, hem de yerel uygulama geliştirme yaşam döngüsü yer alır. Bu yapının en büyük avantajı, geliştirme sürecinin temel Web teknikleri ile gerçekleştirilebilmesi ve aynı zamanda yerel özelliklerden de faydalanılabilmesidir. Bu aşamada Web geliştirme sürecine hakim olan bir kullanıcının ek bir programlama dili öğrenmesine gerek olmamaktadır. Bununla birlikte sağlanan ek bir avantaj da hedef platform için (ör: Android ya da iOS) yapılan bir kodlamanın bir diğer platform için de yeniden kodlama yapılmaksızın kullanılabilmesidir.

2.3.4. Mobil Uygulama Geliştirme

Bu ders kapsamında uygulama geliştirme bölümünde listelenen farklı mimariler ayrı ayrı kullanılarak mobil uygulama geliştirme süreci tüm adımları içerecek şekilde anlatılmıştır. Her bir uygulama geliştirme mimarisinde, kullanılan geliştirme araçlarının kurulumundan itibaren uygulamanın mobil cihaz üzerinde çalıştırılma sürecine kadar uygulama yaşam döngüsü detaylandırılmıştır.

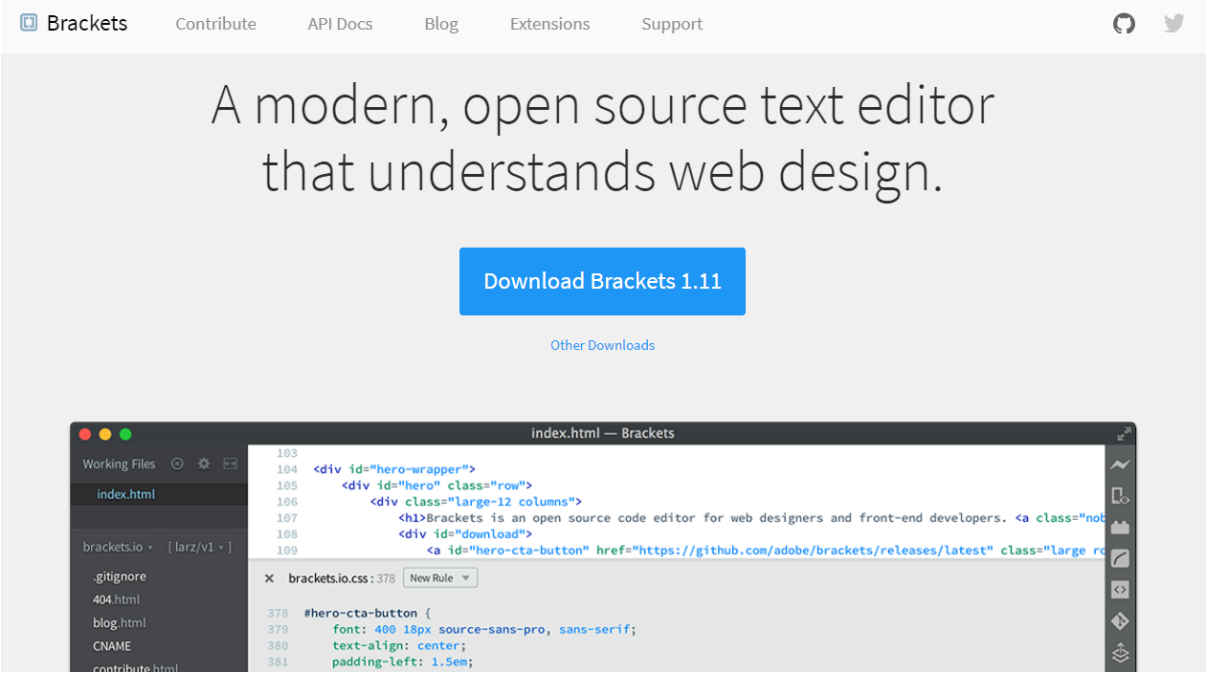
2.3.5. Web Tabanlı Mobil Uygulama Geliştirme

Mobil cihazlarda yaşanan teknolojik gelişmeler ve özellikle işlemci hızının artması, masaüstü bilgisayarlardaki İnternet kullanım deneyimine benzer bir deneyimin mobil cihazlarda da yaşanmasına olanak sağlamaktadır. Bu aşamada mobil cihazların özellikle ekran boyutları dikkate alınarak geliştirilen Web tabanlı uygulamalar bir mobil uygulama olarak çalışabilmekte ve mobil işletim sisteminden bağımsız olarak kullanılabilmektedir.

2.3.5.1 Web Tabanlı Mobil Uygulama Geliştirme Ortamı

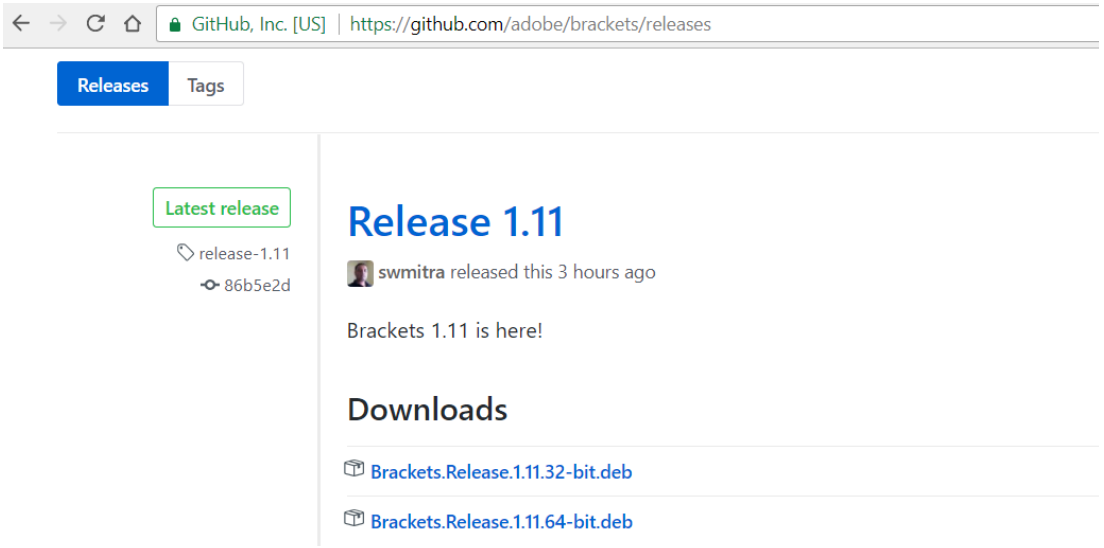
Web tabanlı mobil uygulama geliştirme ortamı olarak herhangi bir metin editörü kullanılabilir. Metin editörü seçiminde dikkat edilmesi gereken en önemli nokta editörün Web tabanlı uygulama geliştirme açısından verdiği desteklerdir (kod renklendirme, kod tamamlama vb. gibi.).

Bu ders kapsamında açık kaynak kodlu olan ve ücretsiz olarak sunulan Adobe firmasının Brackets isimli metin editörü kullanılacaktır. Bu editör http://brackets.io/ adresinden tüm popüler işletim sistemleri için indirilebilir ve kişisel bilgisayarlara kurulabilir (Şekil 1). Kurulum gerçekleştikten sonra arayüz Türkçe olarak kullanılabilir.



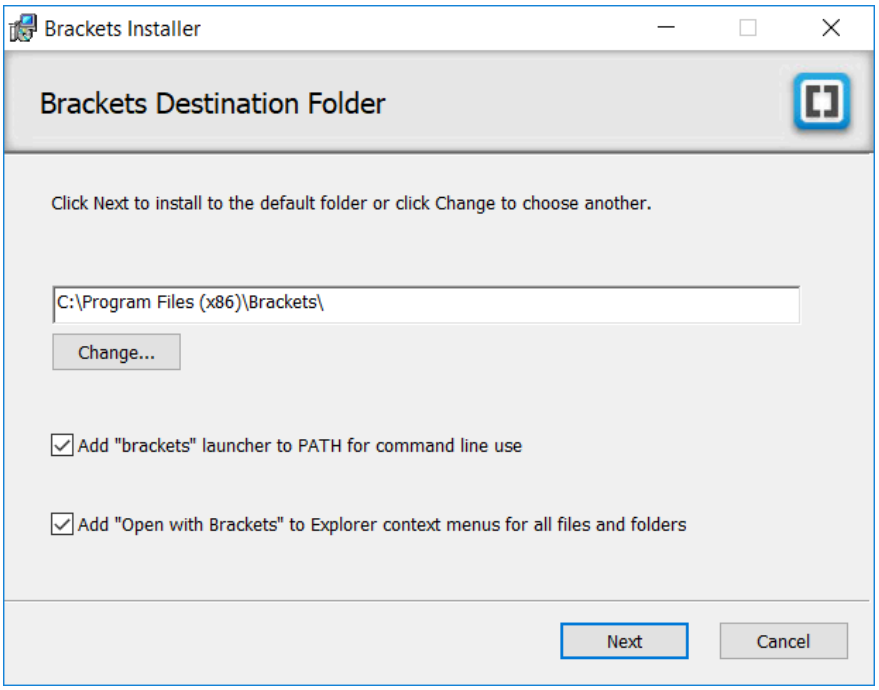
Şekil 1. Adobe Brackets Metin Editörü İnternet Sitesi (http://brackets.io/)

Brackets Web sitesindeki ana sayfada yer alan “Download Brackets” (Brackets’ı İndir) isimli butona tıklandığında kurulum dosyasının indirilme işlemi başlatılmış olur. O an hangi işletim sistemi kullanarak indirme işlemini gerçekleştiriliyorsa o işletim sistemine uygun kurulum dosyası indirilmeye başlanacaktır (Ör: Windows işletim sistemi için “Brackets.Release.1.11.msi”). Desteklenen diğer işletim sistemleri için sunulan kurulum dosyalarını ise “Other Downloads” (Diğer İndirmeler) bağlantısına tıklanarak görüntülenebilir (Şekil 2).



Şekil 2. Diğer İşletim Sistemleri için Kurulum Dosyaları

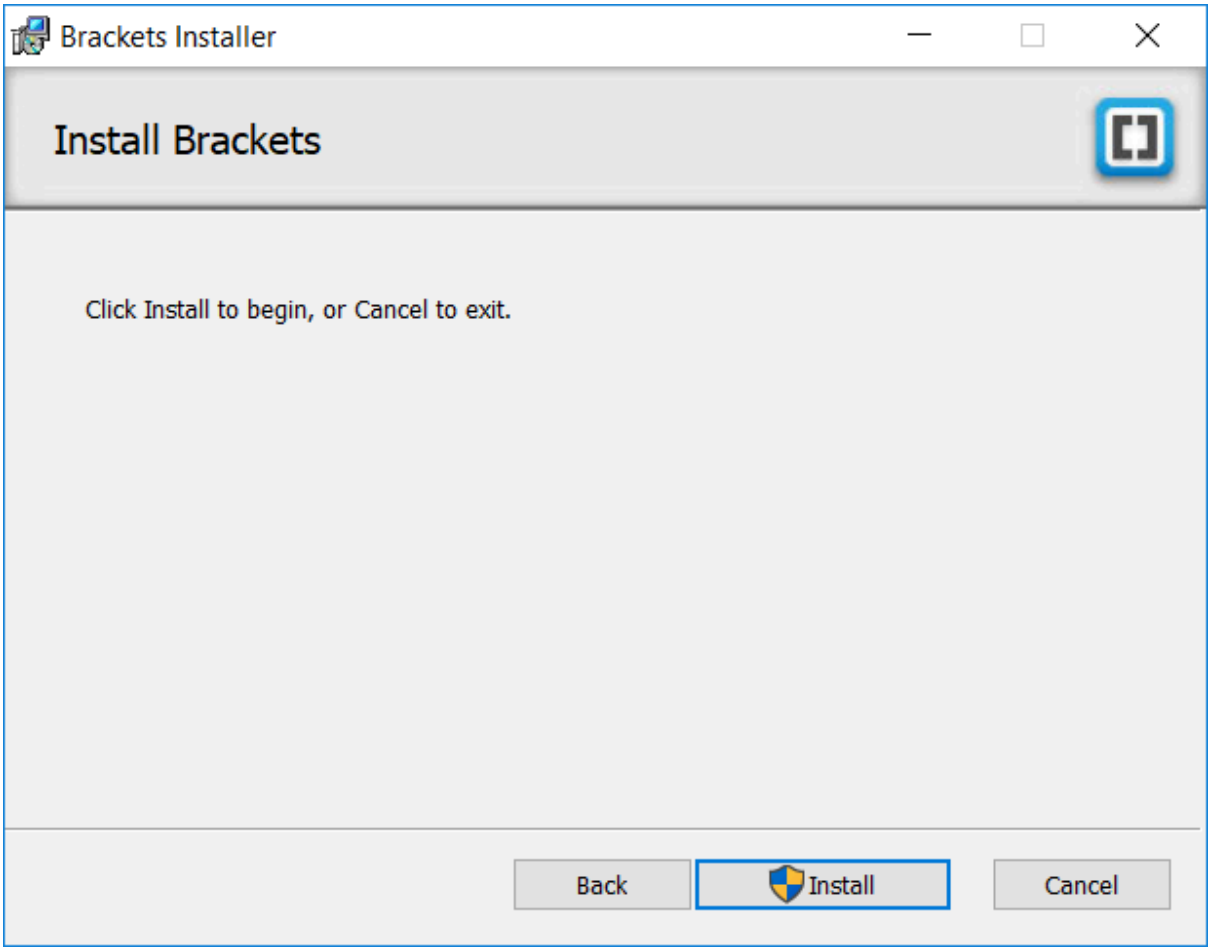
İnternet tarayıcısı aracılığıyla indirilmiş olan kurulum dosyasına “İndirilenler” adlı klasörden erişebilirsiniz. Bu klasör işletim sisteminize ve İnternet tarayıcınıza göre farklı isimlerde ve sabit diskiniz üzerinde farklı konumlarda olabilir. Kurulum dosyasına çift tıklandığında kurulum otomatik olarak başlatılacaktır (Şekil 3).



Şekil 3. Kurulum Yazılım

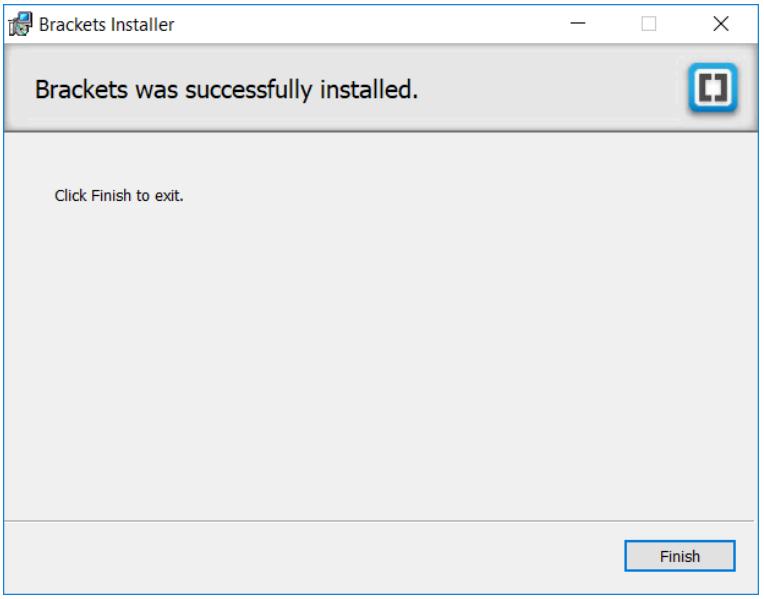
Kurulum yazılımındaki ayarlar standart olarak bırakılabilir. Sabit disk üzerinde farklı bir alana kurulumun gerçekleşmesi isteniyorsa bu bölümden yeni bir alan

tanımlaması yapılabilir. “Next” (İlerle) butonuna tıklayarak bir sonraki aşamaya geçilir. Bir sonraki aşamada “Install” (Kur) butonuna tıklayarak kurum gerçekleştirilir (Şekil 4).



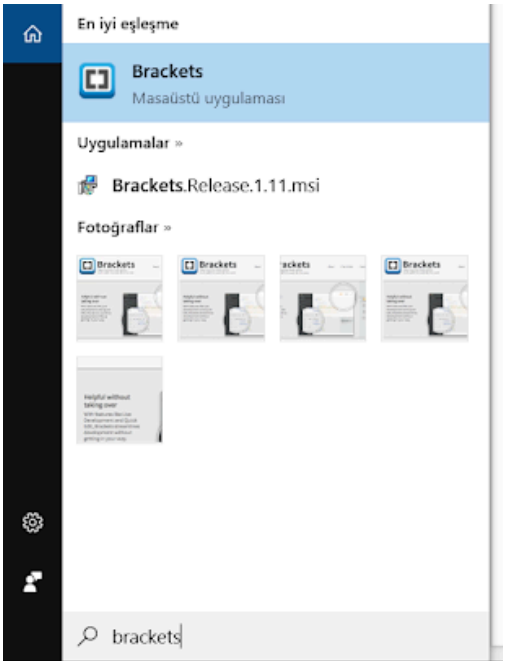
Şekil 4. “Install” (Kur).

Kurulum gerçekleştikten sonra Şekil 5’de yer alan programın sorunsuz kurulduğu ile ilgili bilgi ekranı görüntüleniyorsa kurulum tamamlanmış demektir. “Finish” (Tamamlandı) butonuna tıklayarak kurulum programından çıkılabilir. Brackets programı artık kullanılabilir durumdadır.



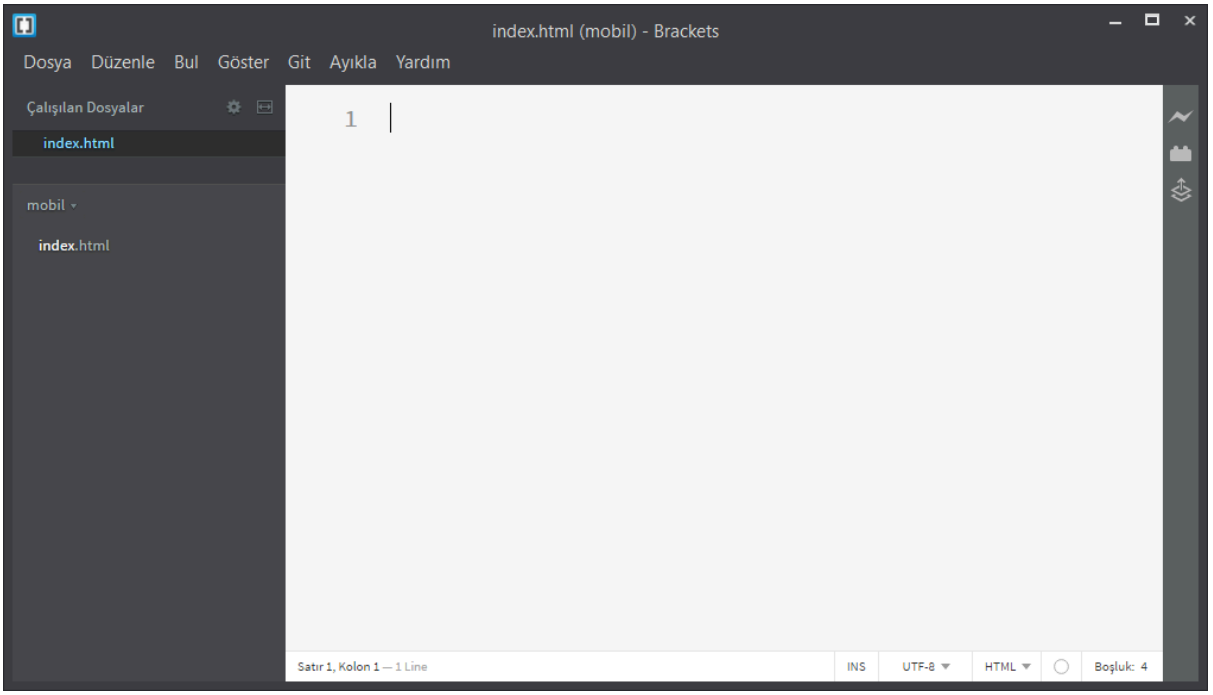
Şekil 5. Kurulumun Tamamlandığını Belirten Bilgi Ekranı

Kurulum işlemi tamamlanan Brackets yazılımına bilgisayardaki diğer programların da yer aldığı liste içerisinden erişilebilir. Yazılımı başlatmak için bilgisayar içerisinde "Brackets" kelimesi aratılarak da yazılıma erişilebilir (Şekil 6).



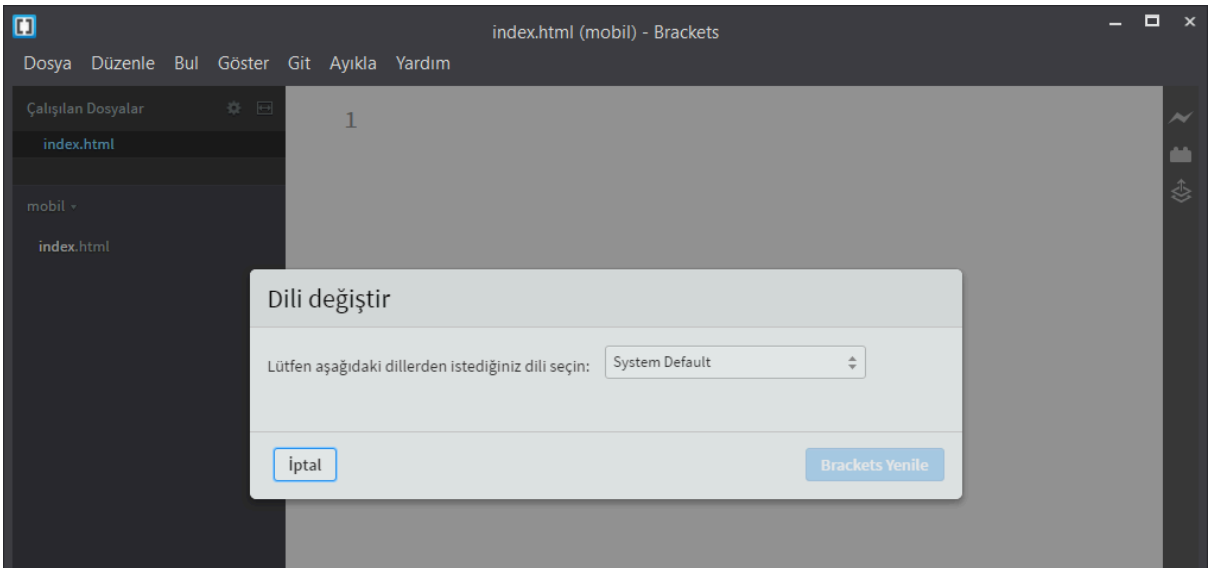
Şekil 6. Brackets Yazılımının Aratılması ve Aram sonucu

Liste içerisinden ya da arama sonuçlarından Brackets ikonuna tıklandığında yazılım çalışacak ve Şekil 7'deki arayüz görüntülenecektir.



