Düzce Üniversitesi
Siber Güvenlik ve Dijital Ekonomi

Düzce University
Journal of Cybersecurity and Digital Economics
https://dergipark.org.tr/tr/pub/cdej

Araştırma Makalesi/Research Article

iOS Mobil Uygulamalarda Performans, Veri Güvenliği ve Gizlilik Testleri: Çok Kullanıcılı Uygulama Geliştirme

Enver Küçükkülahlı[a,*], *Ezgi Kara Timuçin*[a], *Yasin Türkyılmaz*[a], *Mohamad Khoja*[a]

[a]Düzce Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, Düzce, Türkiye.
[*]**Sorumlu Yazar/Corresponding author:** enverkucukkulahli@duzce.edu.tr

Makale Bilgileri/Article Information:
Geliş/Received: 16/05/2025, Revizyon/Revision: 22/06/2025, Kabul/Accepted: 25/06/2025.

ÖZET

Mobil uygulamalar giderek popülaritesi artan bir alandır. Artan popülarite ise mobil uygulama testlerini daha önemli ve işlevsel hale getirmektedir. Mobil yazılım testleri geliştirilen uygulamaların doğru, tutarlı çalışıp çalışmadığının kontrolü ve kullanıcı beklentilerinin ne düzeyde karşılandığını belirlemek için kullanılan test prosedürlerini içerir. Literatürde mobil yazılım testleriyle ilgili odak noktası Android platformu üzerinedir. iOS mobil uygulama testleri ise literatürde eksik kalan çok fazla çalışılmamış alanlardandır. Yapılan çalışma, ios platformunda geliştirilen çok kullanıcılı mobil uygulama geliştirilmesini ve bu uygulamaların testini içermektedir. Geliştirilen mobil yazılım uygulamasının farklı koşullar altında çalışmasını test edebilmek için, test senaryolarının manuel ve otomatik testlerini içermektedir. Ayrıca performans ve veri güvenliği ve gizliliği testleri uygulanmaktadır. Elde edilen sonuçlar değerlendirilerek iOS yazılım geliştirme konusunda geliştiricilere tavsiyeler sunulmuş ve literatürdeki iOS mobil yazılım test konusunda literatürde eksik kalan kısımları tamamlanmıştır.

Anahtar Kelimeler: *iOS test, Mobil Uygulama Testi, Yazılım Testi, Veri Güvenliği*

Performance, Data Security and Privacy Testing in iOS Mobile Applications: Multi-User Application Development

ABSTRACT

Mobile applications are an area of increasing popularity. This increasing popularity makes mobile application testing more important and functional. Mobile software testing includes test procedures used to check whether the developed applications work correctly, and consistently and to determine how well they meet user expectations. In the literature, the focus on mobile software testing is on the Android platform, while iOS mobile application testing is one of the most understudied areas in the literature. This study involves the development and testing of multi-user mobile applications developed on the iOS platform. It includes manual and automated testing of test scenarios to test the operation of the developed mobile software application under different conditions. In addition, performance and data security and privacy tests are applied. By evaluating the results obtained, recommendations are presented to developers on iOS software development and the missing parts of the literature on iOS mobile software testing in the literature are completed.

Keywords: *Data Security, iOS testing, Mobile Application Testing, Software Testing,*

Atıf için: Küçükkülahlı, E., Kara Timuçin E., Türkyılmaz Y. & Khoja M. (2025). iOS Mobil Uygulamalarda Performans, Veri Güvenliği ve Gizlilik Testleri: Çok Kullanıcılı Uygulama Geliştirme. *Düzce Üniversitesi Siber Güvenlik ve Dijital Ekonomi Dergisi* 1(1): 43-55.

I. GİRİŞ

Mobil uygulamaların popülerliği her geçen gün artmaktadır. Günümüzde, mobil uygulamaların hızlı bir şekilde gelişen dünyasında, yazılım testleri giderek daha karmaşık ve önemli hale gelmektedir. Bu sebeple mobil uygulamalar geliştirilirken güvenlik ve kaliteyi sağlayan yöntemlere ihtiyaç duyulmaktadır. Yazılım test süreçleri yazılım yaşam döngüsünün önemli bir parçasıdır. Bu durum mobil yazılımlar için yapılan testlerin önemini her geçen gün arttırmaktadır. Bir mobil uygulama test edilerek mobil uygulama geliştiricilerine gerçek dünya simülasyonu ve farklı koşullarda uygulamayı test etme olanağı tanır. Test süreci uygulamaların performansını arttırmada, hata tespitinde ve kullanıcı deneyimleri hakkında fikir edinmede önemli bir role sahiptir. Ayrıca mobil uygulama testi kullanıcı beklentilerinin değişimine ayak uydurabilmek için kritik ve stratejik bir öneme sahiptir. Bu sebeple uygulamalara özgü deneysel test ortamlarının geliştirilmesi gerekmektedir. Bu kapsamda yapılan manuel testler test sürecini uzamasına sebep olmaktadır. Testlerin etkin bir şekilde gerçekleştirilebilmesi için test otomasyonlarının kullanılması verimliliği arttırmaktadır.

Yazılım test süreci, sistemin güvenilirliğini ve işlevselliğini sağlamak amacıyla çeşitli test türlerini içermektedir. Bu kapsamda, entegrasyon testleri, farklı modüllerin birlikte uyum içinde çalışıp çalışmadığını değerlendirmeye yönelik olup sistem bileşenlerinin etkileşimini sınar (Myers, Sandler & Badgett, 2011). Kabul testleri, geliştirilen yazılımın kullanıcı gereksinimlerini karşılayıp karşılamadığını belirlemek amacıyla yapılır ve genellikle müşteriler veya son kullanıcılar tarafından yürütülür (Pressman & Maxim, 2014). Gelişimin ileri safhalarında uygulanan beta testleri, sınırlı kullanıcı grubu tarafından gerçek kullanım ortamında gerçekleştirilerek, yazılımın yaygın dağıtımdan önceki son değerlendirmesini sağlar (Kaner, Falk & Nguyen, 1999). Otomatik testler, tekrarlanabilir ve tutarlı sonuçlar elde etmek amacıyla yazılım araçlarıyla yürütülürken, performans testleri, sistemin hız, tepki süresi ve kaynak kullanımı gibi niteliklerini ölçerek, yüksek yük altında gösterdiği davranışları analiz eder (Jorgensen, 2013). Bu test türlerinin birlikte uygulanması, yazılımın kalitesini güvence altına alırken, potansiyel hataların erken aşamalarda tespit edilmesine de olanak tanımaktadır.