Şekil 7. Brackets Yazılımının Arayüzü

Brackets, kurulu olduğu işletim sisteminin dil ayarlarını algılayarak otomatik olarak arayüzünü o dile göre yapılandırır. Arayüzün dil seçimi isteğe göre menüler yardımıyla da yapılabilir. Arayüzün dilini değiştirmek için “Ayıkla” menü grubundaki “Dili Değiştir” bölümü kullanılır (Şekil 8).



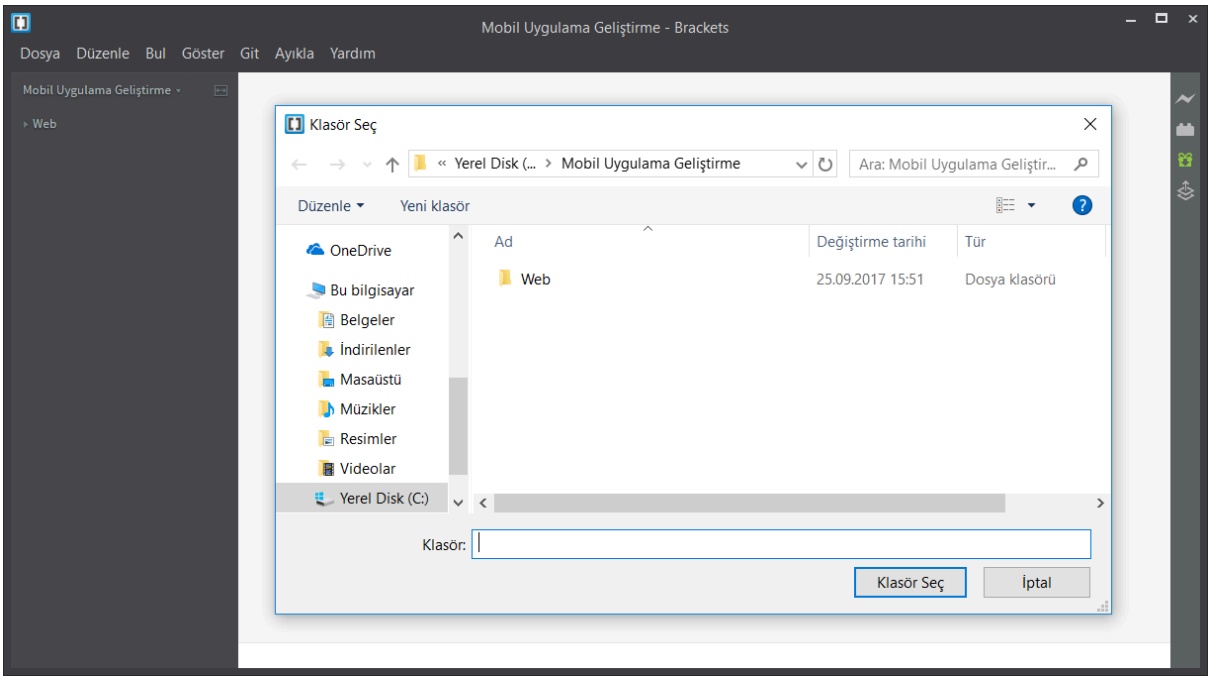
Şekil 8. Arayüzde Dil Değişikliği Yapılması.

Brackets, temel olarak iki ana bölümden oluşmaktadır. Bu bölümler solda yer alan, klasör ve dosya görüntüleme alanı ve sağda yer alan, kod yazma ve görüntüleme

alanıdır. Mobil uygulama geliştirmeye başlamadan önce bilgisayardaki diğer dosya ve klasörlerle uygulama dosyalarının karışmasını engellemek için bu çalışmaya özel bir klasör oluşturmak ve o klasörün içinde çalışmak uygun olacaktır.

Bu ders kapsamında geliştirilecek uygulamaların depolanması için sabit diskte “Mobil Uygulama Geliştirme” adlı ana bir klasör ve bu klasörün içerisinde de uygulama geliştirme mimarilerine bağlı olarak alt klasörler oluşturulmuştur. Web tabanlı uygulamaların yer alacağı klasörün adı ”Web”’dir.

Brackets açıldıktan sonra yapılması gereken ilk şey çalışma alanı olarak oluşturulmuş klasörün seçilmesidir. Bu seçim işlemi “Dosya” menü grubu altındaki “Klasörü Aç” (Ctrl+Alt+O) öğesine tıklayarak gerçekleştirilir (Şekil 9). Klasör seçimi bir kere gerçekleştirildikten sonra Brackets her açılışında aynı klasörün içeriğini gösterir. Brackets’ı her açışta yeniden bu işlemin gerçekleştirilmesine gerek kalmaz.



Şekil 9. Çalışmaların Yapılacağı Klasörün Brackets İçerisinde Tanımlanması

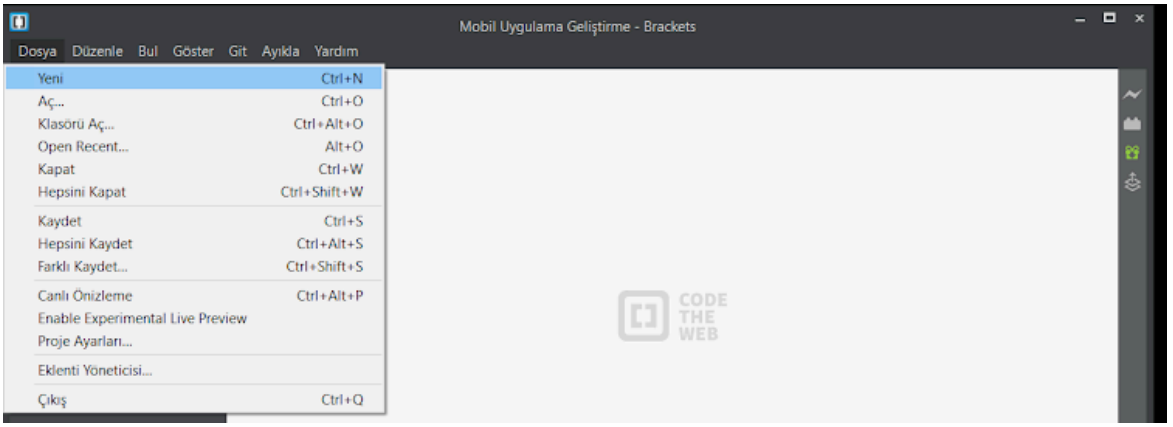
Klasör seçimi ile birlikte Web tabanlı mobil uygulama geliştirme ortamının hazırlık aşaması da tamamlanmış olur. Bu aşamadan sonra artık uygulama geliştirmeye başlanabilir.

2.3.5.2 Web Tabanlı Örnek Bir Mobil Uygulama Geliştirme

Web tabanlı mobil uygulama geliştirmeye en iyi başlangıç bu iş için özel olarak geliştirilmiş yazılım kütüphanelerinin kullanılması ile olabilir. HTML5, CSS3 ve JavaScript'in birlikte kullanımı ile oluşturulmuş birçok farklı yazılım kütüphanesi bulunmasına rağmen en fazla kullanılan, dokümantasyonu ve örnekleri en iyi hazırlanmış kütüphanelerden biri JQuery Mobile adlı yazılım kütüphanesidir. Jquery Mobile ile ilgili detaylı bilgi, dokümantasyon ve örneklere https://jquerymobile.com/ adresinden erişilebilir.

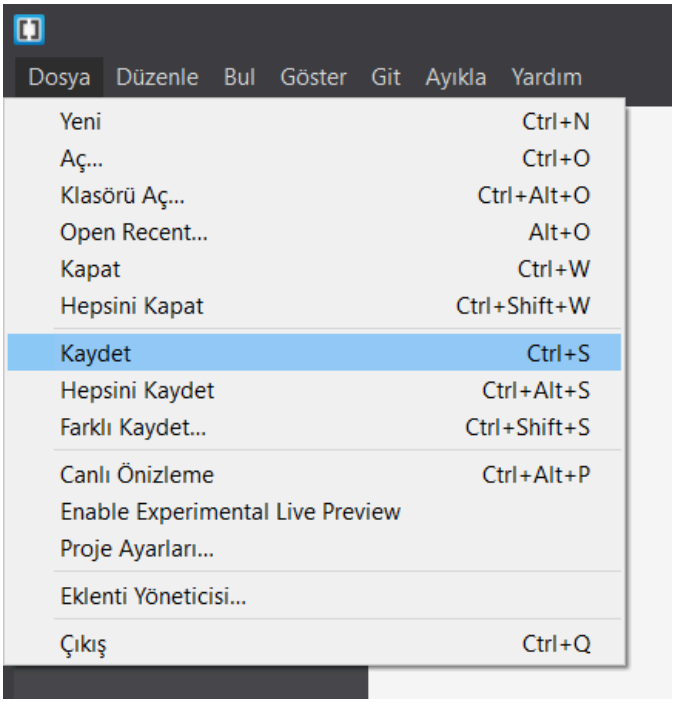
Örnek mobil uygulamamızda Jquery Mobile kütüphanelerini kullanarak basit bir sözlük uygulaması gerçekleştireceğiz. Bu sözlük uygulaması kelime listesi ve bu kelime listesi içerisinde bir arama bölümü içerecek şekilde tasarlanmış olacak. Şimdi adım adım bu uygulamayı geliştirelim.

Öncelikle kelime listesinin ve arama formunun yer alacağı ana sayfayı oluşturalım. Web tabanlı uygulamalarda aynen Web tasarımında olduğu gibi ana sayfanın adının "index.html" olarak belirlenmesi gerekmektedir. Bu şekilde bir klasör içerisinde duran uygulamayı yorumlayacak olan mobil İnternet tarayıcı hangi dosyayı başlangıç dosyası olarak kullanacağını belirleyebilecektir. Brackets içerisinde "index.html" isimli dosyayı "Dosya" menü grubu altındaki "Yeni" (Ctrl+N) öğesine tıklayarak oluşturabilirsiniz (Şekil 10).



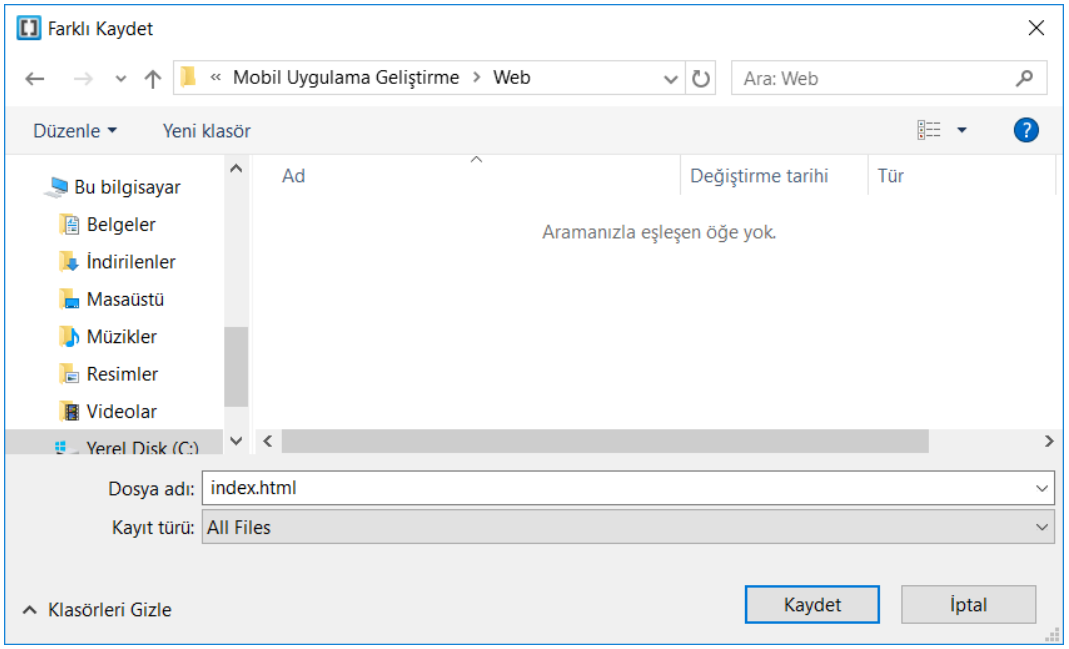
Şekil 10. Yeni bir dosya oluşturulması.

Yeni bir dosya oluşturduğunuzda bu dosya “Adsız” adıyla oluşturulur. Oluşturulan bu dosyayı “Dosya” menü grubu altındaki “Kaydet” (Ctrl+S) öğesine tıklayarak “index.html” adıyla içinde çalıştığınız klasörün altına kaydediniz (Şekil 11).



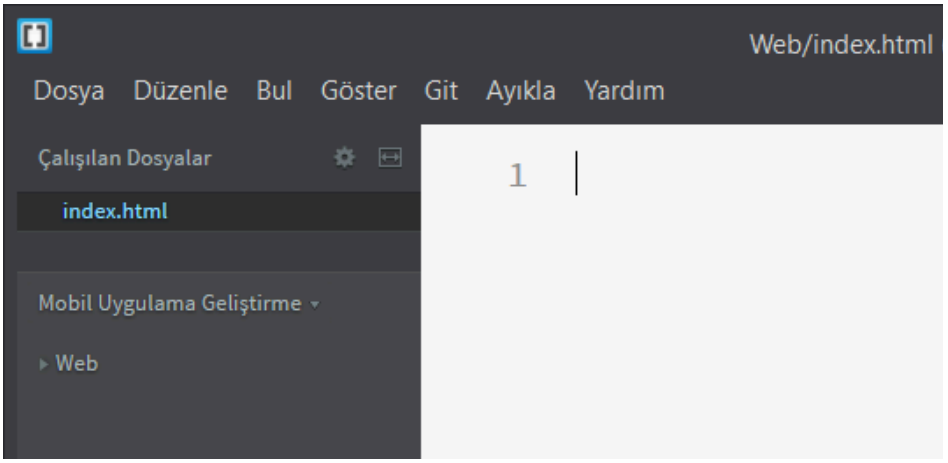
Şekil 11. Oluşturulan dosyanın kaydedilmesi

Bu örnek uygulamada çalışmalar “Mobil Uygulama Geliştirme” ana klasörü içerisindeki “Web” adlı alt klasöre kaydedilmektedir (Şekil 12).



Şekil 12. Dosya kaydedilirken klasör seçimi ve dosya isimlendirilmesi.

Dosyanın oluşturulması ve isimlendirme yapılarak kayıt işleminin tamamlanmasının ardından kod yazımı ile ilgili bölüme geçiş yapılabilir. Öncelikle Brackets'ın sol bölümünde yer alan dosya bölümünde index.html adlı dosyanın seçili olduğunu kontrol edin. Brackets'ın üst bölümünde şu an düzenleme için açık olan dosyanın ismi yazmalıdır. Aynı zamanda yine sol üst bölümde "Çalışılan Dosyalar" adı altında index.html adlı dosyanın listelenmesi gerekmektedir (Şekil 13).



Şekil 13. "index.html" adlı dosyanın düzenlemeye hazır hale getirilmesi.

Şimdi “index.html” içerisinde ihtiyacımız olan kodları yazmaya başlayalım. Aşağıdaki kod bloğu standart olarak HTML kodlamada kullanılan kod bloğudur (Şekil 14).

```
1   <!DOCTYPE html>
2   <html>
3   <head>
4       <title>Page Title</title>
5   </head>
6
7   <body>
8
9   </body>
10  </html>
```

Şekil 14. Standart HTML Kod Bloğu

Bu kod bloğunda 1. satırda dokümanın tipi tanımlanmaktadır. 2. satırdan 10. satıra kadar yer alan bölüm HTML olarak tanımlanmıştır. HTML içerisindeki tanımlamalar etiket adı verilen yapıların açılıp kapatılması ile gerçekleştirilir. Örnek uygulamamızın 2. satırında <html> ifadesi ile HTML etiketi açılmakta ve 10. satırda </html> ifadesi ile HTML etiketi kapatılmaktadır. Aynı etiketin açılıp kapatılması arasında kalan tüm kodlar o etiketin genel özellikleri ile tanımlanmış olur. Yine örnek uygulamamızdan yola çıkarak HTML dokümanının üst bölümü olarak tanımlanmak istenen alanda HEAD etiketi açılıp kapatılmıştır. <head> ile </head> arasında kalan tüm kodlar HEAD (başlık) etiketinin özelliğine sahip olacak şekilde tanımlanmıştır.

Bir HTML dokümanında iki ana bölüm bulunmaktadır. Bu bölümler Başlık (Head) bölümü ve Gövde (Body) bölümüdür. Başlık bölümünde İnternet tarayıcısına özel bir takım tanımlamalar yer alır. Bu tanımlamalar, sayfanın başlığı, metin kodlaması, kullanılacak biçimlendirme dosyalarının (CSS) ve JavaScript kütüphanelerinin adresleri ve bir takım üst veri alanlarıdır.

Örnek uygulamamızda başlık (<head>) etiketi içerisinde aşağıdaki tanımlamaların yapılması gerekmektedir. Bu tanımlamalar JQuery Mobile biçimlendirme dosyalarının ve kod kütüphanelerinin adreslerini içermektedir (Şekil 15).

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Örnek Mobil Uygulama</title>
5      <meta charset="utf-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1">
7      <link rel="stylesheet" href="http://code.jquery.com/mobile/1.4.5/jquery.mobile-
       1.4.5.min.css" />
8      <script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
9      <script src="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js">
       </script>
10 </head>
11
12 <body>
13
14 </body>
15 </html>
```

Şekil 15. Başlık bölümü tanımlamaları.

Başlık bölümü içerisindeki kod satırlarını detaylandırmamız gerekirse;

- 4. satır uygulamanın başlığını tanımlamaktadır ve mobil İnternet tarayıcının üst bölümünde uygulamanın adının yer almasını sağlayacaktır.
- 5. satır uluslararası dil kodlaması olarak belirlenmiş olan UTF-8'in sayfanın dil kodu olarak belirlenmesini sağlar. Bu satır özellikle latin alfabesi dışındaki harflerin kullanımı açısından önemlidir.
- 6. satır mobil İnternet tarayıcısına hazırlanan sayfanın boyutları ile ilgili bir düzenleme yapmasını söyler. Bu şekilde boyutlandırma açısından ideal bir oran yakalanmaya çalışılır.
- 7. satır JQuery Mobile biçimlendirme dosyasının adresini içerir. Bu sayede belirli bir temaya göre yazı biçimleri (fontlar), boyutları, renkler, form ve diğer HTML bileşenlerinin biçimsel özellikleri belirlenmiş olur.
- 8. satır JQuery'nin ana kütüphanesinin adresini içermektedir. Burada bu çalışma çerçevesi (framework) için kullanılan JavaScript kütüphanesi bulunmaktadır.
- 9. satır JQuery'nin mobil cihazlar için geliştirdiği JavaScript kütüphanesinin adresini içerir.

Başlık bölümü tamamlandığına ve gerekli tanımlar yapıldığına göre artık uygulamanın kullanıcılar tarafından görüntülenecek kısmı olan “gövde” (body) üzerinde çalışmalara başlanabilir. Uygulamamızın gövde kısmında bir arama formu ve bir kelime listesi için gerekli kodlar yer alacaktır. Bu kodlar şekil 15’de gösterilmektedir.