Mobil yazılım test süreci birim testleri ile başlamaktadır. Entegrasyon testleri, kabul testleri, beta testleri, otomatik testler ve performans testleriyle tamamlanmaktadır. Doğru test stratejileri kullanarak uygulamayı farklı senaryolarda, farklı cihazlarda ve gerçek kullanım koşullarında test etmek önemlidir. Bu amaçla test ortamları incelenerek alt yapıya en uygun test aracı seçilmiştir. İncelenen test ortamları şu şekilde sıralanabilir. TestFlight, Xcode, Appium bu ortamlardan birkaç tanesidir. Yapılan araştırmalar sonucunda, uygulamaları test etmek için en uygun simülatörün Xcode olduğuna karar verilmiştir. Xcode simülatörünün seçim sebebi ise şunlardır:

- Farklı Cihazları Taklit Etme
- Kolay Erişim
- Hızlı Test Döngüsü
- Hata Ayıklama
- Ekran Görüntüsü Alımı
- Sensör Taklitleri

Xcode Simulator, iOS mobil uygulamaları test etmek için kullanılabilecek güçlü bir araçtır. Uygulamanın farklı cihazlarda, farklı senaryolarda ve gerçek kullanım koşullarında nasıl davrandığını test etmek ve olası hataları tespit etmek için Xcode Simulator'ü etkili bir şekilde kullanabilir.

iOS mobil yazılım testleri, iOS platformunda geliştirilen mobil uygulamaların doğruluğunu, kalitesini ve kullanılabilirliğini sağlamak amacıyla yapılan testlerdir. Geliştirilen projede iOS platformunda mobil uygulamalar geliştirilerek, bu uygulamalar için deneysel ortam hazırlanmış ve Xcode version 14.3.1 platformunda testleri yapılmıştır.

Mobil yazılım testi, mobil uygulamaların doğru, güvenilir ve hatasız bir şekilde çalışmasını sağlamak için yapılan bir süreçtir. Bu testler, uygulamanın farklı cihazlarda ve işletim sistemlerinde sorunsuz bir şekilde çalışmasını sağlamayı hedefler. Mobil uygulamalar olağanüstü bir şekilde benimsenirken, mobil uygulamaların doğrulanması ve onaylanması için herhangi bir özel test yaklaşımını için bir standart oluşturulmamıştır. H.

Muccini ve arkadaşları yapmış oldukları çalışmada, üç araştırma sorusunu yanıtlayarak mobil uygulama test otomasyonuna ilişkin yeni araştırma yönlerini keşfetmek istemektedir: Farklı ve uzmanlaşmış yeni test teknikleri gerekli midir? Mobil uygulamaların test edilmesindeki yeni zorluklar ve araştırma yönleri nelerdir? ve mobil uygulamaların test edilmesinde otomasyonun oynayabileceği rol nedir? Sorularına cevap aramışlardır (Muccini et al., 2012). V Kartkikeyani yapmış olduğu çalışmada, mobil uygulamaların önemine ve artan kullanımına odaklanmaktadır ve mobil uygulama test etme stratejilerini ele alınmıştır. Mobil cihazlarda ve kablosuz ağlarda, mobil uygulamaların sorunsuz ve güvenli bir şekilde çalışmasının ticari başarı için kritik olduğunun altı çizilmiştir. Mobil uygulamaların hatalarını belirlemek ve düzeltmek, kolay ve güvenli bir kullanıcı deneyimini sağlamak, uygulamanın tutarlı çalışmasını garanti altına almak, güvenlik açıklarını tespit etmek ve uygulama performansını optimize etmek için gerekli olduğunu belirtilmiştir. Mobil uygulama testi, uygulamanın büyümesini ve yayılmasını sağlamak için de önemli olduğunu belirtilmektedir (Selvam, R., & Karthikeyani, 2011).

D. Amalfitano ve ark. yapmış oldukları bir çalışmada, Android platformu için geliştirilen mobil uygulamaların otomatik testi için bir yöntem sunar. Bu yöntem, test edilecek uygulamanın GUI (Graphical User Interface - Grafiksel Kullanıcı Arayüzü)' sini otomatik olarak oluşturan ve otomatik olarak yürütülebilecek test durumlarını elde eden bir 'crawler' üzerine dayanmaktadır. Android Mobil Uygulama Testi için “Tırmanma Tabanlı Teknik" adıyla adlandırılır. Bu yöntem, testlerin yürütülmesinden önce uygulamanın otomatik olarak taranmasını içerir. Taranan bilgiler, Java test metotlarına dönüştürülerek, testlerin yürütülmesinde kullanılır. Çalışma sonucunda bu tırmanma tabanlı teknik kullanılarak Android mobil uygulamalarının otomatik testlerinin gerçekleştirilebileceği ve yeni bulunan hataların otomatik olarak tespit edilebileceğidir. Bu teknik, robotik tabanlı test araçları kullanarak Android uygulamalarının testini kolaylaştırmakta ve test sürecinin sürdürülebilirliğini sağladığı belirtilmektedir (Amalfitano et al., 2011).

Ivans K. ve ark. yapmış oldukları çalışmada, fonksiyonel testleri etkileyen yönleri, mobil uygulamalar genelinde, araştırmaktadır ve genelde zaman kısıtlamalarından dolayı genellikle göz ardı edilen güvenlik kabiliyetlerini ve risklerini gözler önüne sermektedir. Ayrıca, mobil UI test otomasyon araçlarının incelenmesi ve gruplandırılması gerçekleştirilmiştir. Çalışmada, mobil uygulama testlerinin kalite özellikleri incelenmiştir. Bunlar; fonksiyonel uygunluk, performans verimliliği, uyumluluk, güvenilirlik, bakım kolaylığı ve taşınabilirlik olarak belirlenmiştir. Ayrıca, ISO (2011) bu kalite özelliklerini belirlemektedir. Kullanılabilirlik testleri kapsam dışında bırakılmıştır (Kuļešovs et al., 2018).

T. ÇİLOĞLU ve ark. yapmış oldukları çalışmada mobil uygulama geliştirme hakkında bilgiler vermişlerdir. Çalışmanın bir kısmında ise mobil uygulama testlerinden bahsedilmiş ve şöyle denilmiştir; Mobil uygulama testleri için otomatik testler gerçek cihazlarda çalıştırılır. Bu testler hizalama sorunlarını, güvenlik açıklarını ve diğer problemleri tespit edebilir. Firebase Test Lab'te ise özelleştirilmiş testler uygulanarak kalite ile ilgili sorunlar daha ayrıntılı bir şekilde araştırılır. Bu, uygulamanın görsel olarak daha etkili olmasını ve Google Play gibi mağazalarda daha hızlı eklenmesini ve ekonomik getiri sağlamasına olanak tanır. Devamında Android için takip edilmesi gereken adımları belirtmişlerdir (Çiloğlu et al., 2021).

I. Kulesovs'un yapmış olduğu çalışmada yerel iOS uygulamalarının testini etkileyen yönleri (yani özellikleri ve/veya sınırlamaları) belirlemeyi amaçlamaktadır. Çalışma, iOS uygulamalarının test edilmesiyle ilgili olarak akademik dünyada var olan boşluğu ortadan kaldırıyor. Uygulayıcılar ayrıca iOS uygulama test stratejilerini belirlenen yönleriyle yerine getirmeye teşvik ediyor. Yazar, çalışmanın, iOS uygulama testi yönlerinin tanımlanması ve ayrıntılı açıklaması konusunda akademik dünyada var olan boşluğu ortadan kaldırdığı sonucuna varmıştır. Bu ayrıntılar, iOS test stratejilerini daha sağlam ve eksiksiz hale getirmek isteyen uygulayıcılar için de yararlı olması gerektiğidir (Kulesovs, 2015).

S. Zein ve ark. yapmış oldukları çalışmada mobil uygulama yazılım mühendisliği alanında sistematik incelemeler ve sistematik haritalama çalışmalarından oluşan üçüncül bir çalışma yürütmüşlerdir. 24 ikincil çalışma belirlemişlerdir. İkincil çalışmalar, efor tahmini, kalite güvencesi ve test otomasyonu, kullanılabilirlik,

yazılım geliştirme modelleri, gereksinim mühendisliği, mimari tasarım, makine öğrenimi kullanarak kusur tahmini, bulut tabanlı geliştirme ve model odaklı geliştirme gibi çok sayıda araştırma alanını kapsar. Mobil uygulama geliştirme odağındaki eğilimi araştırırken, son yıllardaki eğilimin mimari tasarım, işlevsel olmayan gereksinimlerin test edilmesi ve bağlama duyarlı testler gibi daha spesifik yazılım mühendisliği konularına doğru ilerlediğini görülmüştür (Zein et al., 2023).

P.H. Kuroishi ve ark. yapmış oldukları çalışma ikincil çalışmaları gözden geçirerek mobil uygulama testine genel bir bakış sunmayı amaçlamaktadır. Mobil uygulama testi ile ilgili farklı araştırma konularını ele alan 21 ikincil çalışma ele almışlardır. Bu makaleler dört ana gruba ayrılmıştır: Genel, Otomasyon, Bağlam Farkındalığı ve Çevre. Ardından, her çalışmanın ele aldığı belirli araştırma konusuna dayanarak çalışmaları özetlenmiştir. Çoğu deneysel çalışma araştırmalarını gerçekleştirmek için Android'e dayanmaktadır ve iOS uygulama testi ile ilgili çalışmaların eksikliğini göstermektedir. (Kuroishi et al., 2023). L. Li ve ark. yapmış oldukları çalışmada Android uygulamalarının statik analizini inceleyen bir literatür taraması yapmaktadır. Araştırma, 124 adet yazılım mühendisliği, programlama dilleri ve güvenlik alanlarında yayınlanan makaleyi incelemektedir. Makalede, statik analiz yaklaşımlarının trendleri, odak noktaları ve gelecekteki araştırmalar için gereken önemli konular belirtilmektedir. Bu araştırmanın sonuçlarına göre, Android uygulamalarının statik analizinde en çok odaklanılan konular güvenlik sorunlarıdır. Araştırmada yapılan incelemelerde, güvenlik açıklarını tespit etmek için statik analiz yaklaşımlarının kullanıldığı görülmüştür. Bu bulgu, diğer araştırmalarda da benzer şekilde ortaya çıkmıştır (Li et al., 2017).

R.Gyorodi ve ark. yapmış oldukları çalışmada Android ve iOS için geliştirilen uygulamalar arasındaki performans farkları incelenmiştir. Asenkron çoklu iş parçacığı yürütme, görüntülerin ekrana çizimi ve temel ağ iletişimi gibi üç temel sistem özelliği analiz edilmiştir. Her iki platformda da performans ölçümleri yapılmış ve sonuçlar karşılaştırılmıştır. Makalede, uygulama geliştirme için optimizasyon önerileri sunulmuştur. Çalışmada her iki platformda da performans ölçümleri yapılmış ve sonuçlar karşılaştırılmıştır. Yapılan testler, her iki platformun da farklı görevler için daha iyi performans gösterdiğini göstermektedir. Örneğin, iOS, ekrana çizilen görünümler için daha hızlıdır, ancak Android, ağ iletişimi için daha uygun olduğu görülmüştür (Gyorodi et al., 2017).