```
12 ▼ <body>
13 ▼     <form>
14           <input id="filter-for-listview" data-type="search" placeholder="Arama
             yapmak için birkaç harf giriniz...">
15       </form>
16 ▼     <ul data-role="listview" data-inset="true" data-filter="true" data-
         input="#filter-for-listview">
17           <li><a href="#">Armut</a></li>
18           <li><a href="#">Elma</a></li>
19           <li><a href="#">Muz</a></li>
20           <li><a href="#">Şeftali</a></li>
21           <li><a href="#">Üzüm</a></li>
22       </ul>
23   </body>
```

Şekil 15. Uygulamanın gövde bölümünde yer alan kodlar.

Gövde bölümü içerisindeki kod satırlarını detaylandırmamız gerekirse;

- 13. ve 15. satırlar arasında yer alan kodlar bir HTML formunu tanımlamaktadır. Bu form JQuery Mobile JavaScript kütüphanesi içerisindeki “filter-for-listview” adlı fonksiyonu da içinde barındıracak şekilde yapılandırılmıştır. Bu fonksiyon listeyi süzmemize imkan sağlayacak ve bu sayede basit bir arama işlemi gerçekleştirilebilecektir. Arama kutucuğu içerisinde yer alan varsayılan mesajı da yine bu bölümdeki “placeholder” değişkenine atanan değer ile değiştirmek mümkündür.
- 16. ve 22. satırlar arasında yer alan kodlar, kelimelerin listelenmesini sağlamaktadır. Listelenen kelimelere bir Web sayfası ya da Web sitesi adresi atamak da mümkündür. Böylece tıklanan kelime ile ilgili bir detay sayfasına ya da bir başka siteye yönlendirme yapılabilir.

Uygulamamızın temel özelliklerini tamamladığımıza göre ilk test işlemini gerçekleştirebiliriz. Bu aşamada uygulama içerisinde kullanılan CSS ve JavaScript kütüphanelerini İnternet üzerinden kullandığımız için test işlemini gerçekleştireceğimiz cihazımızda İnternet bağlantısının olması gerekmektedir.

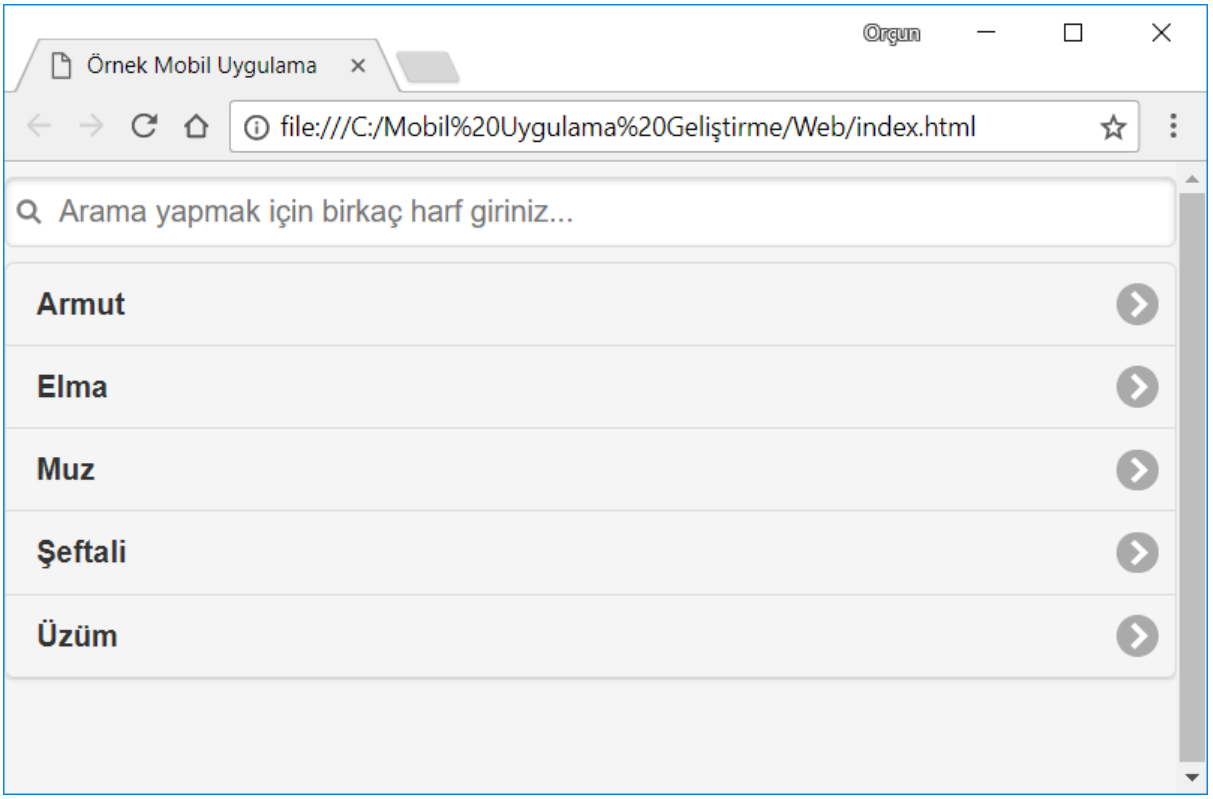
Uygulamanın İnternet bağlantısına ihtiyaç duymasını istemiyorsak CSS ve JavaScript kütüphanelerini JQuery Mobile'ın İnternet sitesinden (http://jquerymobile.com/download/) bilgisayarımıza indirip proje klasörü içerisine kaydetmeliyiz. Daha sonra kaydettiğimiz bu dosyalara HTML sayfası içerisinden bağlantı sağlamalıyız.

Test işlemi için öncelikle kendi kişisel bilgisayarımızda yer alan herhangi bir İnternet tarayıcısını kullanabiliriz. Daha sonra uygulamamızı istersek dosyaları İnternet üzerine yüklediğimiz bir lokasyondan istersek de uygulama dosyalarını direkt mobil cihazın içerisine yükleyerek test edebiliriz.

2.3.5.3 Mobil Uygulamanın Kişisel Bilgisayar Üzerinde Yer Alan İnternet Tarayıcı ile Test Edilmesi

Test işlemini kişisel bilgisayar üzerinde gerçekleştirmek için takip edebileceğimiz iki yol bulunmaktadır. Bunlardan birincisi eğer bilgisayarımızda Google Chrome tarayıcı yüklü ise Brackets içerisindeki canlı önizleme (Live Preview) özelliğini kullanmak, diğeri ise direkt olarak bilgisayarımızın "Dosya Gezgini" yardımı ile uygulamamızın başlangıç dosyası olan "index.html" dosyasına çift tıklamak. Bu iki yol da İnternet tarayıcıda uygulamamızın açılmasını ve test edebilmemizi sağlayacaktır.

"Dosya Gezgini" ile proje klasörüne girip "index.html" dosyasına çift tıkladığımızda açılan uygulamamızın görüntüsü şekil 16'daki gibi olmalıdır.

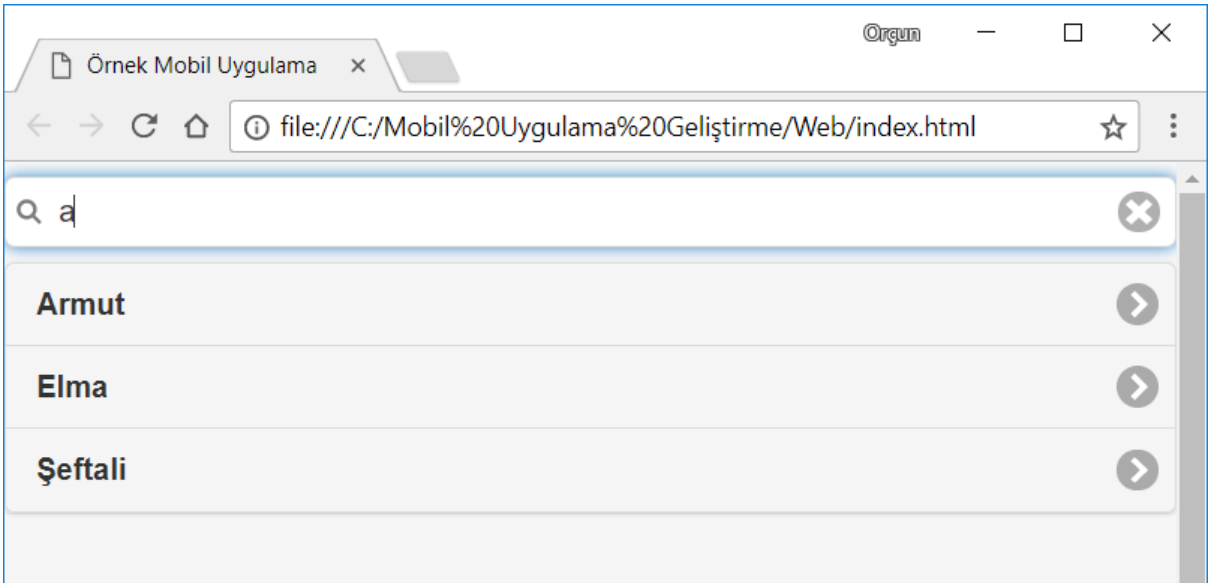


Şekil 16. Mobil uygulamanın İnternet tarayıcısındaki görünümü.

Bilgisayarınızda varsayılan İnternet tarayıcınızın hangisi olduğuna bağlı olarak uygulamanız ilgili tarayıcıda görüntülenir.

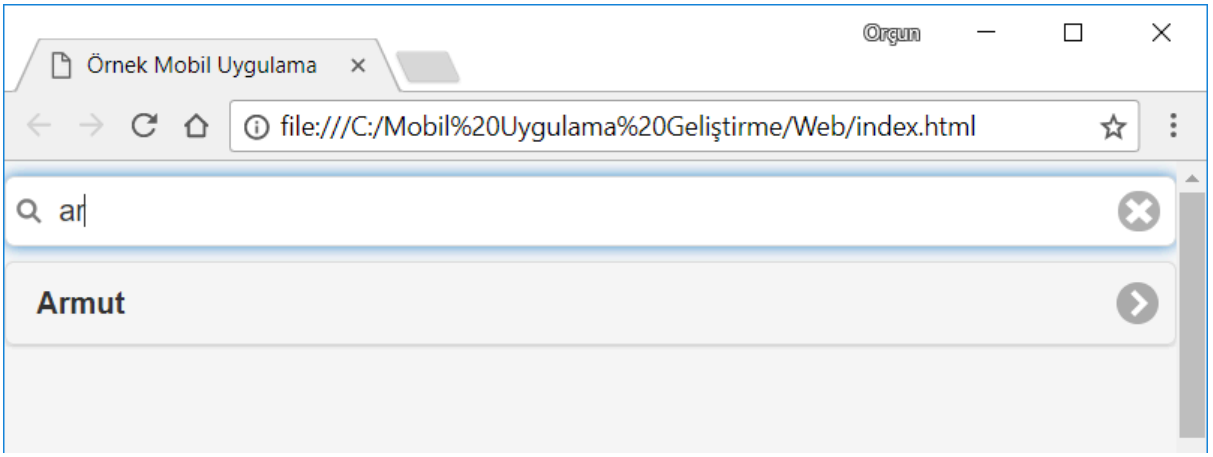
Farklı İnternet tarayıcılarda uygulamanızı test etmek için “Dosya Gezgini”’inde “index.html” dosyasının üzerine sağ klik yaparak “Birlikte aç” bölümünden farklı bir İnternet tarayıcı seçebilirsiniz.

Şimdi uygulamamızın arama bölümünü test edelim. Örneğin arama kutucuğuna “a” harfi yazdığımızda içinde “a” harfi bulunan kelimeler haricindeki kelimeler listeden çıkartılacaktır (Şekil 17).



Şekil 17. Arama kutucuğuna harf girişi ile süzme işleminin gerçekleştirilmesi.

Uygulama ekranında görüldüğü gibi “a” harfi girildikten sonra kelime listesinde yer alan “Üzüm” ve “Muz” içlerinde “a” harfini barındırmadığı için listeden çıkartıldılar. Şimdi “a” harfine bir de “r” harfi ekleyerek devam edelim. Bu durumda listede sadece “Armut”’un kaldığını göreceksiniz (Şekil 18).



Şekil 18. İkinci harfin eklenmesi ile listenin yeniden oluşması.

Örnek uygulamamızı 5 kelimeden oluşacak şekilde hazırlamıştık. Uygulamaya yeni kelimeler eklemek için Şekil 19’da işaretli satırı (ör. 18. satır) çoğaltarak yeni kelimelere göre düzenlememiz yeterli olacaktır (Şekil 20).

```
<body>
    <form>
        <input id="filter-for-listview" data-type="search" placeholder="Arama
        yapmak için birkaç harf giriniz...">
    </form>
    <ul data-role="listview" data-inset="true" data-filter="true" data-
    input="#filter-for-listview">
        <li><a href="#">Armut</a></li>
        <li><a href="#">Elma</a></li>
        <li><a href="#">Muz</a></li>
        <li><a href="#">Şeftali</a></li>
        <li><a href="#">Üzüm</a></li>
    </ul>
</body>
```

Şekil 19. Uygulamaya yeni kelimelerin eklenmesi.

```
<body>
    <form>
        <input id="filter-for-listview" data-type="search" placeholder="Arama
        yapmak için birkaç harf giriniz...">
    </form>
    <ul data-role="listview" data-inset="true" data-filter="true" data-
    input="#filter-for-listview">
        <li><a href="#">Armut</a></li>
        <li><a href="#">Elma</a></li>
        <li><a href="#">Erik</a></li>
        <li><a href="#">Muz</a></li>
        <li><a href="#">Şeftali</a></li>
        <li><a href="#">Üzüm</a></li>
    </ul>
</body>
```

Şekil 20. Yeni kelime olarak “Erik”’in listeye eklenmesi.

Artık uygulamamızda 6 sözcük yer almakta. İstenilen sayıda sözcüğü bu şekilde eklenip çıkartılması bu sayede mümkün olmaktadır.

Uygulamamızdaki bir diğer özellik ise seçilen kelime ile ilgili detaylı bilgi verilebilecek bir sayfaya ya da bir harici Web sitesine bağlantı sağlanabilmesidir. Bu bağlantıyı sağlayabilmek için kod bölümünde ilgili kelimenin bulunduğu satırda bir düzenleme yapmamız gerekmektedir. Bu düzenleme HTML içerisinde bağlantı vermek için kullanılan “<a>” etiketinin içindeki “href” tanımlamasında yapılmalıdır.

Örnek uygulamamızda ilk önce “Armut” kelimesi üzerinde bir düzenleme yapalım. Kullanıcı “Armut” kelimesini tıkladığı zaman Türk Dil Kurumu’nun (TDK) Güncel Türkçe Sözlüğü’ndeki armut kelimesinin arama sonucuna yönlendirilmiş olsun. Bu

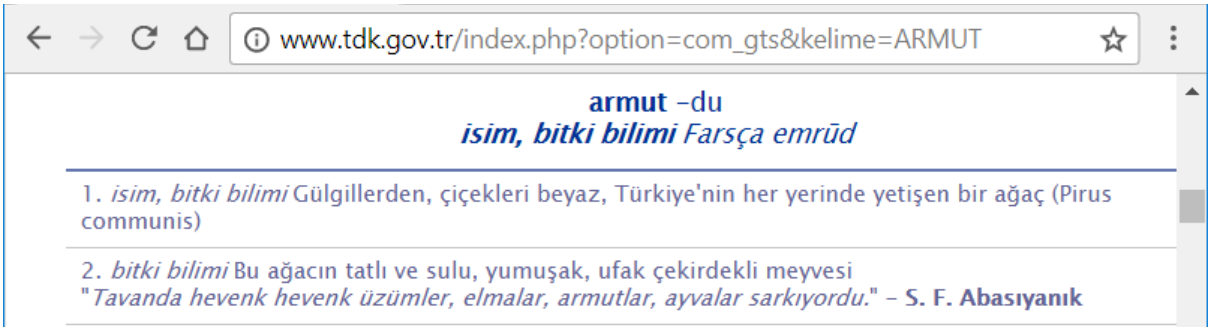
arama sonucunu elde edebilmek için http://www.tdk.gov.tr/ adresini ziyaret edilmeli ve buradaki sözlükte “Armut“ kelimesi aratılmalıdır.

Arama sonucunda gelen sayfanın adresi İnternet tarayıcısından kopyalanarak mobil uygulamamızın kod bölümünde ilgili yere yapıştırılır (Şekil 21).