iOS uygulamaların geliştirilmesi ve test edilmesi konusunda çalışmalarla ilgili olarak: Knitza ve diğerleri. (2019), sağlık alanında mobil uygulamalarının bir analizini gerçekleştirdi ve bu uygulamaları hem iOS hemde Android platformlarında test etti. Kuzmik ve ark. (2023), kullanılabilirliğini artırmak için iOS tabanlı bir uygulamayı yinelemeli testler yoluyla test etti. Sunyaev ve ark. (2014), iOS'ta mobil sağlık uygulaması gizlilik politikalarının kullanılabilirliğini ve kalitesini değerlendirdi. Afjehei ve ark. (2019), Swift'de yazılmış iOS uygulamalarındaki performans sorunları üzerine deneysel bir çalışma ile iOS uygulamalarındaki performans sorunlarını tespit etmeye çalıştı. Murphy ve King (2016), iOS platformundaki uygulamaların, Android uygulamalarıyla karşılaştırdı. Ayrıca Martínez ve Lecomte (2017), iOS ve Android uygulamalarının işlevselliğini doğrulamaya yönelik bir test çerçevesi olan Xamarin UITest'i tartıştı. Wang ve diğerleri. (2018), mobil geliştiricilerin hem iOS hem de Android platformları için uygulamalar yayınlama eğiliminden bahsetti. Kousar ve ark. (2018), Objective-C ve iOS SDK'nın kullanımını vurgulayarak mobil uygulama geliştirmedeki zorlukları ve çözümleri özetledi. Choudhary ve ark. (2015), Android uygulamaları için otomatik test girişi oluşturmayı tartışarak iOS testi için potansiyel bir yön olduğunu belirtti. iOS geliştiricileri, otomatik test girişi oluşturma araçlarından yararlanarak test süreçlerinin verimliliğini ve etkinliğini artırabilir. Bu yaklaşım, çeşitli test senaryolarının oluşturulmasına, test kapsamının iyileştirilmesine ve potansiyel sorunların geliştirme döngüsünün başlarında belirlenmesine yardımcı olabilir. Ayrıca, (Dörr ve diğerleri, 2013) tarafından da vurgulandığı gibi, akıllı telefonların artan karmaşıklığı ve yetenekleri göz önüne alındığında, gelecekteki iOS mobil testleri, yalnızca işlevsel yönleri değil aynı zamanda görsellik gibi işlevsel olmayan yönleri de değerlendirebilen gelişmiş test araçlarının geliştirilmesine odaklanabilir. Ayrıca Qin ve ark. (2019), GUI test senaryolarının iOS'tan Android'e taşınmasına ilişkin, gelecekteki iOS mobil testlerinin, test senaryosunun taşınması ve farklı platformlarda yeniden kullanılmasına yönelik teknikleri keşfedebileceğini öne sürüyor. Bu

yaklaşım sayesinde, birden fazla platformda çalışan geliştiriciler için test çalışmalarını kolaylaştırabilir ve test uygulamalarında tutarlılığı destekleyebilir.

iOS mobil uygulamalarında otomatik test, model çıkarımı ve otomatik test girdisi oluşturma tekniklerinin entegrasyonundan faydalanabilir. Salva ve diğerleri (2015) mobil uygulamalar için gezinme yollarını ve uygulama durumlarını ifade eden resmi modellerin öğrenilmesini içeren bir otomatik test yöntemi önermiştir. Bu tür tekniklerden yararlanarak iOS geliştiricileri, çıkarılan modellere dayalı test senaryolarının oluşturulmasını otomatik hale getirebilir ve test süreçlerinin verimliliğini ve kapsamını artırabilir. Ayrıca Choudhary ve diğerleri (2015) tarafından yapılan araştırma, uygulama testlerinin otomatikleştirilmesinin önemini vurgulayarak son yıllarda test süreçlerinin otomatikleştirilmesine yönelik artan bir ilgiye işaret etmektedir. Bu durum, iOS mobil uygulama geliştirmede otomatik test uygulamalarının kullanılmasının önemini ve potansiyel etkisini vurgulamaktadır. Buna ek olarak, Muccini ve diğerleri (2012) tarafından yapılan çalışma, mobil uygulamaların test edilmesinde otomasyonun rolü hakkında önemli sorular ortaya atmış ve mobil uygulamaların benzersiz özellikleri nedeniyle özel test tekniklerinin ve otomasyonun gerekli olabileceğini öne sürmüştür. Bu durum, iOS mobil uygulama geliştirme bağlamında özel otomatik test yaklaşımlarına duyulan ihtiyacın altını çizmektedir. Sonuç olarak, iOS mobil uygulamalarında otomatik testin geleceği, mobil uygulamaları etkili bir şekilde test etmenin farklı zorluklarını ve gereksinimlerini ele almak için model çıkarımı, otomatik test girdisi oluşturma ve özel otomasyon tekniklerinin entegrasyonunu içerebilir.

Sonuç olarak, iOS mobil testinin geleceği, iOS uygulama testinin kalitesini ve verimliliğini artırmak için otomatik test girişi oluşturma tekniklerinin, işlevsel olmayan yönlerin değerlendirilmesine yönelik gelişmiş test araçlarının ve test senaryosunun platformlar arasında taşınması ve yeniden kullanılmasına yönelik stratejilerin entegrasyonunu içerebilir. Ayrıca iOS uygulama geliştirme ve test etme araştırmaları, iOS ve Android platformları arasındaki kullanılabilirlik farklarını, gizlilik politikası değerlendirmelerini, iOS uygulamalarındaki performans sorunlarını ve iOS uygulamalarının belirli işlevlerdeki üstünlüğünü derinlemesine incelemektedir.

II. MATERYAL METOT

Test senaryoları manuel ve otomatik testleri içermektedir. İlgili veri girişi, Şekil 1' de otomatik olarak yapılarak ilgili menüler kontrol edilmiştir.

Şekil 1. Otomatik test görüntüsü

A. Otomatik Test Aşamaları

Şekil 2'de gösterilen işlem otomatik bir şekilde testin yapıldığına dair görüntüdür. Gösterilen aşamaların hepsi otomatik olarak gerçekleşmektedir. İşlem text menüsüne tıklayarak 10 saniye bekledikten sonra klavye açılıp merhaba girdisini vermektedir. Ardından uygulamadan, ana ekrana geri dönüp *alert* menüsüne tıklayıp ekrana "bu bir uyarıdır" mesajını yazdırmaktadır. Böylelikle iOS cihazlarda otomatik yazma testi yapılmıştır.

Şekil 2. Otomatik test görüntüsü aşamaları

B. Geliştirilen Uygulama ve Testleri

Geliştirilen uygulama, X benzeri bir sosyal medya platformudur ve kullanıcılara kayıt, giriş, paylaşım yapma, diğer kullanıcıları arama ve profillerini düzenleme gibi özellikler sunmaktadır. Proje sürecinin analiz aşamasında, kullanıcıların beklentileri ve projenin hedefleri belirlenmiştir. İki farklı program için kullanıcıların nasıl etkileşime geçmek istedikleri ve hangi özelliklere ihtiyaç duydukları anlaşılmıştır. İki program için ayrı analiz süreçleri yürütülmüş ve kullanıcıların gereksinimleri temel alınarak tasarım aşamasına geçilmiştir. Proje gelişim süreci, analiz aşamasının ardından başlamış ve her iki programın da ayrı ayrı geliştirilmesini içermektedir. Swift programlama dili ve Xcode geliştirme ortamı kullanılarak her iki programın temelleri oluşturulmuştur. Yazılım geliştirme sürecinde, her aşamada düzenli olarak iç testler ve hata ayıklama işlemleri gerçekleştirilmiştir. Kullanıcı arayüzü tasarımı ve kullanılabilirlik testleri her iki program için özenle gerçekleştirilmiştir.

Proje tamamlandıktan sonra, program için kapsamlı testler gerçekleştirilmiştir. Performans, güvenlik ve kullanılabilirlik testleri sonuçlarına göre gerekli düzeltmeler ve iyileştirmeler yapılmıştır. Kullanıcıların geri bildirimleri dikkate alınarak her iki program da kullanıcı dostu hale getirilmiştir.

C. Sosyal Medya Uygulaması Detayları

Bu uygulama, kullanıcıların sanal bir sosyal platformda etkileşimde bulunmalarını sağlayan kapsamlı bir sosyal medya deneyimi sunmaktadır. İşte bu uygulamanın temel başlıkları şunlardır.

1.Ana Sayfa

Ana sayfa, kullanıcıların paylaştığı gönderileri görüntülediği merkezi kısımdır. Kullanıcılar burada arkadaşlarının ve diğer kullanıcıların güncel gönderilerini inceleyebilmektedir. Her gönderinin yanında, kullanıcılar gönderiyi beğenmekte veya beğenmeyebilmektedir. Ayrıca gönderi altında, kaç beğeni olup olmadığı kullanıcılar tarafından görüntülenebilmektedir.

2.Gönderi Paylaşma

Kullanıcılar, kendi gönderilerini oluşturup paylaşabilmektedir. Gönderiler metin, fotoğraf veya video içerebilir. Kullanıcılar paylaştıkları gönderileri diledikleri zaman silme yetkisine sahiptir. Bu durum, içeriklerin güncel ve kişiselleştirilmiş tutma açısından büyük önem arz etmektedir.

3.Gönderi Silme

Her kullanıcı, kendi gönderilerini silme yetkisine sahiptir. Bu durum, kullanıcıların gönderilerini istedikleri zaman kaldırabilmesini sağlamaktadır. Silinen gönderi ise anlık olarak sayfadan kaldırılmaktadır.

4.Kullanıcı Sayfaları

Kullanıcılar, diğer kullanıcıların profillerine erişebilme yetkisine sahiptir. Profil sayfalarında kullanıcının adı, profil fotoğrafı, paylaştığı gönderiler, kısa bir biyografi ve kişisel internet sayfası bilgisi yer almaktadır. Herkesin hesabı bütün kullanıcılara açıktır.

5.Hesap Silme

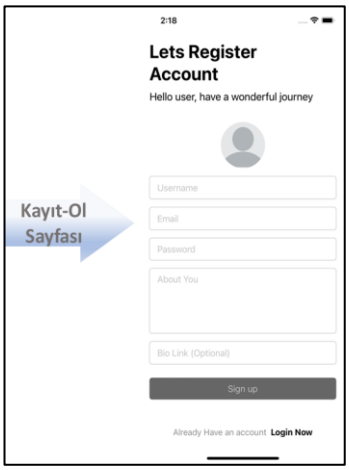
Kullanıcılar, hesaplarını istedikleri zaman silme yetkisine sahiptir. Hesap silindiğinde, kullanıcının tüm bilgileri ve gönderileri uygulamadan tamamen kaldırılır. Bu, kullanıcıların uygulamayı kullanmayı kalıcı olarak silmelerine olanak sağlamaktadır.

D. Geliştirilen Çok-Kullanıcılı Uygulamanın Tanıtılması

Kullanıcılar uygulamayı çalıştırdıklarında, ilk olarak giriş yapma zorunluluğu bulunmaktadır. Şekil 3' te görüldüğü üzere geçerli e-posta ve şifrelerini girerek, uygulamaya giriş işlemleri başarıyla gerçekleşmektedir.

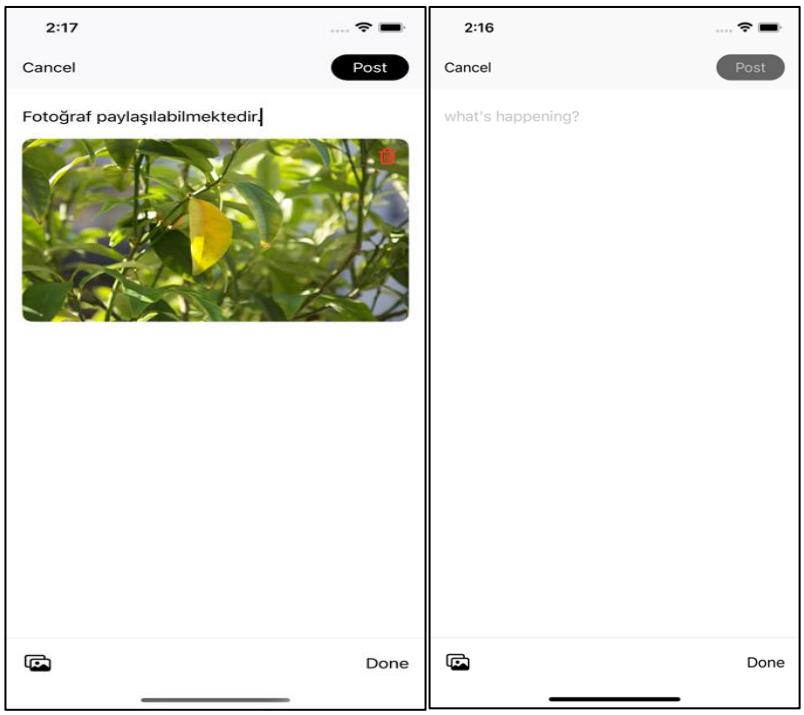
Şekil 3. Uygulama giriş sayfası

Eğer kullanıcının kayıtlı bir e-posta adresi yoksa, Şekil 4'teki *Register Now* sayfasına yönlendirilip kullanıcının kaydolması gerekmektedir. Kayıt işlemleri yapılırken doldurulan alanların boş geçilmemesi ve maillerin "@" işareti koymadan kayıt yapılmaması kullanıcıya uyarı olarak dönüş sağlamaktadır. Aynı zamanda kayıtlı işlemleri doğru bir şekilde tamamlandıktan sonra, alttaki *Login Now* kısmında bir önceki sayfaya başarılı bir şekilde geçiş yapılmaktadır. Bu giriş sayfasında hem e-posta adresi hem de şifre kontrolü sağlanmaktadır.