```
<body>
    <form>
        <input id="filter-for-listview" data-type="search" placeholder="Arama
        yapmak için birkaç harf giriniz...">
    </form>
    <ul data-role="listview" data-inset="true" data-filter="true" data-
    input="#filter-for-listview">
        <li><a href="http://www.tdk.gov.tr/index.php?
        option=com_gts&kelime=ARMUT">Armut</a></li>
        <li><a href="#">Elma</a></li>
        <li><a href="#">Erik</a></li>
        <li><a href="#">Muz</a></li>
        <li><a href="#">Şeftali</a></li>
        <li><a href="#">Üzüm</a></li>
    </ul>
</body>
```

Şekil 21. Yönlendirme yapılacak harici Web sitesinin adresinin kod içine yerleştirilmesi.

Yönlendirme yapılacak adres kod bölümüne eklendikten sonra uygulama test edildiğinde armut üzerine tıklandığında TDK’nın ilgili sayfasına ulaşıldığı görülür (Şekil 22).



Şekil 22. Yönledirme yapılan sayfanın İnternet tarayıcısında görüntülenmesi.

2.3.5.4 Mobil Uygulamanın Mobil Cihaz Üzerinde Yer Alan İnternet Tarayıcı ile Test Edilmesi

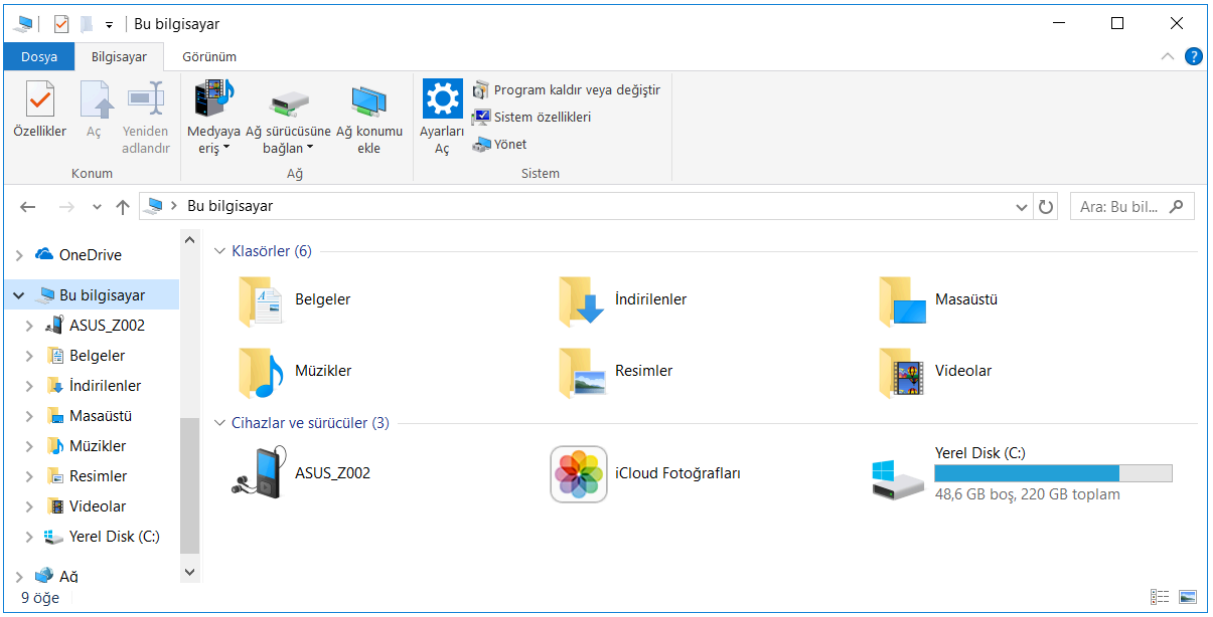
Önceki bölümde bilgisayarımızın içindeki İnternet tarayıcı ile yapmış olduğumuz testi şimdi mobil cihazımızın içinde yer alan İnternet tarayıcı ile yapalım. Bunu gerçekleştirirken iki farklı yol izleyebiliriz. Bu yollardan ilki mobil uygulamamızı oluşturan dosyaları İnternet üzerindeki bir lokasyona aktarmak ve mobil İnternet tarayıcı ile bu lokasyonu ziyaret ederek mobil uygulamayı görüntülemek. İkinci yol ise uygulamayı oluşturan dosyaları mobil cihazımızın depolama alanına kopyalamak ve buradan mobil İnternet tarayıcı yardımı ile uygulamayı görüntülemek.

Yukarıda belirtilen ilk yol için bir İnternet sitesine, bu İnternet sitesine erişim haklarına ve dosyaları aktarabileceğimiz bir yazılıma (FTP yazılımı) ihtiyacımız var. Bunların temin edilmesinde sorunlar olabileceğinden yola çıkarak uygulamamızı mobil cihaz üzerinde ikinci yolu kullanarak test edeceğiz.

Öncelikle mobil cihazımızın şarj kablosunu cihazımızın şarj çıkışına ve kişisel bilgisayarımızın USB girişine bağlayarak fiziksel bağlantısını kuruyoruz. Bu aşamada mobil cihazın açık konumda olması gerekiyor, aksi halde bilgisayarın mobil cihazı algılaması mümkün olmaz.

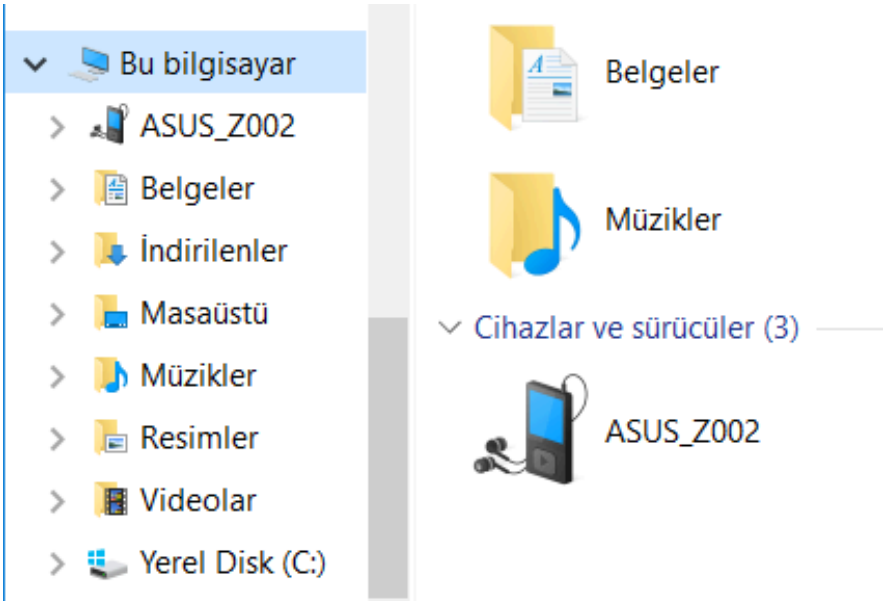
Fiziksel bağlantı kurulduktan sonra kimi mobil cihazlar bağlantının sağlanabilmesi için bir onay ekranı görüntülerler, bu ekrandan onay verilmesi ile birlikte bağlantı tam olarak sağlanır. Bazı durumlarda ise mobil cihazın ekran kilidinin açılması gerekmektedir.

Gerekli izinler verildikten sonra bilgisayarınızın Dosya Gezgini uygulaması aracılığıyla mobil cihazınızı görüntüleyebiliyor olmanız gerekir (Şekil 23).



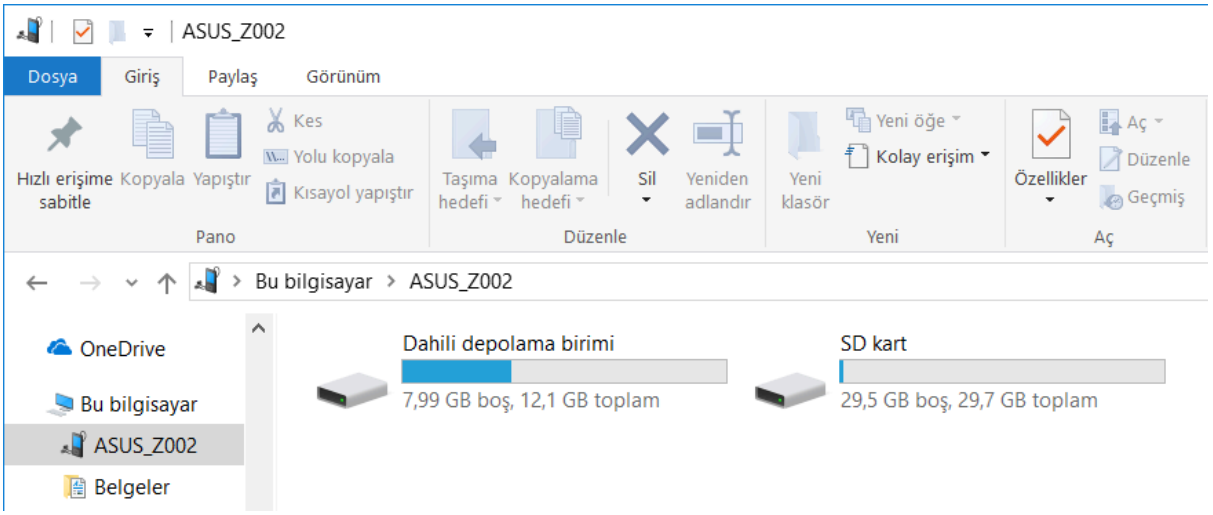
Şekil 23. Dosya Gezgini aracılığıyla mobil cihazın görüntülenmesi.

Dosya Gezgini içerisinde mobil cihazınızın varsayılan olarak görüntüleneceği yer “Bu bilgisayar” adlı bilgisayarınızın sabit disklerinin de görüntülendiği bölümdür. Mobil cihazınız marka ve model adı ile “Cihazlar ve sürücüler” altında görüntüleniyor olmalıdır (Şekil 24).



Şekil 24. Mobil cihazın marka ve model adı ile görüntülenmesi.

Bu aşamada mobil cihazınızı temsil eden ikona çift tıklayarak mobil cihazınızın depolama alanları görüntülenir. Mobil cihazınızda tek bir depolama alanı olabileceği gibi birden çok dahili depolama alanı ve / veya dışarıdan eklenebilen (ör: SD kart vb. gibi) depolama alanları olabilir (Şekil 25).

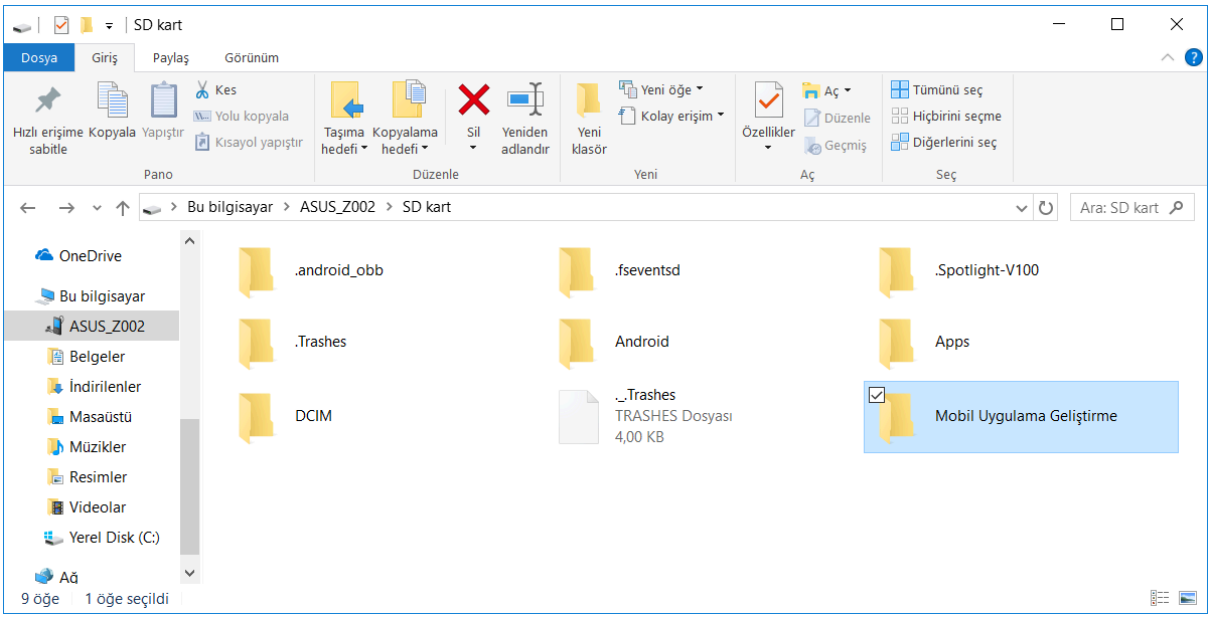


Şekil 25. Mobil cihazın depolama alanları.

Şekil 25'de görüldüğü gibi test cihazımızda bir dahili ve bir de SD kart olmak üzere 2 farklı depolama alanı bulunmakta. Bu testimizde uygulama dosyalarını kopyalamak için SD kart alanını kullanacağız. Bunun için SD kartı temsil eden ikona çift tıklıyoruz.

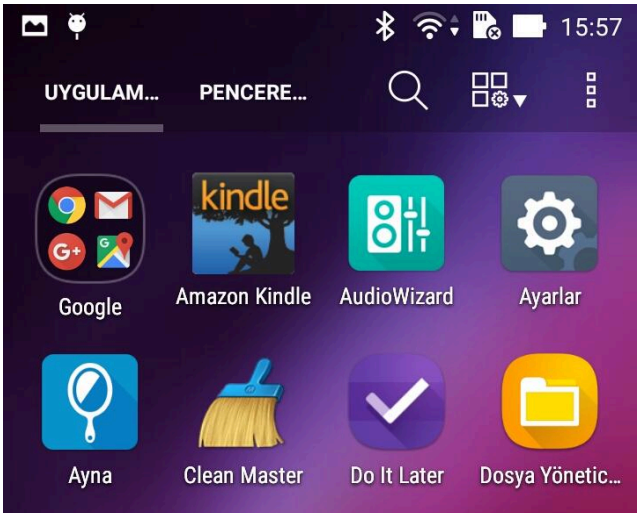
SD kart cihazımızın özelliklerine göre farklı dosyalar ve klasörler içerebilir. Bu dosya ve klasörler arasında mobil uygulamamızı rahatça bulabilmek için tüm çalışmalarımızı kaydettiğimiz ana klasör olan "Mobil Uygulama Geliştirme" adlı klasörü bilgisayarımızın sabit diskinde yer alan lokasyonundan kopyalayarak SD kart içerisine yapıştırmamız yeterli olacaktır. Kopyalama ve yapıştırma işlemi tamamlandığında mobil cihazımızın SD kart bölümünün görünümü Şekil 26'daki gibi olmalıdır.

Geliştirdiğimiz uygulamanın klasör ve dosyaları dahili depolama alanına da yerleştirilebilir, burada dahili depolama alanı ve SD kart depolama alanı arasında bir fark bulunmamaktadır.



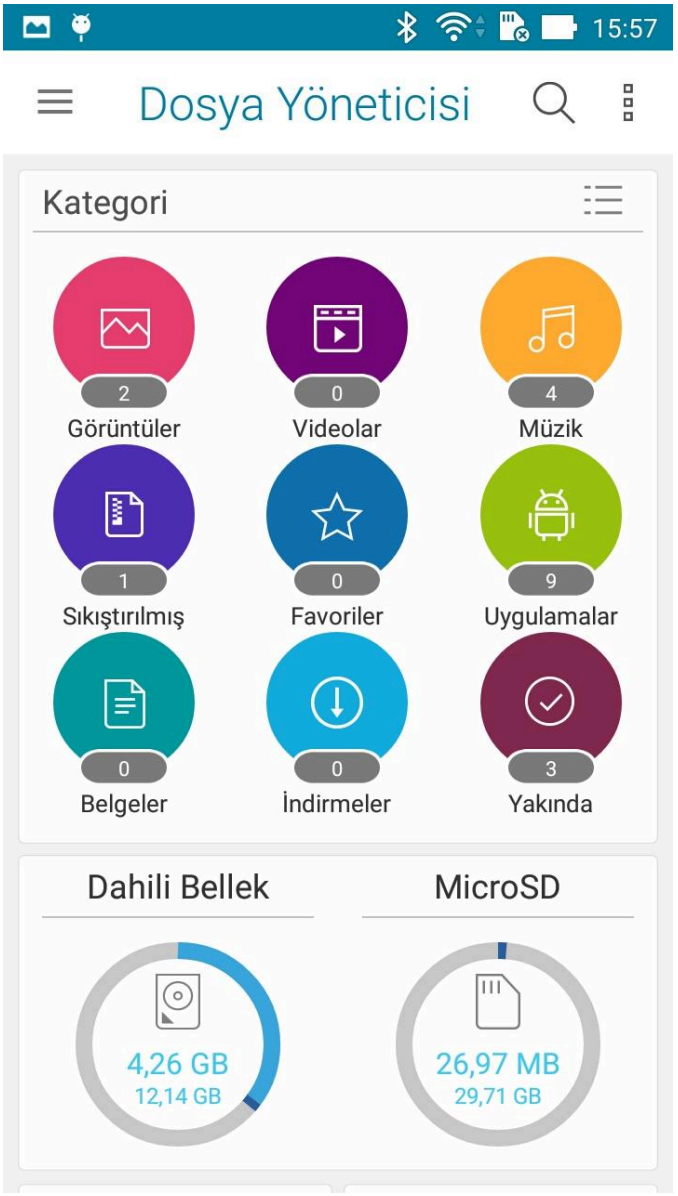
Şekil 26. “Mobil Uygulama Geliştirme” klasörünün SD karta kopyalanması.

Mobil uygulamamızı oluşturan dosya ve klasörlerin mobil cihaza aktarılma işlemi tamamlandı. Artık testi gerçekleştirmek için mobil cihazımızın uygulamaları içinde yer alan dosya yöneticisi ya da benzeri bir program yardımıyla uygulama dosyalarına ulaşabiliriz (Şekil 27).



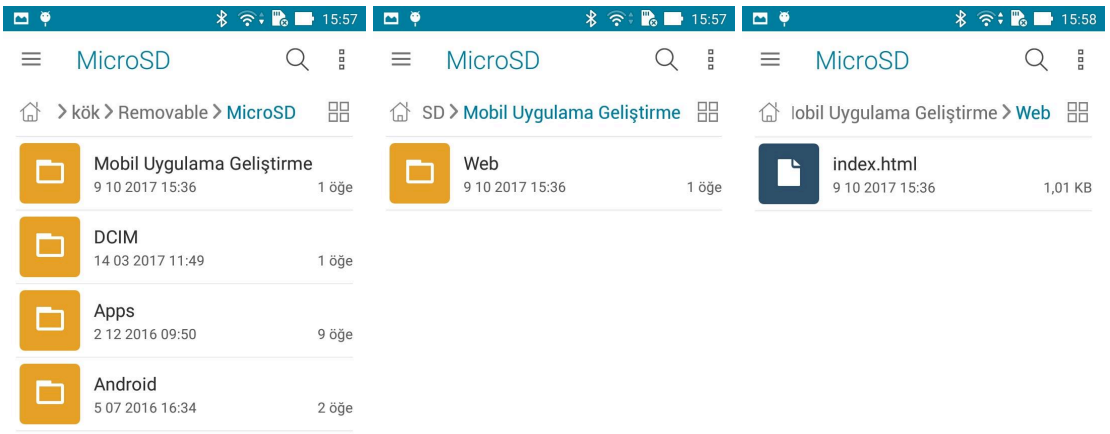
Şekil 27. Mobil cihazın uygulamalar bölümü.

Dosya yöneticisi uygulamasına giriş yaptıktan sonra Micro SD kartın olduğu bölüme giriş yapıyoruz (Şekil 28).



Şekil 28. Dosya Yöneticisi uygulaması ve mobil cihazın depolama alanları.

Micro SD kart içerisine giriş yaptıktan sonra sırasıyla “Mobil Uygulama Geliştirme” ve “Web” klasörlerinin içine girerek uygulamamızın yer aldığı klasöre kadar ilerlemiş oluyoruz (Şekil 29).



Şekil 29. Mobil uygulamanın yer aldığı klasöre erişim.

“index.html” adlı dosyayı gördüğümüzde artık bu dosyanın üzerine dokunarak uygulamamızın testini gerçekleştirebiliriz. Mobil cihazınızda HTML uzantılı dosyaları görüntüleyebilecek birden fazla uygulama yer alabilir. Bu durumda istediğiniz herhangi bir İnternet tarayıcıyı test için seçebilirsiniz (ör: Standart Android İnternet tarayıcı, Chrome, Opera Mobil). Bu çalışmamızda mobil cihazımızda yer alan Standart Android İnternet tarayıcısını (ilk sıradaki “Tarayıcı”) seçiyoruz (Şekil 30).



Şekil 30. Test için tarayıcının seçilmesi.

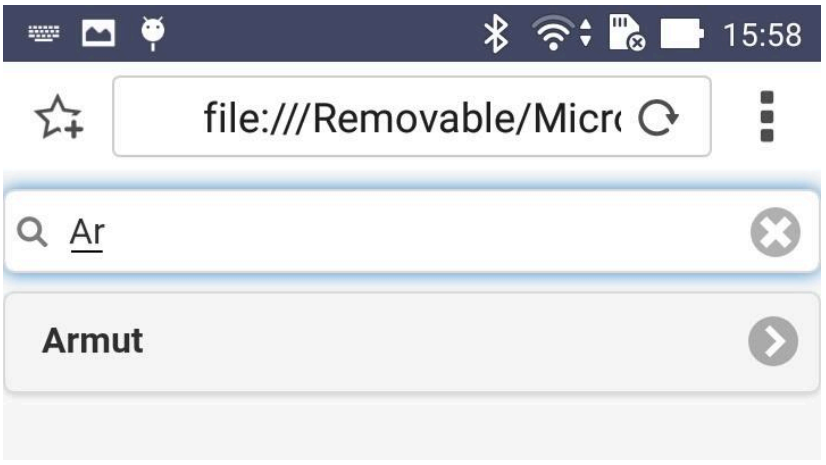
Görüntüleyiciyi seçtiğinizde otomatik olarak ilgili uygulama açılacak ve mobil uygulamanın yer aldığı HTML dosyası ekranda kullanılabilir şekilde yer alacaktır (Şekil 31).



Şekil 30. Mobil tarayıcıda uygulamanın görüntülenmesi.

Bu aşamada artık mobil cihazımızda uygulamamız kullanıma hazır hale gelmiş oldu. Daha önce kişisel bilgisayarımızda test sırasında gerçekleştirdiğimiz işlemleri şimdi mobil cihazımız üzerinde deneyebiliriz.

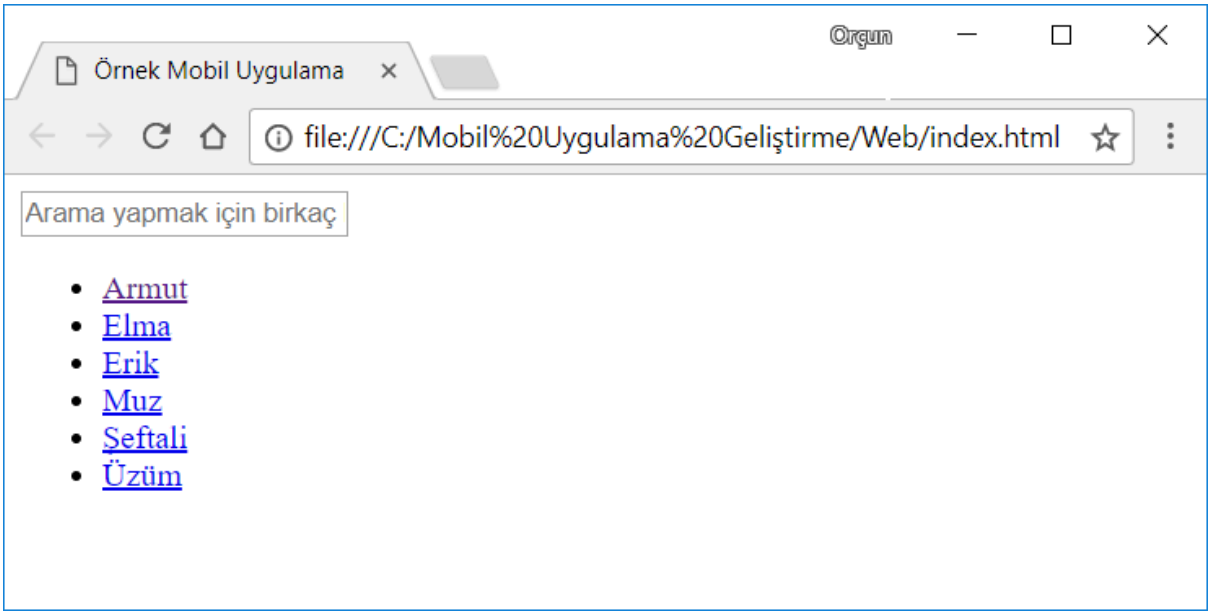
Arama kutucuğuna birkaç harf girdiğimizde listenin filtrelendiğini ve güncellendiğini görüntüleyebiliyor olmanız gerekir (Şekil 31).



Şekil 31. Arama kutucuğuna harf girilerek test işleminin gerçekleştirilmesi.

2.3.5.5 Mobil Uygulamada Test Aşamasında Yaşanabilecek Sorunlar ve Çözümleri

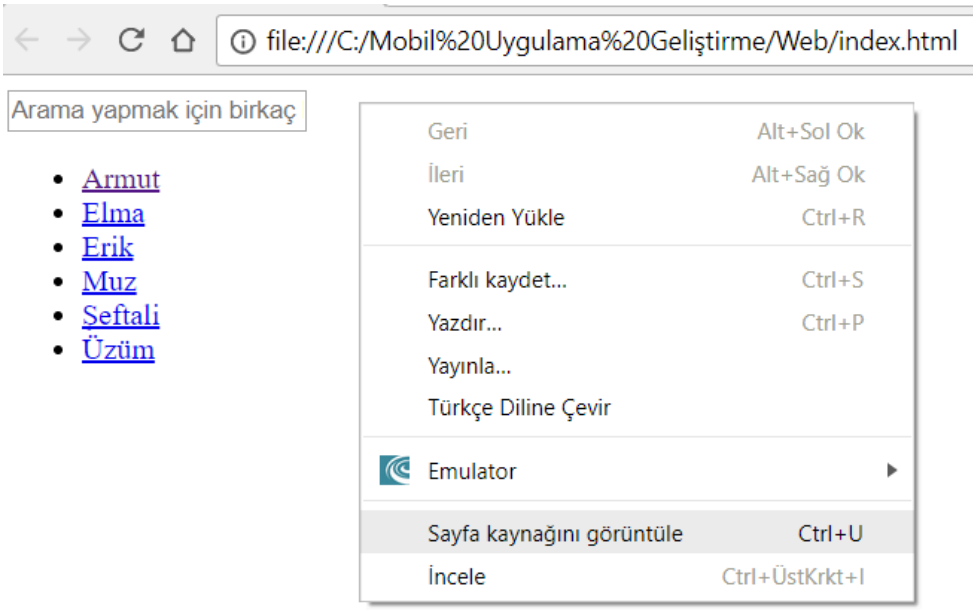
Web tabanlı bir mobil uygulama geliştirme sırasında yaşanabilecek sorunlardan biri uygulamanızda kullandığınız yazılım kütüphanelerine (örnek uygulamamızda kullanılan JQuery, JQuery Mobile vb. gibi) uygulama içerisinden erişilememesidir. Bu durumun temel sebepleri İnternet bağlantısı ile ilgili sorunlar ya da kütüphane adreslerinin yanlış yazılması olabilir. Her iki durumda da, gerekli olan Javascript ve CSS dosyalarına ulaşılamadığı için uygulama tarayıcıda görüntülenecek ancak hem biçimsel olarak hem de fonksiyon açısından istediğimiz sonucu vermeyecektir (Şekil 32).



Şekil 32. Gerekli kütüphanelere erişemeyen uygulamanın İnternet tarayıcıdaki görüntüsü

Şekil 32'de görüldüğü gibi daha önceki testimize göre biçimsel olarak bozulmuş bir arayüz ile karşı karşıyayız; aynı zamanda arama yapmak istediğimizde filtreleme işlemi de gerçekleşmeyecektir. Bu aşamada çözüm için öncelikle İnternet bağlantımızın olup olmadığını kontrol etmemiz, daha sonra ise harici olarak bağlantı kurduğumuz yazılım kütüphanelerinin adreslerini kontrol etmemiz gerekir. İnternet bağlantımızın olduğunu varsayarak harici dosyaların adreslerini nasıl kontrol edebileceğimize bakalım.

Kontrol işlemi için sorun yaşadığımız HTML sayfanının boş bir bölümünde sağ klik yaparak “Sayfa kaynağını görüntüle” menü öğesine tıklayalım (Şekil 33).



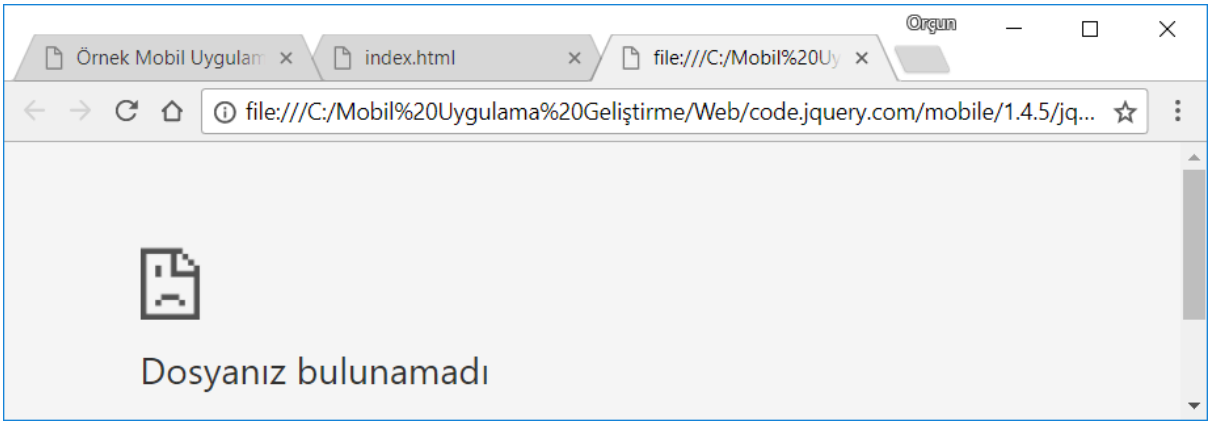
Şekil 33. HTML sayfasının kaynağının görüntülenmesi için gerekli komutun verilmesi.

İnternet tarayıcımızda açılan yeni bir sekmede HTML sayfasının kaynak kodu görüntülenecektir. Bu kaynak kod bizim Brackets içerisinde yazdığımız kodun aynısıdır ancak içerisindeki bağlantılar tıklanabilir durumdadır ve bu sayede harici bağlantılı olan dosyalara erişilebiliyor mu, bunu test etmemiz mümkün hale gelir (Şekil 34).



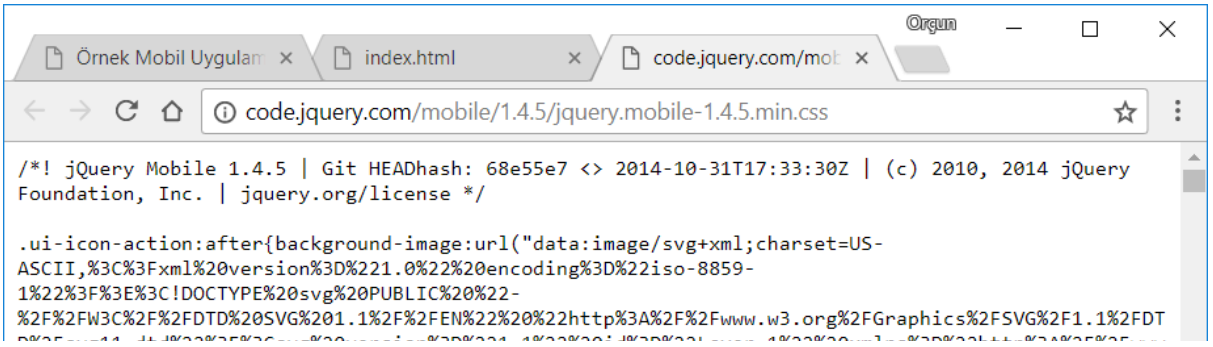
Şekil 34. HTML kaynak kodun İnternet tarayıcıda görüntülenmesi.

Kaynak kod içerisinde yer alan 7., 8. ve 9. satırlar harici kütüphanelerin erişim adreslerini içermektedir. Şimdi bu adreslerin üzerine tıklayarak “css” ve “js” uzantılı dosyalara erişebiliyor muyuz kontrol edelim.



Şekil 35. Dosyanın bulunamadığına dair İnternet tarayıcıda yer alan hata mesajı

Şekil 35’de görüntülendiği gibi dosya bulunamadı hata mesajı ile karşılaşıldı. Bunun sebebi harici dosyaların bağlantı adreslerinin yanlış olmasıdır (İnternet bağlantınızın olduğu varsayılmaktadır). Örnek kodda bağlantı adreslerinin başında bulunan “http://” ön eki silinmiştir. Bu da dosyaya erişimi engellemektedir. Harici dosyalarının bağlantı adreslerini Brackets’dan güncelleyelim ve tekrar erişim sağlanabilecek bir yapıya kavuşturalım. Adresler güncellendiğinde sayfa kaynağındaki bağlantılara tıkladığınızda ilgili dosyayı İnternet tarayıcınızda görüntüleyebiliyor olmanız gerekir (Şekil 36).



Şekil 36. Bağlantısına tıklanan CSS dosyasının İnternet tarayıcıda görüntülenmesi.

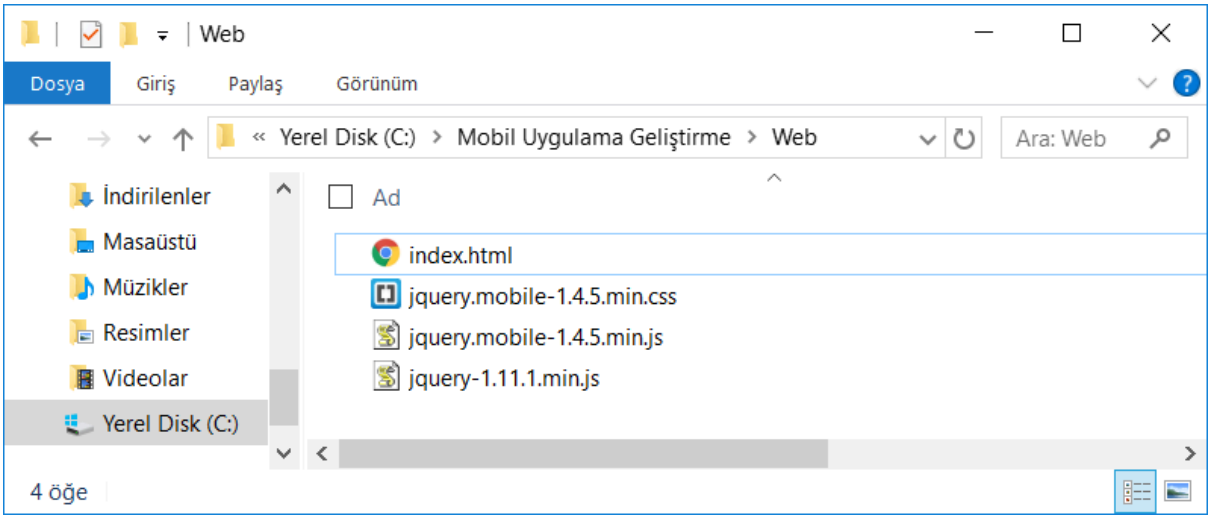
2.3.5.6 Mobil Uygulamada Kullanılan Harici Kütüphanelerin İnternet Bağımlılıklarının Sona Erdirilmesi

Web tabanlı olarak geliştirilen uygulamalarda harici kütüphanelerin kullanımında genelde izlenen yol kütüphanelerin İnternet üzerindeki orijinal kaynaklarından kullanılmasıdır. Bunu sebebi kütüphanelerde yapılan güncelleştirme ve iyileştirmelerden hızlı şekilde faydalanmaktadır. Ancak bu durum beraberinde İnternet'e bağlı olma zorunluluğunu da beraberinde getirir. Bu Web siteleri için çok önemli bir sorun teşkil etmez, zaten bu sitelerin ziyaret edilebilmesi için kullanıcının cihazında İnternet bağlantısı olması gerekir. Bir mobil uygulamada ise bu soruna neden olabilir. İnternet bağlantısı olmayan bir mobil cihaz, uygulamayı düzgün şekilde görüntüleyemeyecektir ve birçok fonksiyon çalışmayacaktır. Bu açıdan kullanılan kütüphanelerin dosyalarını İnternet ortamında indirilerek uygulama klasörünün içinde yer verilmesi tercih edilebilir.

Yukarıda bahsedilen süreç ile ilgili birkaç farklı yol izlenebilir. Bu yollardan birincisi kullanılan kütüphanelerin İnternet sitelerinden ilgili paketlerin indirilmesidir. Örnek olarak uygulamada JQuery Mobile ile ilgili kütüphaneler kullanılacaksa http://jquerymobile.com/download/ adresinden son yayınlanan paketin indirilmesi gerekir.

Diğer yol ise geliştirdiğiniz mobil uygulamanın çalışan halindeki kütüphane bağlantılarını takip ederek bu dosyaları teker teker kişisel bilgisayarınızdaki çalışma klasörünüze indirmektir. Bu yöntem çalışan sistemin içindeki dosyaların indirilmesi açısından daha pratik ve sonuca daha hızlı ulaşabileceğiniz bir yöntemdir.

Yukarıda anlatılan her iki yöntemde de kütüphane dosyalarını kişisel bilgisayarınıza indirip proje klasörünün içine yerleştirdikten sonra HTML kodu içerisinde kütüphanelere bağlantı sağladığınız adresleri yerel sürümleri ile güncellemeniz gerekir. Şekil 37'de proje klasörünüze kaydettiğiniz dosyalara bağlı olarak kodda nasıl bir güncelleme yapmanız gerektiği görüntülenmektedir.

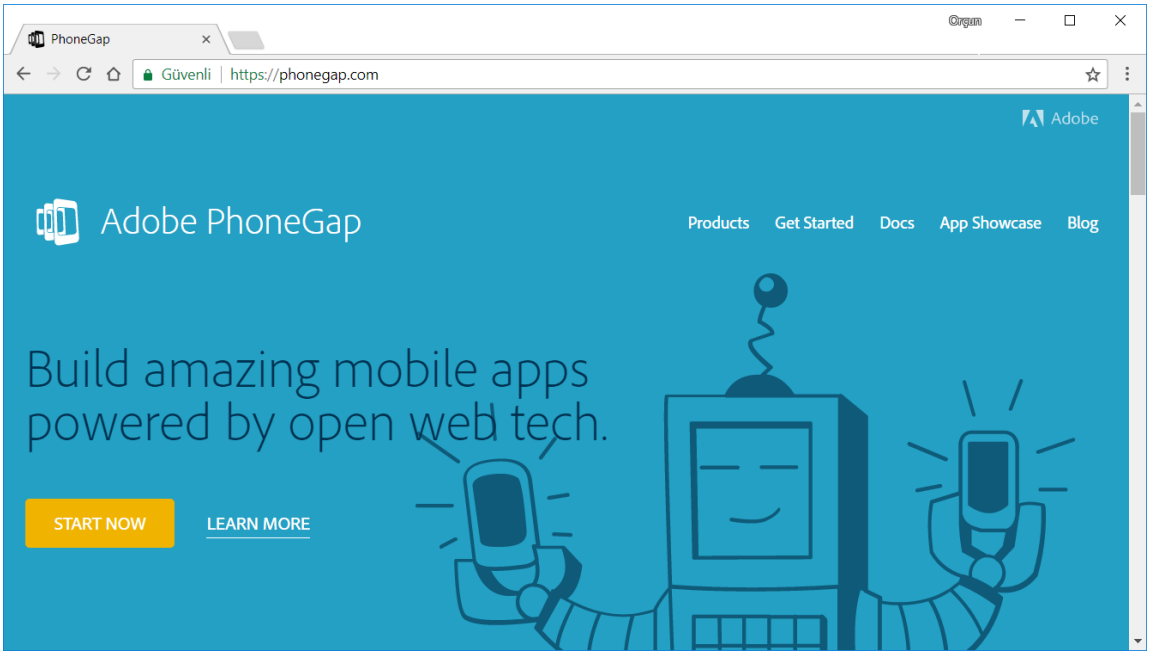