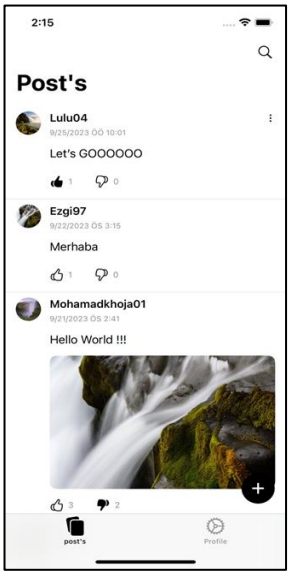
Şekil 4. Uygulamanın kayıt olma sayfası

Şekil 4 ‘te uygulamanın kaydolma sayfasına yer verilmiştir. Kullanıcılar başarılı şekilde kaydolduktan sonra, *Sign in* butonu ile uygulamaya giriş yapıp, kullanıcı gönderi işlemlerini başarılı bir şekilde tamamlayabilmektedir. Şekil 5.a. ve Şekil 5.b.’de görüldüğü gibi kullanıcı isterse hem fotoğraflı hem fotoğrafsız bir şekilde gönderi paylaşımı yapılabilmektedir.



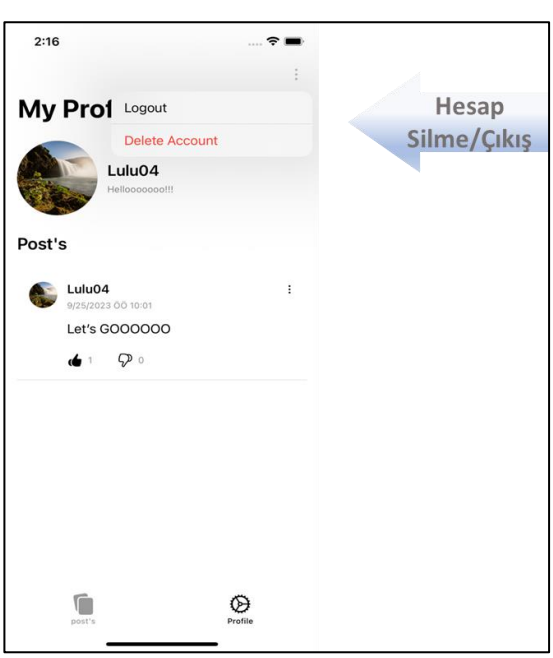
Şekil 5.a. Fotoğraflı gönderi paylaşımı. Şekil 5.b. Fotoğrafsız gönderi paylaşımı.

Kullanıcı kendisi gönderi paylaşabildiği gibi, diğer kullanıcıların da gönderilerini anlık olarak görüp beğenilmektedir. Aynı zamanda hoşuna gitmeyen gönderileri beğenmeyebilmektedir. Şekil 6’da farklı kullanıcıların paylaştığı gönderiler yer almaktadır. Gönderiyi paylaştığı tarih ve saat bilgisi de anlık olarak çekilmektedir. Beğeni ikonlarının aktif olarak çalıştığı aşağıda gösterilmiştir. Beğeni butonuna tıklandığında o kullanıcı, o ikonu siyah olarak görülmektedir.

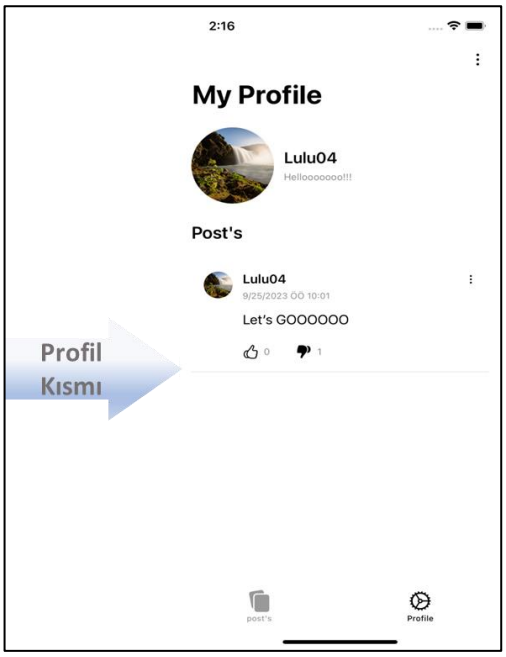


Şekil 6. Farklı kullanıcıların postları

Profilimize tıkladığımızda kullanıcının paylaştığı gönderiler görülmektedir. Bu gönderiler ne kadar beğeni almış, ne kadar kişi beğenmemiş hepsi bulunmaktadır. Anlık olarak kaydedilmektedir. Uygulamada farklı kullanıcılardan giriş yapabilme özelliği bulunmaktadır. Bunun için logout butonuna tıkladığımızdan direkt olarak bizi uygulamanın anasayfasına yönlendirmektedir. Uygulama çıkış işlemleri Şekil 7.a' da, profil sayfası ise Şekil 7.b.' de gösterilmiştir.