```
<html>
<head>
    <title>Örnek Mobil Uygulama</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
    <script src="jquery-1.11.1.min.js"></script>
    <script src="jquery.mobile-1.4.5.min.js"></script>
</head>
```

Şekil 37. Kütüphane dosyalarının yeni lokasyonuna göre adreslerin güncellenmesi.

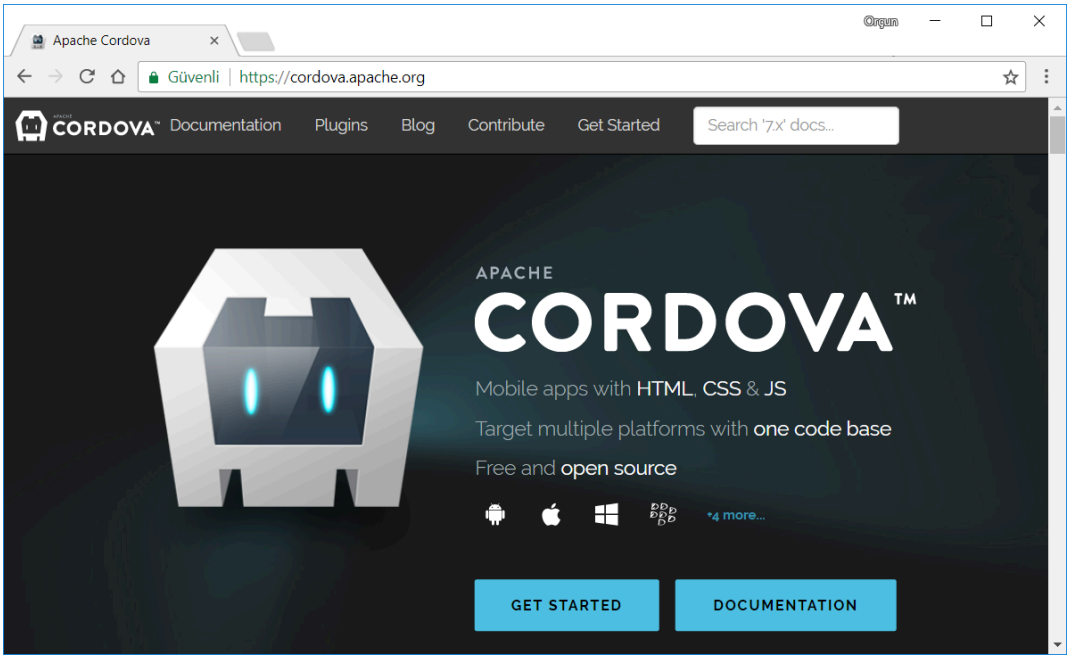
2.3.6 Web Tabanlı Mobil Uygulamanın Yerel (Native) Bir Uygulama Olarak Paketlenmesi

Mobil uygulamaların Web tabanlı olarak geliştirilmesi her ne kadar süreci hızlandırsa da bir takım sınırlılıkları da beraberinde getirmektedir. Hem Web tabanlı geliştirme ortamının avantajlarından yararlanmak, hem de beraberinde getirdiği sınırlılıkları en aza indirmek için geliştirilen uygulamanın sanki Yerel (Native) bir uygulamaymış gibi paketlenip mobil cihazlara yüklenebilmesi mümkündür. Bu şekilde paketleme işlemini yapan birçok platform bulunmaktadır. Bu platformların en gelişmiş olanlarından biri Adobe PhoneGap'dir. PhoneGap'in geliştiricisi Nitobi firmasının Adobe tarafından satın alınmasıyla PhoneGap iki farklı platform üzerinde geliştirilmeye devam etmektedir. Bu platformlardan biri Adobe'nin geliştirme çalışmalarına devam ettiği "Adobe PhoneGap"'dir (Şekil 37).



Şekil 37. Adobe PhoneGap (https://phonegap.com/)

Diğer platform ise PhoneGap'in kaynak kodlarının Apache Yazılım Vakfı'na (Apache Software Foundation) devredilmesiyle ortaya çıkan ve vakıf tarafından desteklenen Cordova Projesi'dir (Şekil 38).

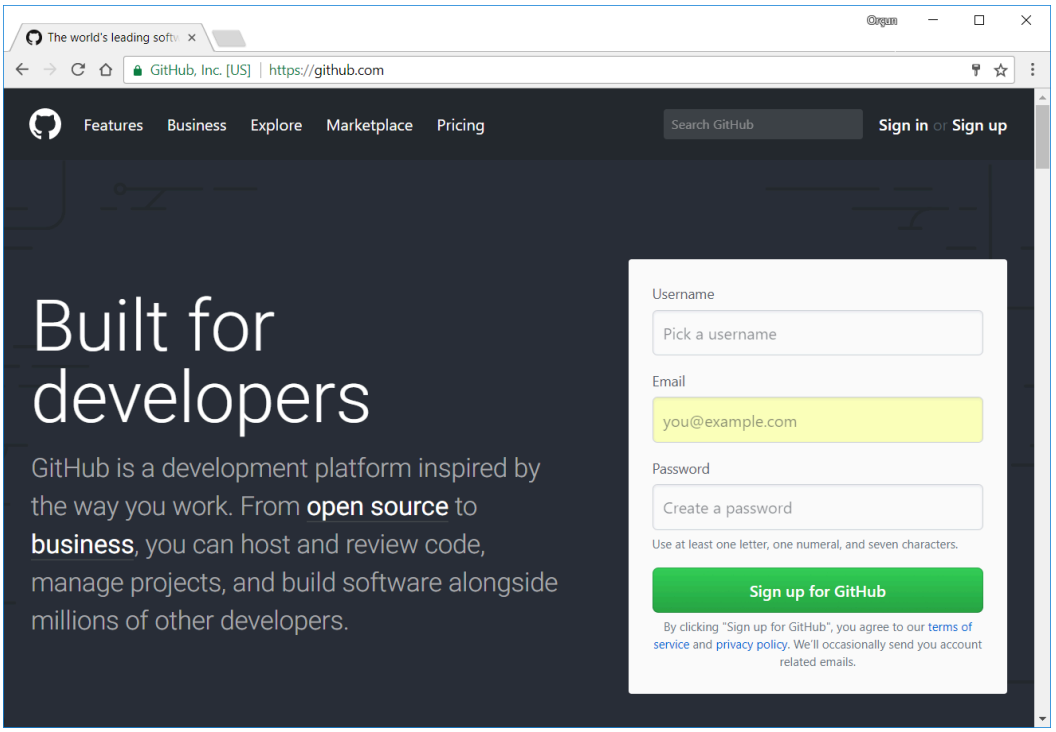


Şekil 38. Apache Cordova Projesi (https://cordova.apache.org/)

Apache Cordova, açık kaynak kodlu bir proje olması açısından, Adobe PhoneGap ise desteklediği yazılım kütüphanelerinin zenginliği açısından avantajlar sunmaktadır. Buna ek olarak Adobe PhoneGap paketleme işlemini kişisel bilgisayarınıza herhangi bir özel yazılım yüklemenize gerek kalmadan bulut üzerinde gerçekleştirebilmektedir. Yine Web tabanlı bir emülatörü de bulunan Adobe PhoneGap bu hizmetleri ücretsiz olarak vermektedir. Bu çalışmamızda geliştirdiğimiz Web tabanlı mobil uygulamayı Adobe PhoneGap platformu yardımıyla paketleyerek mobil cihazımıza yükleyeceğiz.

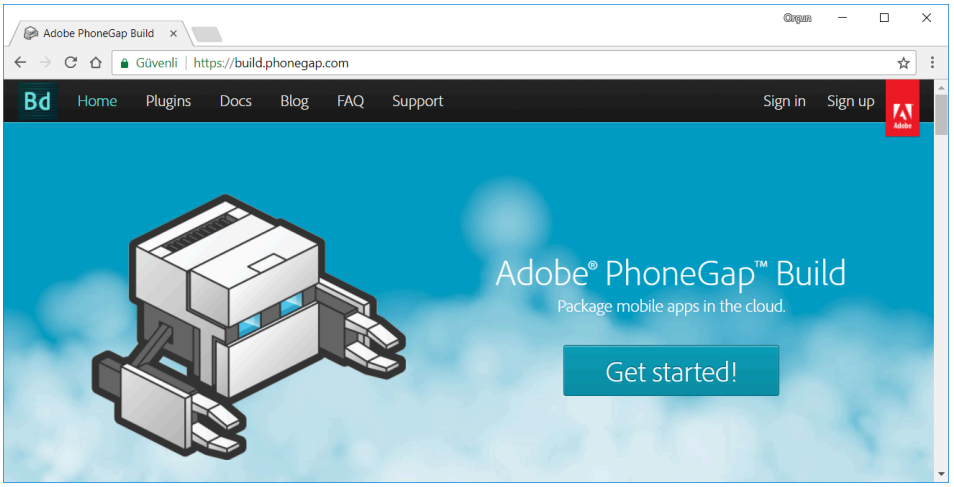
2.3.6.1 Mobil Uygulamanın Paketlenmesi için Kullanılacak Bulut Platformları

Yukarıda da belirtildiği gibi Adobe PhoneGap yardımıyla tüm süreç bulut üzerinde bir İnternet tarayıcısı yardımıyla gerçekleştirilebilmektedir. Bu süreci yönetebilmek için iki farklı bulut platformunda kullanıcı hesabına ihtiyacımız bulunmaktadır. Bu bulut platformlarından ilki dosyalarımızın yükleneceği ve depolanacağı GitHub kod deposudur (Şekil 39).



Şekil 39. GitHub kod deposu (https://github.com/)

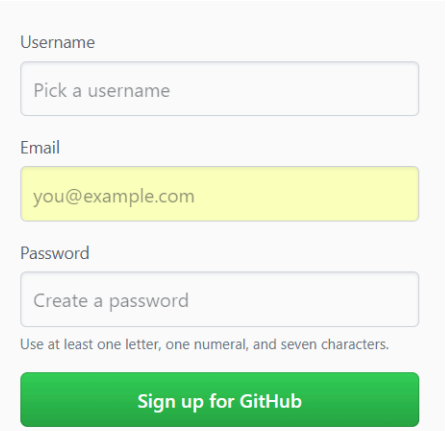
Paketleme işlemini gerçekleştirecek platform olan Adobe PhoneGap Build için de bir Adobe hesabına ihtiyacımız bulunmaktadır (Şekil 40).



Şekil 40. Adobe PhoneGap Built (https://build.phonegap.com/).

Her iki bulut platformu için de kullanıcı üyeliği ücretsizdir. Geçerli bir e-posta adresinizin olması durumunda kullanıcı hesabınızı kolaylıkla oluşturabilirsiniz.

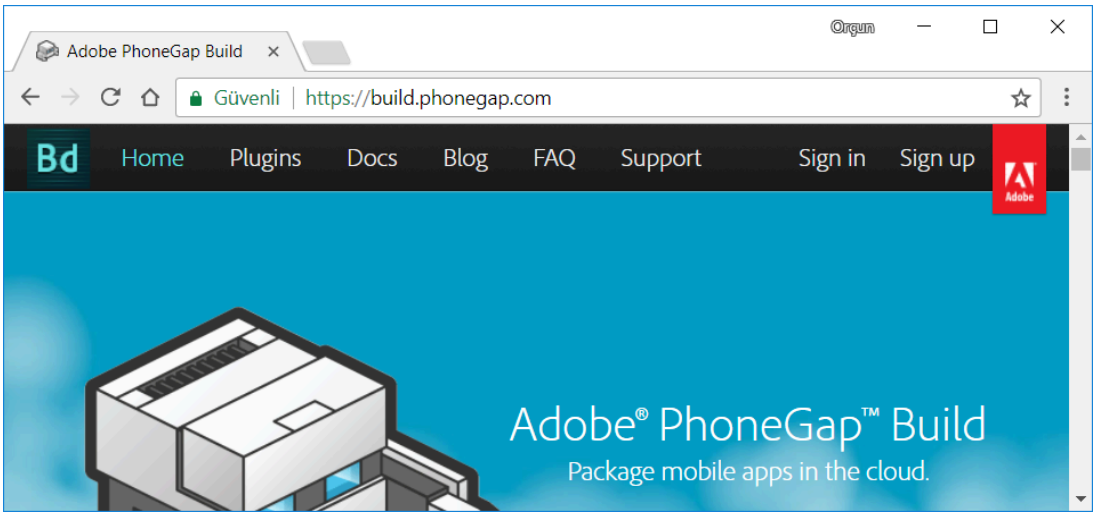
İlk olarak GitHub platformu üzerinde kullanıcı hesabımızı oluşturalım ve giriş yaparak mobil uygulamamızın dosyalarını kod deposuna yükleyelim. İnternet tarayıcımızdan github.com adresine giriş yaptığımızda yeni bir hesap oluşturmak için sayfanın sağ tarafında yer alan formu doldurarak “Sign up for Github” düğmesine tıklayalım (Şekil 41).



Şekil 41. Yeni bir kullanıcı hesabı oluşturmak için kullanılacak iletişim formu.

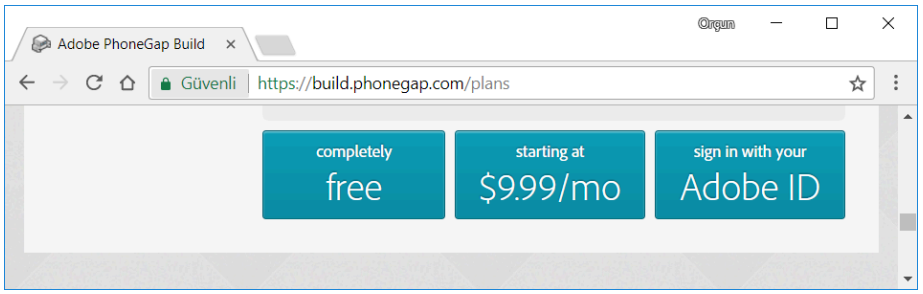
Kayıt işlemi sırasında istenen bilgiler arasında Kullanıcı Adı (Username), E-Posta (Email) ve Şifre (Password) yer almakta. Bu bilgileri dikkatli bir şekilde doldurmakta fayda var çünkü daha sonra hem sisteme giriş yaparken bu bilgilere ihtiyacınız olacak.

Şimdi ise Adobe Built için gerekli olan kullanıcı hesabını oluşturalım. https://build.phonegap.com/ adresine giriş yaptığımızda sağ üst köşede “Sign in” (Giriş yap) ve “Sign up” (Kayıt yaptır) seçenekleri bulunmakta. Bu seçeneklerden “Sign up” (Kayıt yaptır) seçeneğini tıklayalım (Şekil 42).



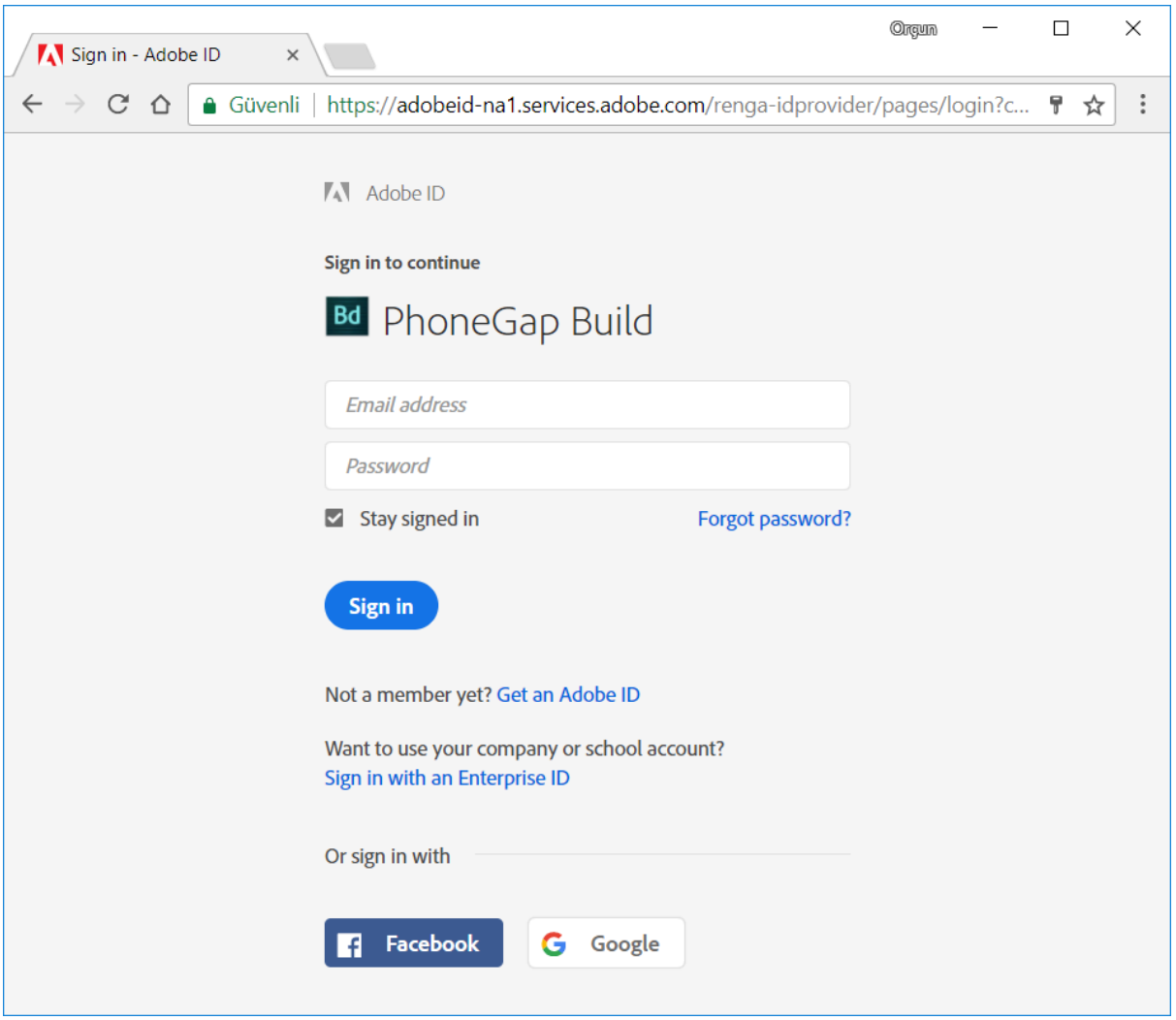
Şekil 42. Adobe Built Web sitesinde “Sign up” (Kayıt yaptır) menü öğesinin seçilmesi.

“Sign up” (Kayıt yaptır) menü öğesine tıklandığında platformu kullanım için sunulan farklı seçeneklerin yer aldığı bir sayfaya ulaşılır. Bu sayfada ücretsiz olan planı “Completely free” düğmesine basarak seçelim (Şekil 43).



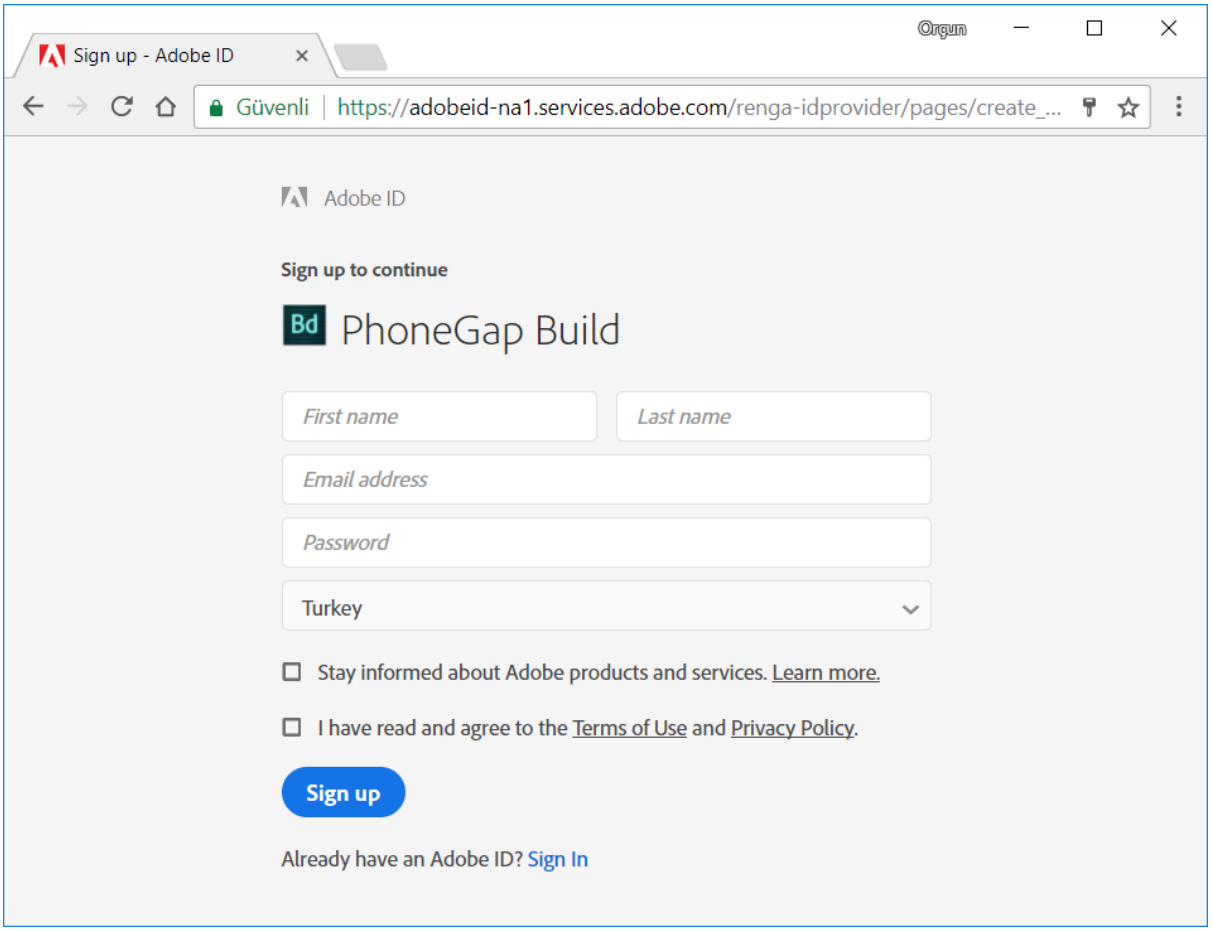
Şekil 43. Ücretsiz olan kullanım planının seçilmesi.

Kullanım planı seçimi sonrasında sisteme giriş için kullanıcı adı ve şifrenin olduğu bölüm ekrana gelecektir. Bu ekranda eğer daha önceden bir Adobe Hesabı varsa bu hesap ile giriş yapılabilir, Facebook ve / veya Google hesapları ile giriş yapılabilir ya da yeni bir Adobe Hesabı oluşturmak için “Get an Adobe ID” (Bir Adobe Kimliği al) bağlantısına tıklanır (Şekil 44).



Şekil 44. Adobe Built platformu için kullanıcı hesabının seçilmesi ya da oluşturulması.

Yeni bir Adobe Hesabı oluştumak için “Get an Adobe ID” (Bir Adobe Kimliği al) bağlantısına tıklandığı zaman kullanıcı hesabının oluşturulacağı iletişim formu görüntülenir. Bu iletişim formundaki bilgileri dikkatli bir şekilde doldurduktan sonra “Sign up” (Kayıt ol) düğmesine tıklayarak kayıt işlemi tamamlanır (Şekil 45).



Şekil 45. Adobe ID (Adobe Kimliği) için iletişim formu.

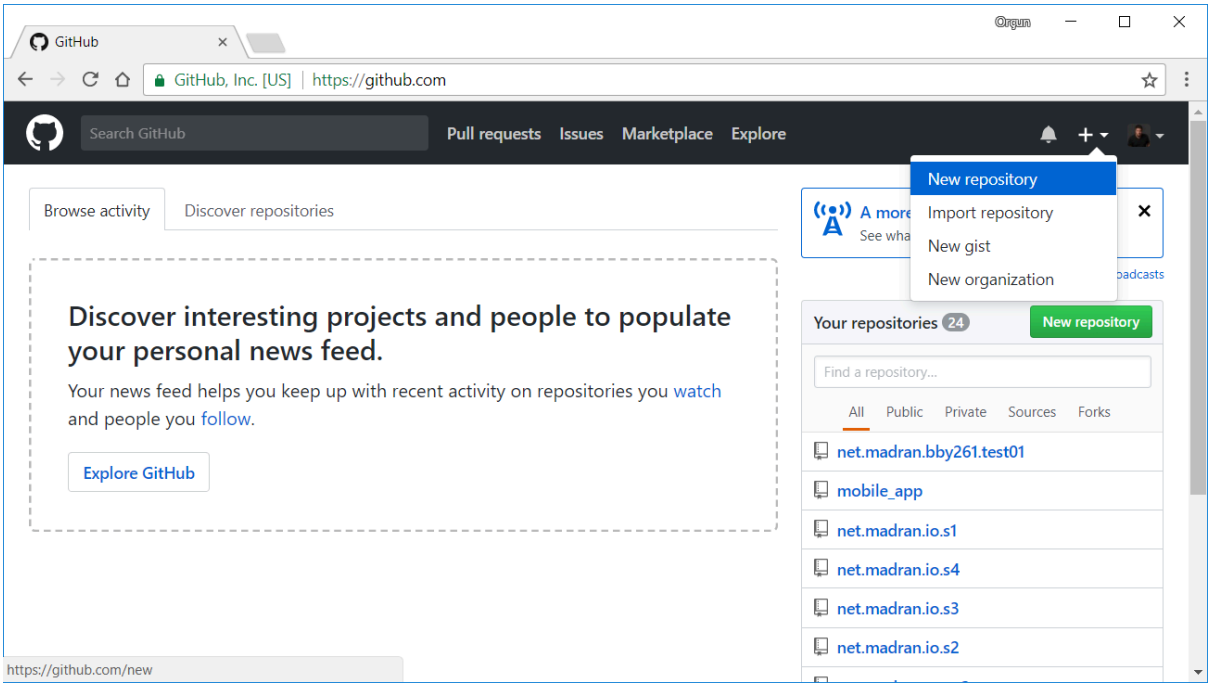
Bu ve benzeri kayıt işlemlerinde bulut platformları çoğunlukla kayıt olurken verilen e-posta adresine bir doğrulama postası gönderir ve bu e-posta mesajının içerisindeki bağlantıya tıklayarak kullanıcı hesabının aktif hale getirilmesini isterler. Bu açıdan kayıt sırasında e-posta adresi forma yazılırken doğruluğunu kontrol etmek çok önemlidir.