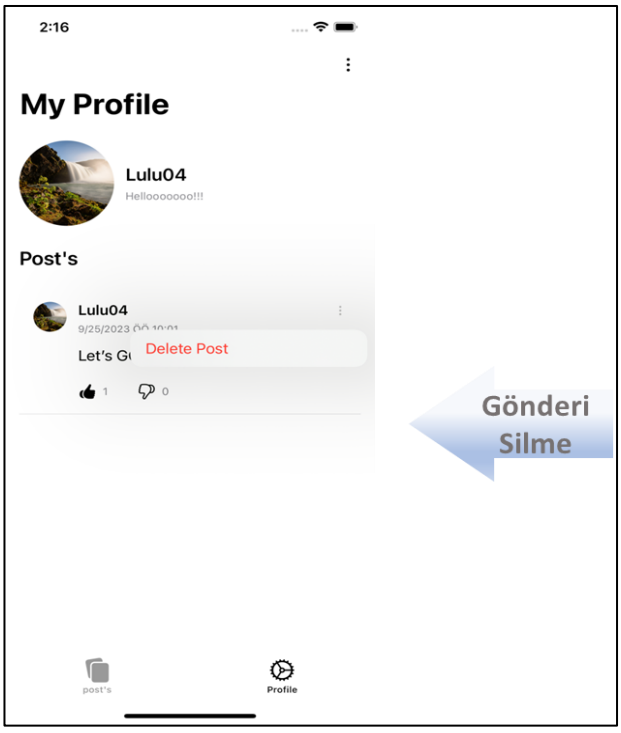


Şekil 7.a. Hesap silme işlemleri



Şekil 7.b. Profil sayfası görünümü

Profil sayfamızda yer alan gönderilerimizi paylaştıktan sonra silebilme özelliği Şekil 8'de gösterilmiştir.



Şekil 8. Gönderi silme ekranı

III. BULGULAR VE TARTIŞMA

Proje kapsamında geliştirilen mobil uygulamalar XCode platformunda test edilerek performans değerlendirmeleri yapılmıştır. Geliştirilen 13.0 sürümüne sahip iOS mobil uygulaması performans testine tabi tutulduğunda belirli bir işlemi tamamlaması için gereken süre 0.992 saniye olarak ölçüldü ve %2 oranında performans düşüşü gözlemlendi. Bu sonuca göre uygulamanın belirli bir fonksiyonunun veya genel işleyişinin, belirli şartlar altında performans kaybı yaşadığını göstermektedir. Geliştirilen uygulamada kullanıcı deneyimleri göz önüne alındığında 0.992 saniyelik işlem süresinin, kullanıcılar açısından az da olsa bir gecikme yaratabileceğini ve uygulamanın genel hız algısını olumsuz etkileyebileceğini göstermektedir. Gözlemlenen %2'lik performans düşüşü birden fazla sebepten kaynaklı olabilmektedir. Örneğin, bellek yönetimi, işlemci kullanımı veya optimize edilmemiş kod parçaları bu duruma sebep olabilmektedir. Bu veriler ışığında, uygulamanın daha verimli çalışması için performans optimizasyonu yapılması gerektiği sonucuna varılmıştır. Özellikle işlem süresini kısaltmak ve performans düşüşünün önüne geçmek için, uygulama kodu üzerinde çeşitli optimizasyonların yapılması gerektiği gözlemlenmiştir. Kod gözden geçirmeleri, gereksiz işlemlerin minimize edilmesi ve iOS 13.0 cihazlara özgü optimizasyonların gerçekleştirilmesi önerilmektedir. Parametreler mobil uygulama testlerinde kullanılmaktadır ve duration metriği, uygulamanın belirli bir işlevi veya işlemi gerçekleştirmesi için geçen süreyi ölçer. Bu metrik, uygulamanın performansını değerlendirmek ve iyileştirmek amacıyla kullanılmaktadır. Average (Ortalama): Uygulamanın bir işlevi veya işlemi gerçekleştirmek için ortalama ne kadar süre harcadığını ifade eder. Örneğin, bir sayfa yükleme işlemi için ortalama süre, kullanıcıların sayfanın tamamen yüklenmesini beklediği süredir. Baseline (Temel Çizgi): Baseline, uygulamanın mevcut performans seviyesini temsil eder. Bu, uygulamanın belirli bir işlemi ne kadar sürede gerçekleştirdiğini gösterir. Yeni bir güncelleme veya iyileştirme yapıldığında, bu temel çizgi ile karşılaştırılır. Metriklerin İzlenmesi: Uygulama performansını izlemek için bu metrikleri düzenli olarak takip etmek önemlidir. Böylece uygulamalara ait olası performans sorunları tespit ederek iyileştirmeler yapılabilir.

IV. SONUÇLAR

Mobil uygulamaların her geçen gün artan popülaritesi, mobil yazılım testlerini daha fazla önemli hale getirmektedir. Bu çalışma mobil iOS platformunda geliştirilen çok kullanıcılı mobil uygulamaların test süreçlerini ele alarak, literatürde eksik kalan bu alanda katkı sağlamaktadır. Geliştirilen mobil uygulamanın işlevselliği test edilmiş ve literatürde eksik kalan iOS mobil uygulama testlerine yönelik çalışmalar yapılmıştır. Bu çalışma, çok kullanıcılı mobil uygulamanın geliştirilmesini ve testini içermektedir. iOS platformunda iki farklı program başarıyla geliştirilerek testleri yapılmıştır. Geliştirilen mobil yazılım uygulamasının farklı koşullar altında çalışmasını test edebilmek için, test senaryolarının manuel ve otomatik testlerini içermektedir. Yapılmış olan testler neticesinde, 0.992 işlem süresi ölçüldüğü, yapılan çalışma ve elde edilen bulgular, iOS yazılım geliştiricileri için, literatürdeki gizlilik ve güvenlik testleri eksikliği noktasında yol gösterici olacaktır. Ayrıca performans ve veri güvenliği ve gizliliği testleri uygulanmaktadır. Mobil yazılım testlerinin sürekli olarak güncellenmesi hatalarının giderilmesi ve yapılan iyileştirmeler, kullanıcıların beklentilerinin karşılanmasını ve teknolojik gelişmelere uyum sağlayan uygulamaların geliştirilmesi açısından kritik bir öneme sahiptir. Bunların yanı sıra elde edilen sonuçlar değerlendirilerek iOS yazılım geliştirme konusunda geliştiricilere tavsiyeler sunulmuştur. Bu tavsiyelerin mobil uygulamaların kalitesini arttırmaya katkı sağlaması ve geliştiricilere rehberlik etmesi beklenmektedir.

BEYANLAR

Teşekkürler: Düzce Üniversitesi Bilimsel Araştırma Projeleri Koordinatörlüğüne teşekkür ederiz.