2.3.6.2 Mobil Uygulamanın GitHub Kod Deposuna Yüklenmesi

Geliştirilen uygulamanın GitHub kod deposuna yüklenebilmesi için daha önce oluşturulan kullanıcı hesabı bilgileri (kullanıcı adı ya da e-posta ve şifre) ile platforma giriş işlemi gerçekleştirilir. Giriş panelinin görüntülenebilmesi için https://github.com/ ana sayfasının sağ üst köşesinde yer alan “Sign in” (Giriş yap) düğmesine tıklanır.

Giriş işlemi gerçekleştirildiğinde öncelikli olarak kod deposu içerisinde yeni bir depo oluşturulması gerekmektedir. Yeni bir deponun oluşturulması için sayfanın sağ üst

köşesindeki “+” işaretine tıklayarak “New repository” (Yeni depo) menü öğesi seçilmeli ya da doğrudan sayfanın sağ bölümünde yer alan “New repository” (Yeni depo) düğmesine tıklanmalıdır (Şekil 46).



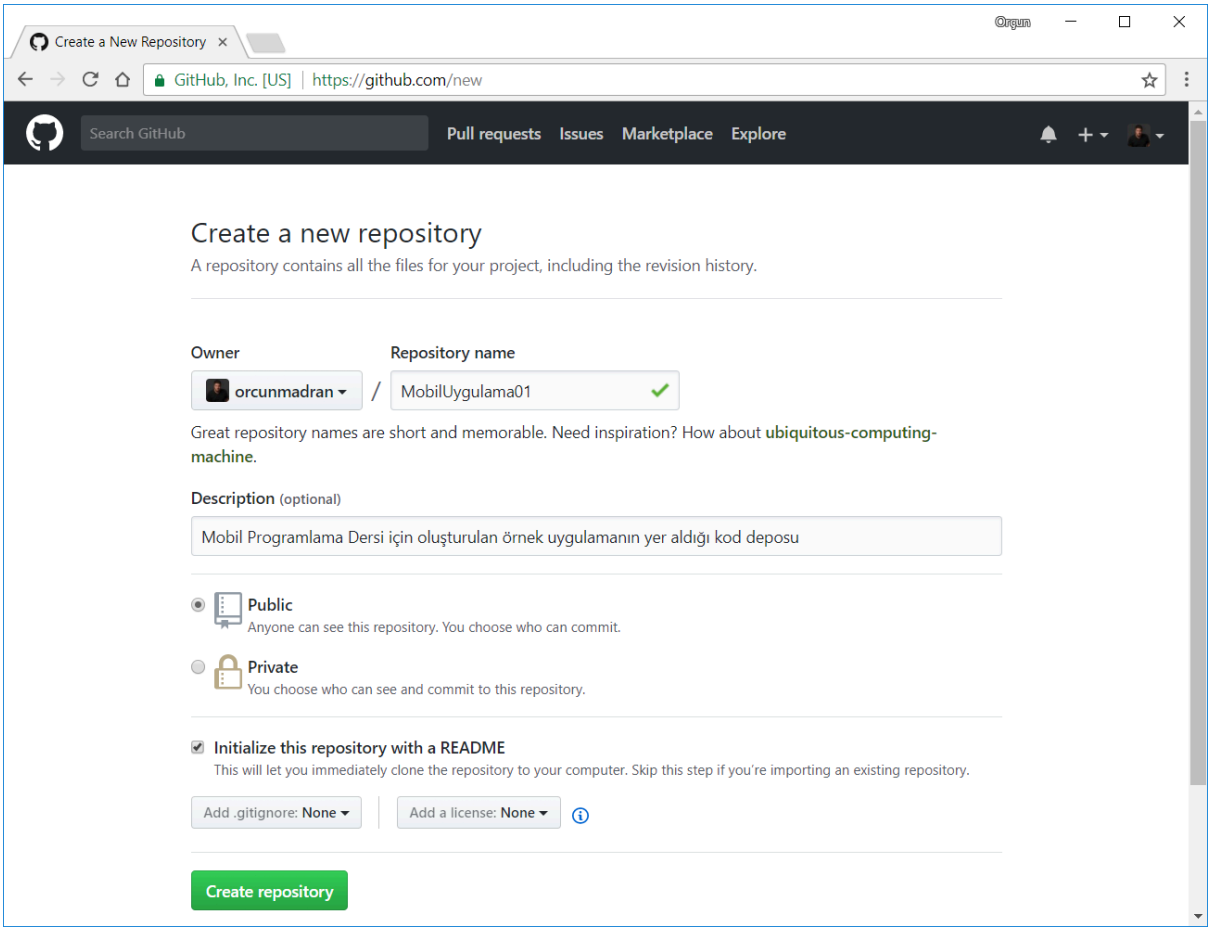
Şekil 46. “New repository” (Yeni depo) oluşturulması.

GitHub içerisinde birden çok kod deposu oluşturulabilir, her bir depo bir projeyi / uygulamayı temsil edecektir. Aynı uygulamanın birden fazla sürümü de bir kod deposu içinde yer alabilir.

Yeni bir depo oluşturmak için Şekil 47’de görüntülenen form aracılığıyla projenin yer alacağı depo ile ilgili temel bilgiler girilir. Bu bilgiler arasında en önemlisi deponun adının yer aldığı “Repository name” alanıdır. Bir depo adı bir kullanıcı hesabı içerisinde ancak bir kere kullanılabilir. Depo adını oluştururken boşluk, Türkçe karakter ve noktalama işaretlerini kullanmamak gerekir (noktalama işaretlerinden bazıları kullanılabilir, ör: nokta).

Depo adının bir başka önemi ise bu adın mobil cihazlar için oluşturulan uygulama marketlerinde paket adı olarak kullanılabilecek olmasıdır. Bu açıdan bir mobil uygulama resmi olarak uygulama marketlerde yer alacaksa, bu marketlerin belirlemiş olduğu paket adı özelliklerine uygun olarak paket adının (dolayısıyla depo adının)

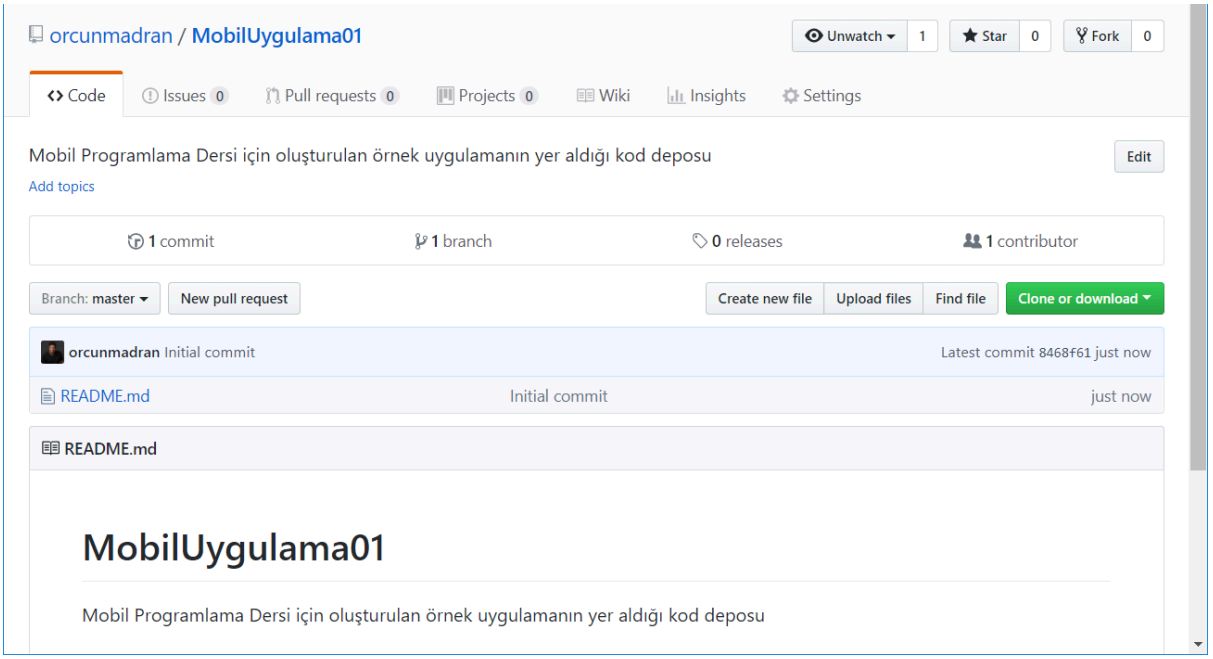
belirlenmesi gerekir. Bu genelde uygulamayı geliştiren organizasyonun Web sitesi olarak kullandığı alan adının tersten kullanımı şeklinde bir algoritma içerir ve bu şekilde dünya üzerinde benzersiz bir paket adı oluşturulabilir.



Şekil 47. Yeni depo oluşturulması için kullanılacak form.

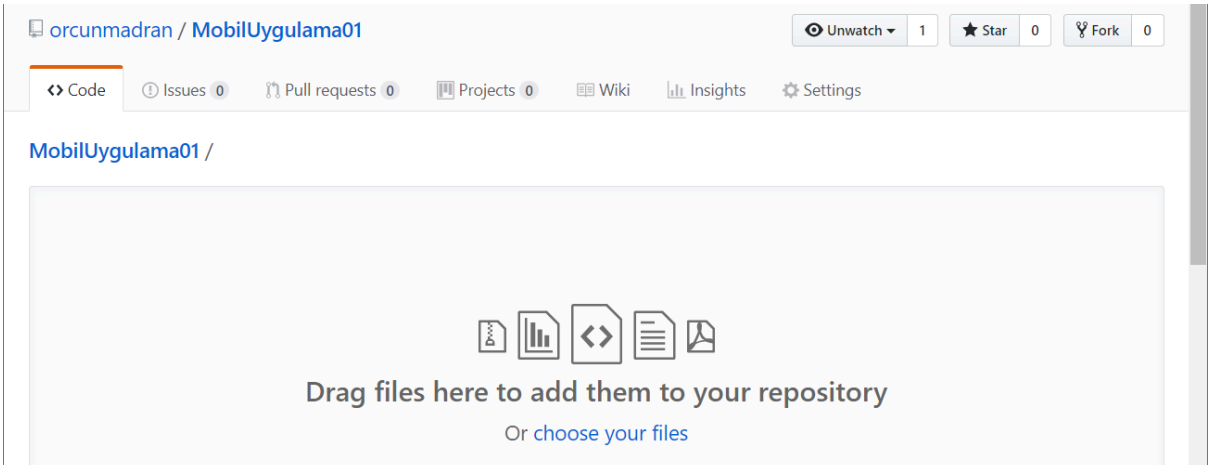
Kod deposu oluşturulurken depo hakkında genel bir bilgi verilmesi amacıyla "README" (BENİOKU) adlı bir dosyanın da oluşturulmasında fayda vardır. Şekil 47'de yer alan formdaki "Initilize this repository with a README" seçeneği işaretlenerek bu dosyanın otomatik olarak oluşturulması sağlanabilir. Bu şekilde kod deposuna Web üzerinden direkt dosya yüklenmesini sağlayacak başlangıç işlemleri de tamamlanmış olur.

"Create repository" (depoyu oluştur) düğmesine tıklandığında yeni deponun oluşturulması için gerekli komut verilmiş olur ve eğer form içeriği ile ilgili bir sorun yoksa Şekil 48'deki yeni kod deposunun içeriğini gösteren ekran görüntülenir.



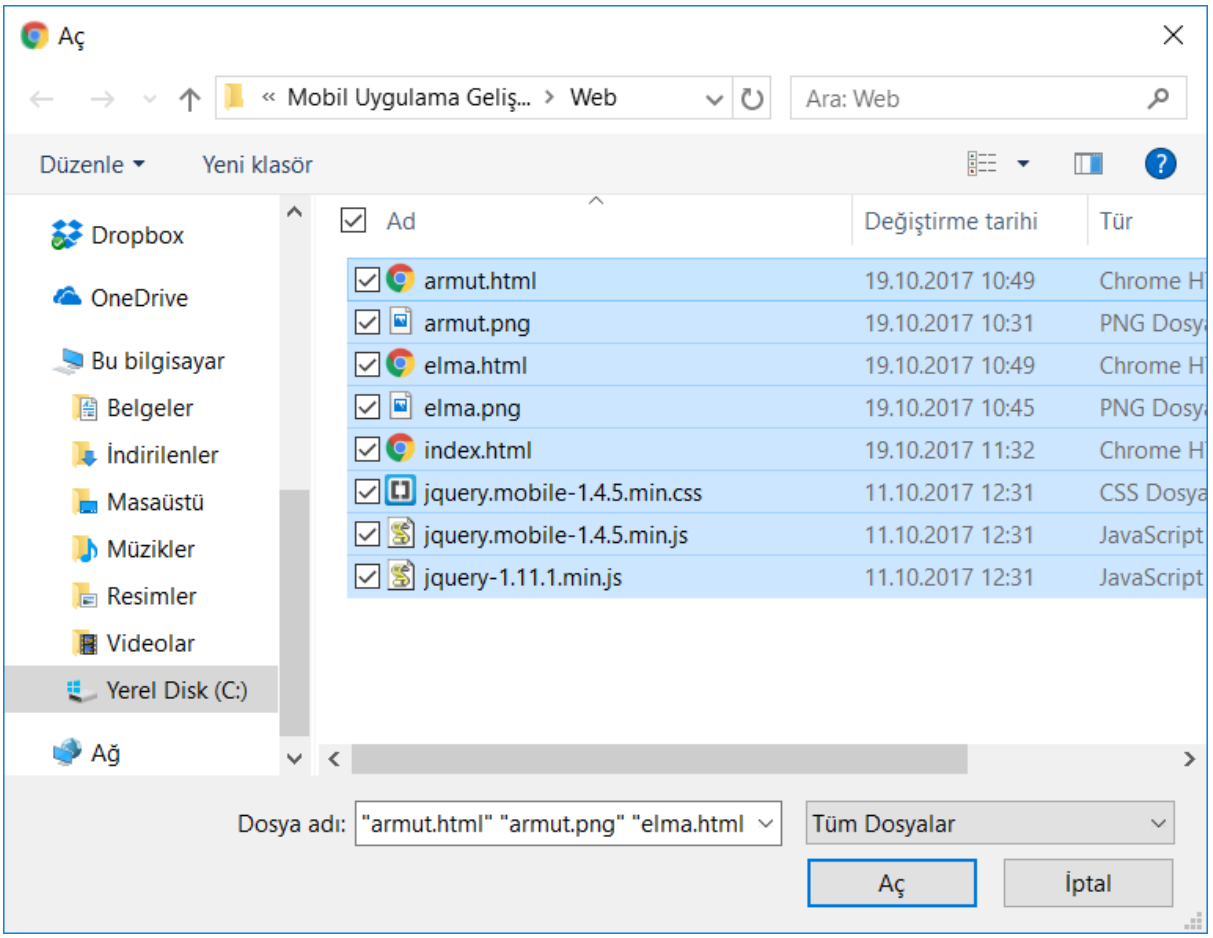
Şekil 48. Oluşturulan yeni deponun görüntülenmesi.

Yeni deponun içinde şu an için sadece uygulama ile ilgili genel bilgilerin yer aldığı README.md dosyası bulunmaktadır. Uygulamayı oluşturan tüm dosyaların bu kod deposuna yüklenmesi gerekmektedir. Bu yükleme işlemi dosya listesinin yer aldığı bölümün hemen sağ üst köşesinde yer alan “Upload files” (Dosyaları yükle) düğmesi ile yapılır. Düğmeye tıklandığında Şekil 49’da yer alan ekran görüntülenir.



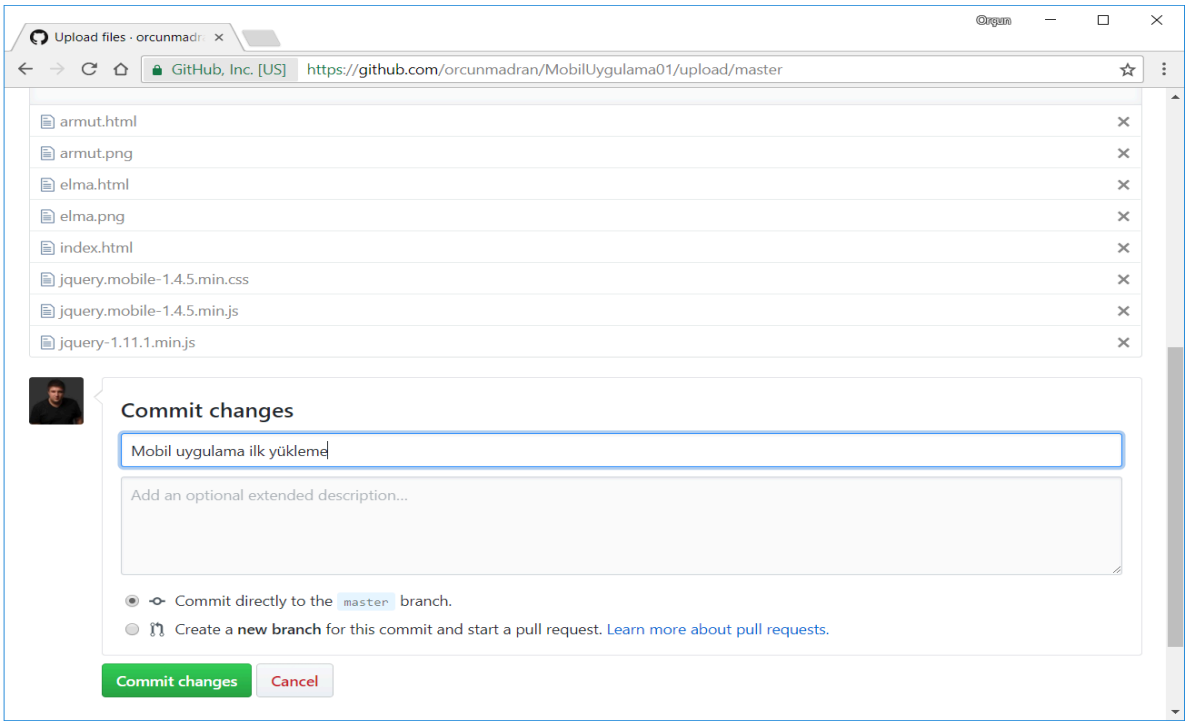
Şekil 49. Dosyaların yükleneceği arayüz.

Bu ekran yardımıyla iki farklı şekilde dosya yükleme işlemi gerçekleştirilebilir; dosyaları seçtikten sonra sürükleyip gri bölüme bırakarak ya da bilgisayardan dosyaları seçerek. Orta bölümde yer alan “choose your files” (dosyalarınızı seçiniz) bağlantısına tıkladığınızda dosyaları seçebileceğiniz iletişim kutusu görüntülenecektir (Şekil 50).



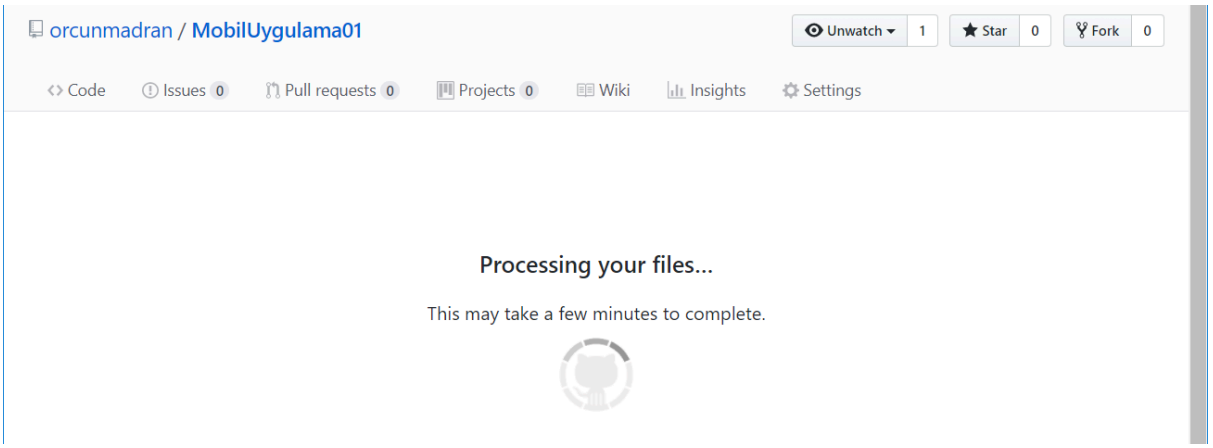
Şekil 50. Dosyaların seçileceği iletişim kutusu.

Tüm dosyalar işaretlendikten sonra “Aç” düğmesine tıklanır ve yükleme işlemi öncesi dosya seçimi tamamlanmış olur. Kod deposuna dosyaların teslim edilmesi için bir mesaj eşliğinde sürecin onaylanması istenir.



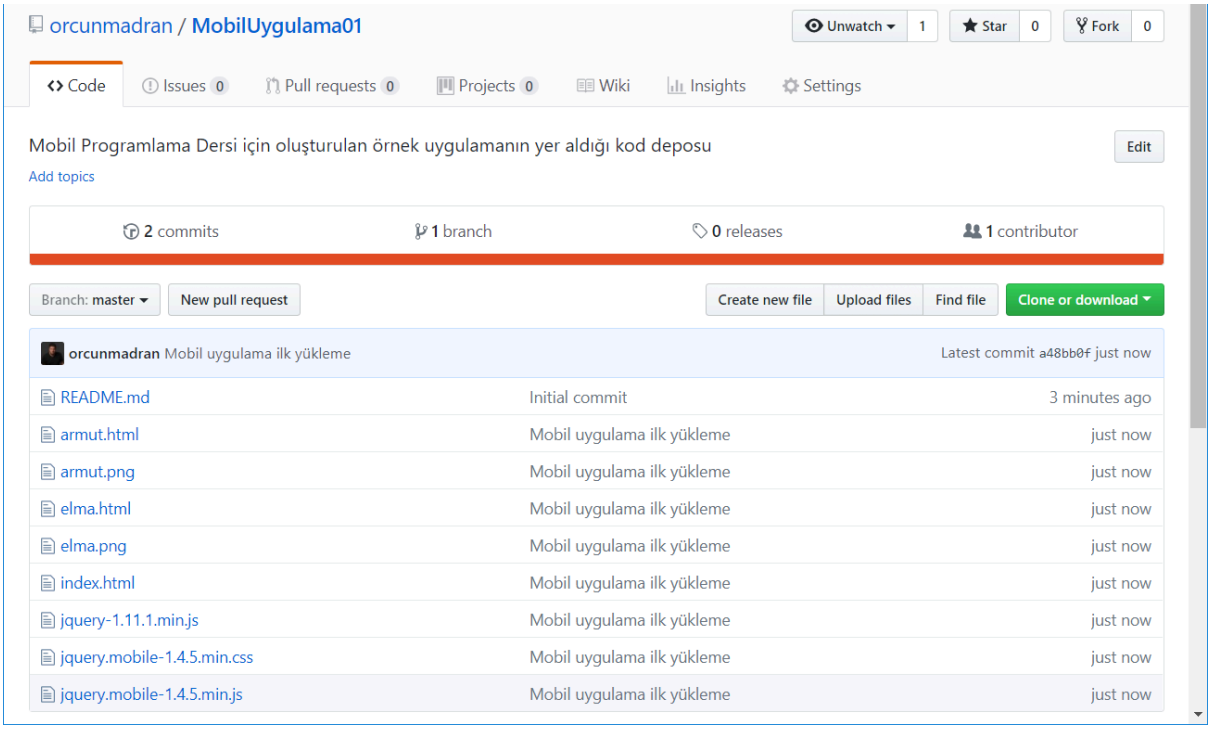
Şekil 51. Değişikliklerin kod deposuna deposuna teslim edilmesi.

Şekil 51'de yer alan forma yükleme işlemi ile ilgili kısa bir bilgi girdikten sonra "Commit changes" (değişiklikleri teslim et) düğmesine tıklandığında kod deposuna uygulama dosyalarının gönderilme işlemi başlatılır. İnternet bağlantısının hızına bağlı olarak dosyaların gönderilme işlemi bir süre devam eder (Şekil 52).



Şekil 52. Dosyaların gönderilme süresi.

Dosyalar kod deposuna teslim edildiğinde tüm uygulama bileşenleri Şekil 53'deki gibi görüntülenir. Kodlarda yapılacak herhangi bir değişiklik sonrası sadece değişiklik yapılan dosyanın yeniden kod deposuna gönderilmesi yeterlidir.



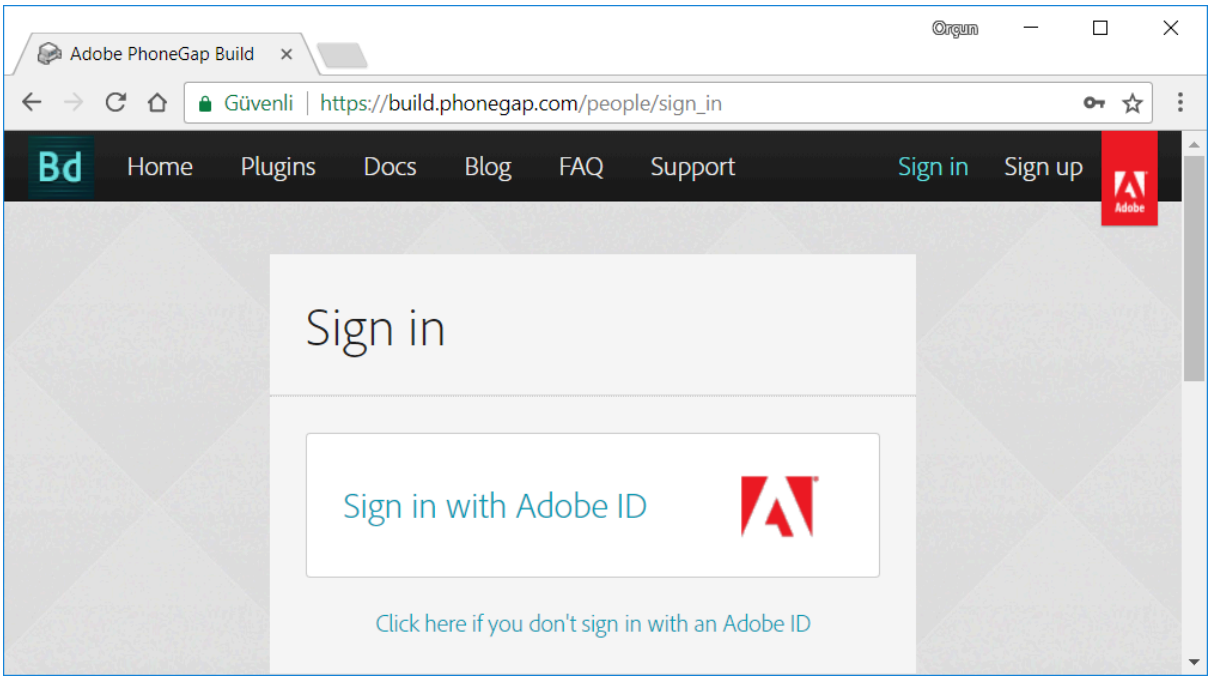
Şekil 53. Kod deposunun dosyalar teslim edildikten sonraki görünümü.

Bu ders kapsamında geliştirilen "MobilUygulama01" adlı örnek uygulamaya GitHub üzerinden aşağıdaki adres yardımıyla ulaşabilirsiniz. Dosyaları ve içeriklerini görüntüleyebilir, kendi kişisel bilgisayarınıza indirebilirsiniz:

https://github.com/orcunmadran/MobilUygulama01

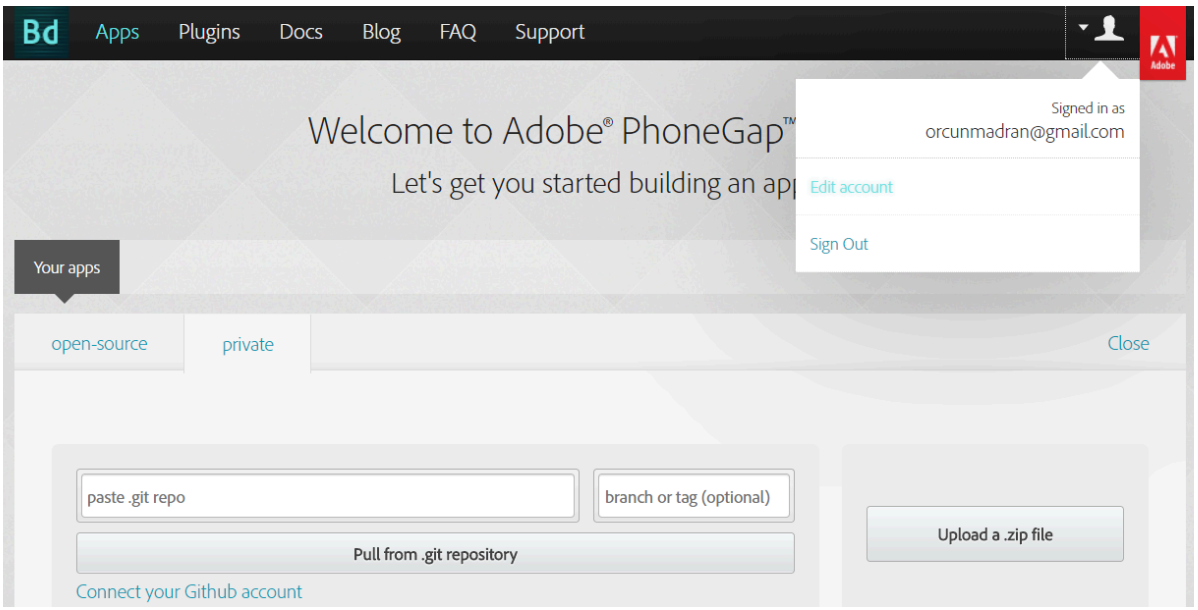
2.3.6.3 Paketleme İşleminin Gerçekleştirilmesi için PhoneGap Build Platformunun Kullanılması.

GitHub kod deposuna uygulama dosyalarının yüklenmesi ile paketleme işleminde kullanılacak dosyalar için yapılan ön hazırlık tamamlanmış olur. Bu aşamada paketleme işleminin gerçekleştirileceği PhoneGap Build (https://build.phonegap.com/) platformuna ücretsiz oluşturulan Adobe ID (Adobe Kimliği) ile giriş yapılır (Şekil 54).



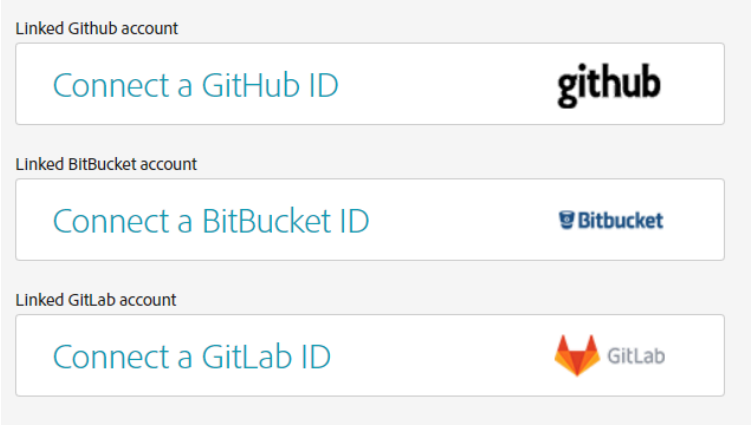
Şekil 54. Adobe ID ile PhoneGap Built platformuna giriş yapılması.

Giriş yapıldıktan sonra sağ üst köşede bulunan platform ile ilgili hesap ayarlarının düzenlendiği bölümden “Edit account” (Hesabı Düzenle) öğesi seçilerek GitHub ile bağlantının kurulacağı sayfa görüntülenir (Şekil 55).



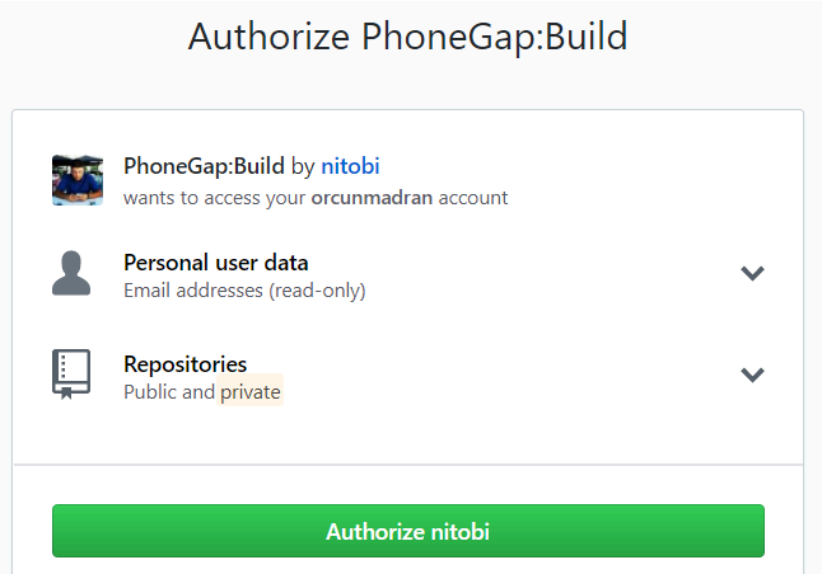
Şekil 55. GitHub ile bağlantının kurulacağı sayfanın görüntülenmesi.

Şekil 56'da görüntülenen sayfadan bağlantı kurulmak istenen platform seçilir. Mobil uygulamamız ile ilgili dosyalar GitHub kod deposunda olduğu için Github platformunun üzerine tıklayarak sonraki aşamaya geçilir.



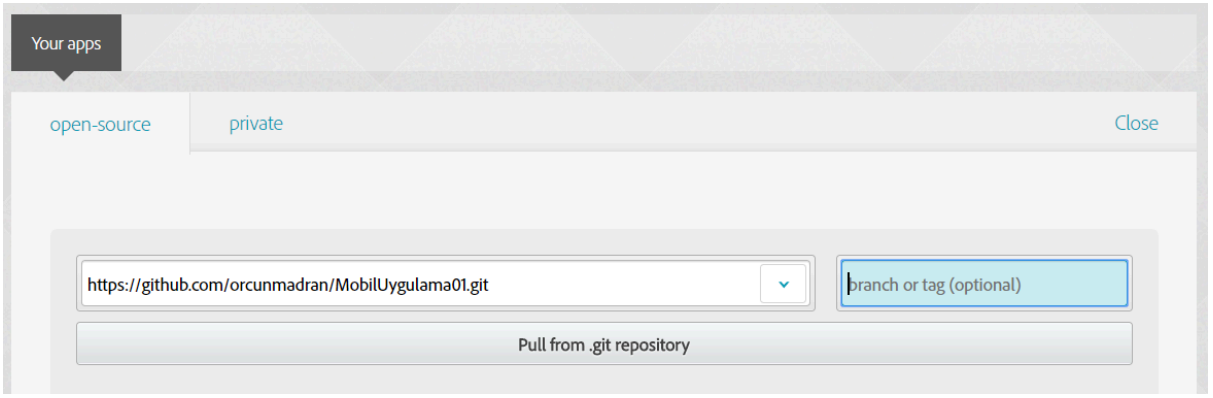
Şekil 56. Bağlantı kurulacak platformun seçilmesi.

Seçim işleminden sonra PhoneGap Build ve Github platformları arasında bağlantı kurulmasına ilişkin iletişim sayfası görüntülenir ve bu sayfada işleme onay verilmesi istenir (Şekil 57).



Şekil 57. Bağlantının kurulabilmesi için onay verilmesi.

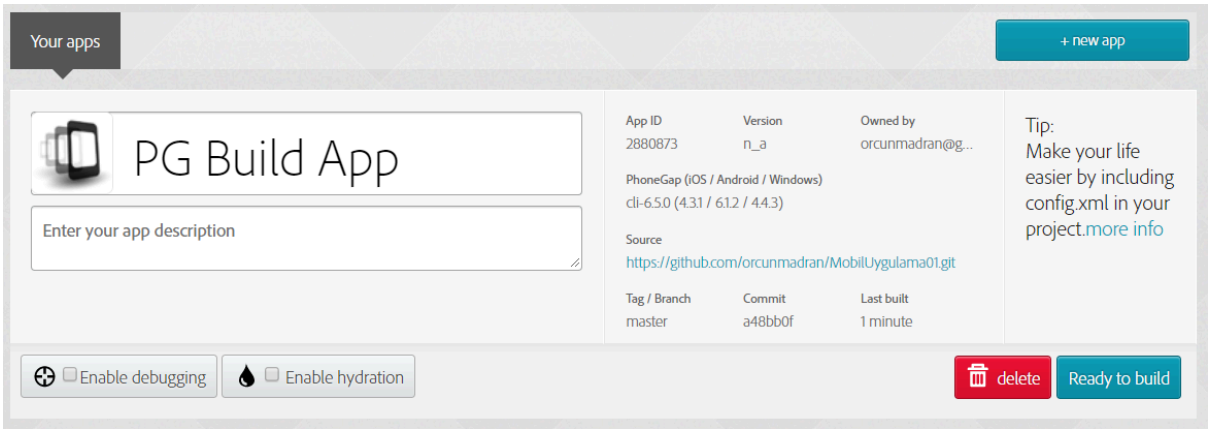
Onay işlemi verildikten sonra PhoneGap Build platformuna otomatik olarak geri dönüldüğünde "Your GitHub account has been linked successfully." (GitHub hesabınız ile başarılı bir şekilde bağlantı kuruldu.) mesajı görüntülenir. PhoneGap Build ana sayfasına dönüş yapıldığında "Your apps" (Uygulamalarınız) altındaki "open-source" (açık-kaynak) sekmesine tıklanır ve "find existing repo / paste .git repo" (varolan depoyu bul) açılır kutusundan GitHub'daki ilgili kod deposu seçilir (Şekil 58).



Şekil 58. GitHub kod deposunun seçilmesi.

Kod deposunun seçilmesinin ardından "Pull from .git repository" (git deposundan çek) butonuna tıklanarak GitHub'da yer alan uygulama dosyalarının paketleme işlemi için PhoneGap Build'e aktarılması sağlanır. Dosyaların aktarılma işlemi

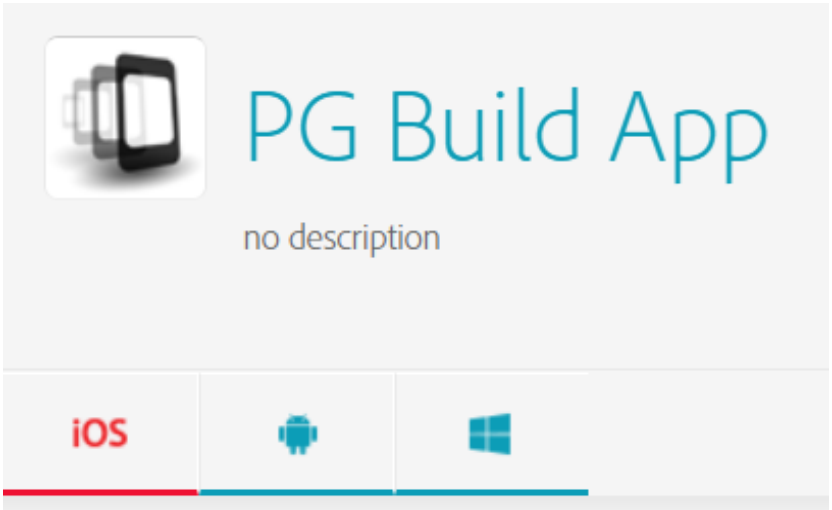
tamamlandıktan sonra uygulama ile ilgili genel bilgilerin yer aldığı sayfa görüntülenir (Şekil 59).



Şekil 59. PhoneGap Build'e aktarılan uygulama ile ilgili genel bilgilerin yer aldığı sayfa.

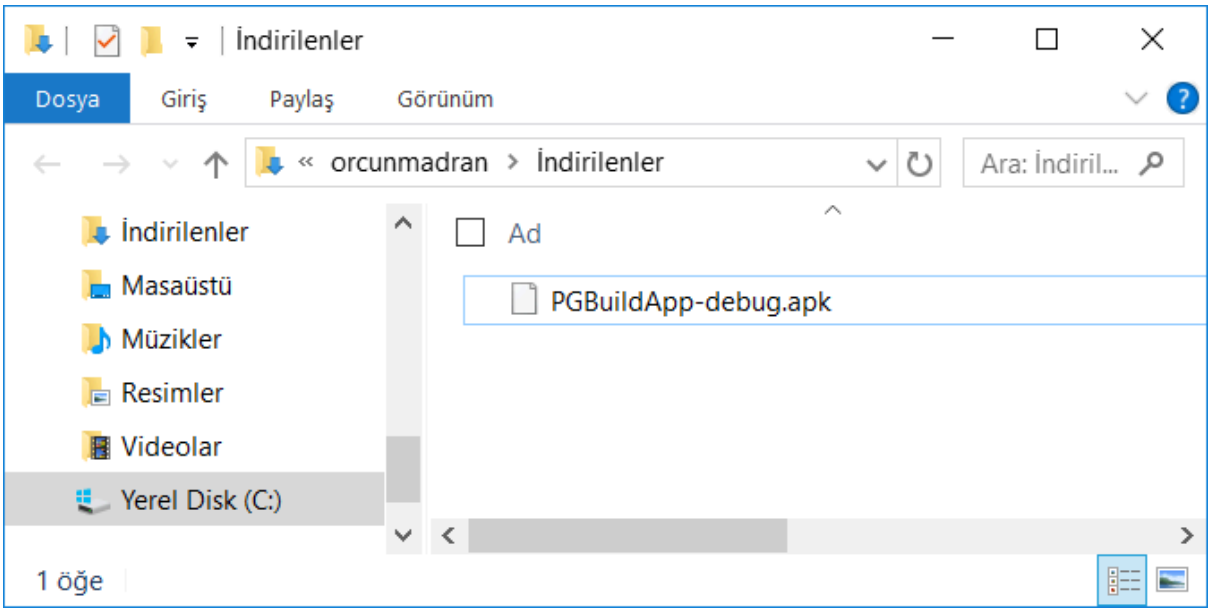
Şekil 59'da görüntülenen sayfa içerisinde uygulamanın adı değiştirilebilir, kısa bir açıklama eklenebilir ve "Ready to build" (Paketlemeye hazır) düğmesine tıklanarak mobil cihaza uygulamanın yüklenebilmesi için "APK" uzantılı dosya oluşturulabilir.

Paketleme işlemi tamamlandığında Şekil 60'da yer alan platform ikonlarına tıklandığında ilgili platformun kurulum dosyası (ör: *.apk) tarayıcı yardımıyla kişisel bilgisayara indirilir.



Şekil 60. Kurulum dosyasının indirilmesi.

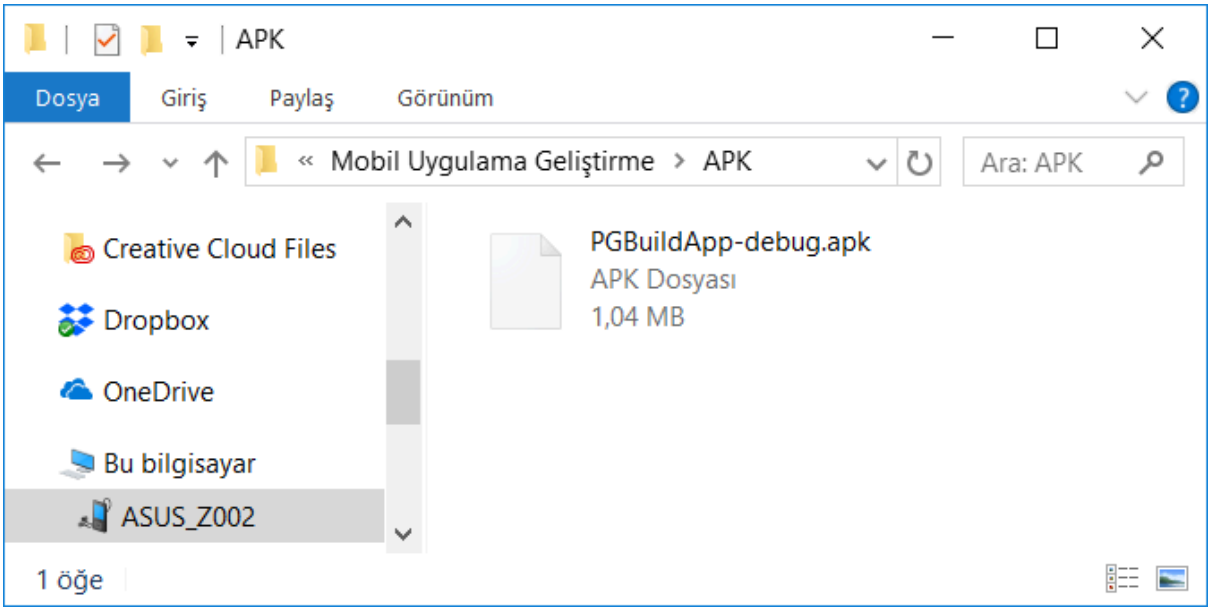
Platform ikonlarından ortada bulunan Android ikonuna tıklandığında “PGBuildApp-debug.apk” kişisel bilgisayara indirilir. Bu dosyayı Dosya Gezgini yardımıyla bulup mobil cihazınızın içine yükleyerek buradan kurulumu gerçekleştirilir ve mobil cihaz üzerinde kullanıma hazır hale gelir (Şekil 61).



Şekil 61. İndirilen kurulum dosyasına Dosya Gezgini ile erişilmesi.

2.3.6.4 APK Uzantılı Kurulum Dosyasının Mobil Cihaza Aktarılması ve Kurulumun Gerçekleştirilmesi

APK uzantılı kurulum dosyasını mobil cihaza aktarmak için mobil cihazın USB kablosunu kişisel bilgisayara takmak yeterli olacaktır. Bağlantı sağlandıktan sonra APK uzantılı dosya daha sonra mobil cihaz üzerinden erişimin kolay olabileceği bir lokasyona kopyalanır (Şekil 62).

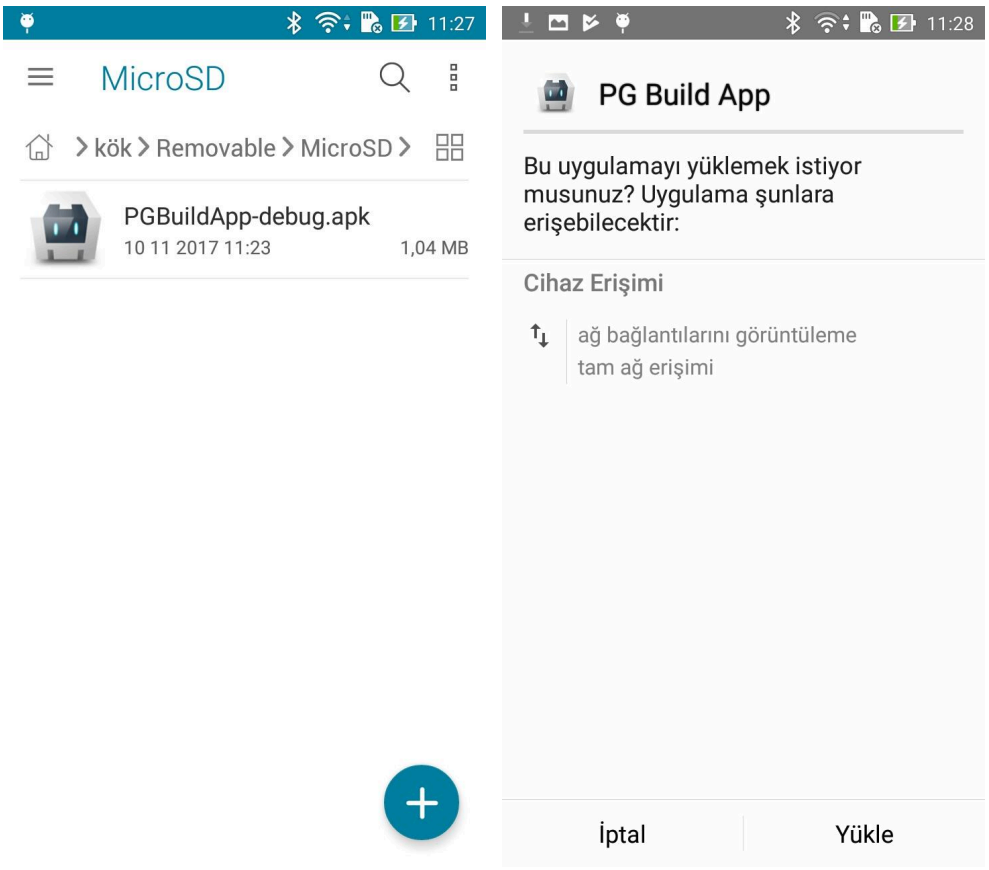


Şekil 62. APK uzantılı kurulum dosyasının mobil cihaz içerisine aktarılması.

APK uzantılı kurulum dosyası mobil cihaza aktarıldıktan sonra mobil cihazdaki dosya yöneticisi ya da benzeri bir program yardımıyla dosyaya ulaşılır ve çalıştırılır. Bu aşamada dikkat edilmesi gereken nokta, resmi uygulama marketi dışından bir kurulum dosyasının mobil cihazda çalışması için gerekli izinlerin verilmesi gerekliliğidir.

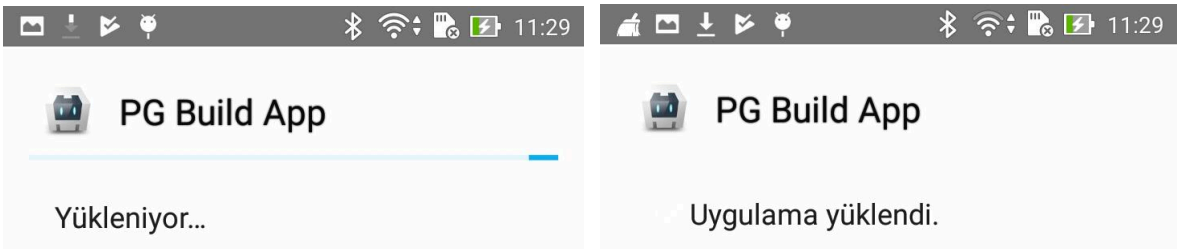
Geliştirici seçenekleri olarak tanımlayabileceğimiz bir dizi özelliğin kontrolüne mobil cihazdaki “Ayarlar” altında yer alan “Geliştirici Seçenekleri” bölümünden ulaşılır.

Kurulum dosyası çalıştırıldığında “Bu uygulamayı yüklemek istiyor musunuz? Uygulama şunlara erişebilecektir:” şeklinde bir onay bölümü mobil cihazda görüntülenir. Ekranın sağ alt tarafında “Yükle” butonuna tıklayarak uygulamanın mobil cihaza kurulumu başlatılmış olur (Şekil 63).



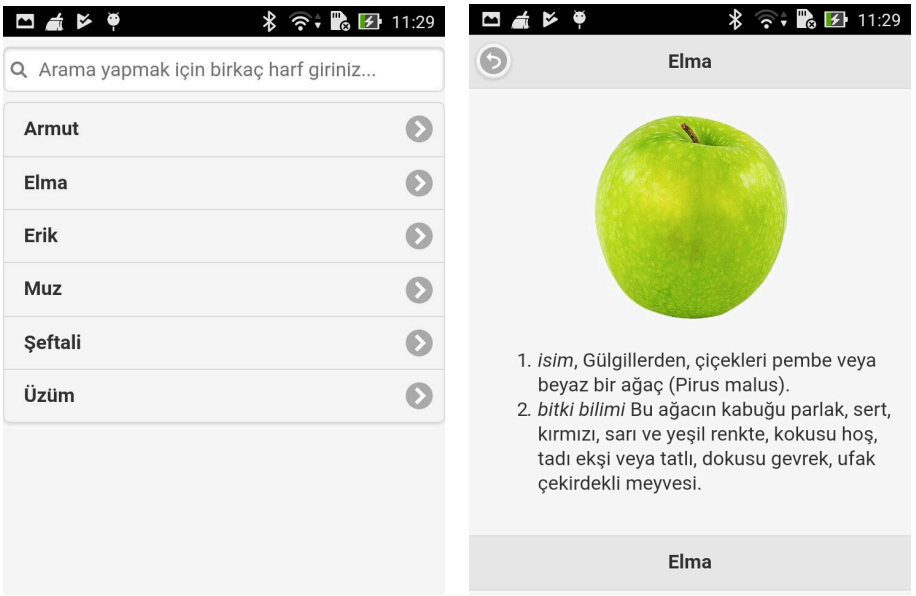
Şekil 63. Kurum dosyasına erişim ve yükleme işleminin başlatılması.

Uygulamanın boyutuna ve mobil cihazın işlemci gücüne bağlı olarak yükleme işlemi belli bir süre alacaktır. Yükleme işlemi tamamlandığında “Uygulama yüklendi.” mesajı ekranda görüntülenir (Şekil 64).



Şekil 64. Uygulamanın yüklenme süreci ve yükleme işleminin tamamlanması.

Uygulama artık program listesi içerisinde yer almaktadır. Uygulama çalıştırıldığında Şekil 65'deki gibi görüntülenir.



Şekil 65. Mobil uygulamanın cihazda görüntülenmesi.

Faydalanılabilecek Çevrim İçi Kaynaklar

Adobe PhoneGap ana portalı: https://phonegap.com/
Adobe PhoneGap Build: https://build.phonegap.com/
Adobe PhoneGap Emulate: http://emulate.phonegap.com/
Android platformu ana portalı: https://www.android.com/
Android platformu geliştirici portalı: https://developer.android.com/
Android platformu için tasarım çalışmaları: https://developer.android.com/design
Android platformu için geliştirme çalışmaları: https://developer.android.com/develop
Apache Cordova Projesi: https://cordova.apache.org/
Brackets Metin Editörü: http://brackets.io/
GitHub kod deposu: https://github.com/
HTML5 eğitim ve referans portalı: https://www.w3schools.com/html/default.asp
iOS platformu geliştirici portalı: https://developer.apple.com/ios/
JQuery JavaScript kod kütüphanesi: http://jquery.com/
JQuery Mobile kod kütüphanesi: http://jquerymobile.com/
JQuery Mobile uygulama örnekleri: http://jquerymobile.com/demos/
Windows Phone geliştirici portalı: https://developer.microsoft.com/en-us/windows