Yazarın Katkıları: Kavramsallaştırma, E.K.; metodoloji E.K. ve E. K. T.; doğrulama, M. K. araştırma, Y.T.; kaynaklar, E. K. T.; veri düzenleme, E. K. T.; yazma-orijinal taslak hazırlama, E. K. T.; yazma-inceleme ve düzenleme, E. K. T.; ve E.K. denetim, E.K.

Çıkar Çatışması Açıklaması: Yazar herhangi bir çıkar çatışması beyan etmemektedir.

Destekleyen/Destekleyen Kuruluşlar: Bu araştırma herhangi bir dış fon almamıştır.
Etik Onay ve Katılımcı Onayı: Bu çalışmada herhangi bir etik onay gereksinimi söz konusu değildir.
İntihal Beyanı: Bu makale intihal programıyla taranmıştır. İntihal tespit edilmemiştir.
Veri ve Materyallerin Kullanılabilirliği: Veri paylaşımı geçerli değildir.
YZ Araçlarının Kullanımı: Yazar, bu makalenin oluşturulmasında Yapay Zeka (YZ) araçlarını kullanmadığını beyan etmektedir.

KAYNAKLAR

Amalfitano, D., Fasolino, A. R., & Tramontana, P. (2011). A GUI crawling-based technique for android mobile application testing. Proceedings - 4th IEEE International Conference on Software Testing, Verification, and Validation Workshops, ICSTW 2011, 252–261. https://doi.org/10.1109/ICSTW.2011.77

Afjehei, S., Chen, T., & Tsantalis, N. (2019). Iperfdetector: characterizing and detecting performance anti-patterns in ios applications. Empirical Software Engineering, 24(6), 3484-3513. https://doi.org/10.1007/s10664-019-09703-y

Choudhary, S., Gorla, A., & Orso, A. (2015). Automated test input generation for android: are we there yet? (e).. https://doi.org/10.1109/ase.2015.89

Çiloğlu, T., Üniversitesi, B., Özeren, E., & Üstün, A. B. (2021). MOBİL UYGULAMA GELİŞTİRME, YAYIMLAMA VE EKONOMİK GELİR ETME AŞAMALARININ İNCELENMESİ: İOS VE ANDROİD SİSTEMLERİNİN KARŞILAŞTIRMASI. In e-Journal of New Media (Vol. 5, Issue 1, pp. 60–77). Istanbul Aydin University. https://doi.org/10.17932/IAU.EJNM.25480200.2021/ejnm_v5i1006

Dörr, M., Lesmes, L., Lü, Z., & Bex, P. (2013). Rapid and reliable assessment of the contrast sensitivity function on an ipad. Investigative Opthalmology & Visual Science, 54(12), 7266. https://doi.org/10.1167/iovs.13-11743

Gyorodi, R., Zmaranda, D., Győrödi, C., Győrödi, R., & Georgian Adrian, V. (2017). A Comparative Study between Applications Developed for Android and iOS. Article in International Journal of Advanced Computer Science and Applications, 8(11). https://doi.org/10.14569/IJACSA.2017.081123

Jorgensen, P. C. (2013). Software Testing: A Craftsmans Approach. CRC Press.

Kaner, C., Falk, J., & Nguyen, H. Q. (1999). Testing Computer Software. Wiley.

Knitza, J., Tascilar, K., Messner, E., Meyer, M., Vossen, D., Pulla, A., … & Krusche, M. (2019). German mobile apps in rheumatology: review and analysis using the mobile application rating scale (mars). Jmir Mhealth and Uhealth, 7(8), e14991. https://doi.org/10.2196/14991

Kousar, N., Malik, M., Sarwar, A., Mohy-ud-din, B., & Shahid, A. (2018). Software engineering: challenges and their solution in mobile app development. International Journal of Advanced Computer Science and Applications, 9(1). https://doi.org/10.14569/ijacsa.2018.090127

Kulesovs, I. (2015). iOS Applications Testing. ENVIRONMENT. TECHNOLOGIES. RESOURCES. Proceedings of the International Scientific and Practical Conference, 3(0), 138–150. https://doi.org/10.17770/etr2015vol3.187

Kuļešovs, I., Borzovs, J., Susters, A., Arnicane, V., Arnicans, G., Keiduns, K., & Skutelis, J. (2018). An Approach for iOS Applications' Testing. Baltic J. Modern Computing, 6(1), 56–91. https://doi.org/10.22364/bjmc.2018.6.1.05

Kuroishi, P. H., Maldonado, J. C., & Vincenzi, A. M. R. (2023). Towards the definition of a research agenda on mobile application testing based on a tertiary study. Information and Software Technology, 107363. https://doi.org/10.1016/J.INFSOF.2023.107363

Kuzmik, A., Hannan, J., Boltz, M., Shrestha, P., Husser, E., Fick, D., … & Marcantonio, E. (2023). A pilot study testing the ios ub-cam delirium app. Journal of the American Geriatrics Society, 71(6), 1999-2002. https://doi.org/10.1111/jgs.18252

Li, L., Bissyandé, T. F., Papadakis, M., Rasthofer, S., Bartel, A., Octeau, D., Klein, J., & Traon, L. (2017). Static analysis of android apps: A systematic literature review. Information and Software Technology, 88, 67–95. https://doi.org/10.1016/J.INFSOF.2017.04.001

Martínez, M. and Lecomte, S. (2017). Towards the quality improvement of cross-platform mobile applications.. https://doi.org/10.1109/mobilesoft.2017.30

Muccini, H., Di Francesco, A., & Esposito, P. (2012). Software testing of mobile applications: Challenges and future research directions. 2012 7th International Workshop on Automation of Software Test, AST 2012 - Proceedings, 29–35. https://doi.org/10.1109/IWAST.2012.6228987

Murphy, E. and King, E. (2016). Testing the accuracy of smartphones and sound level meter applications for measuring environmental noise. Applied Acoustics, 106, 16-22. https://doi.org/10.1016/j.apacoust.2015.12.012

Myers, G. J., Sandler, C., & Badgett, T. (2011). The Art of Software Testing. John Wiley & Sons.

Pressman, R. S., & Maxim, B. R. (2014). Software Engineering: A Practitioner's Approach. McGraw-Hill Education.

Qin, X., Zhong, H., & Wang, X. (2019). Testmig: migrating gui test cases from ios to android.. https://doi.org/10.1145/3293882.3330575

Salva, S., Laurencot, P., & Zafimiharisoa, S. (2015). Model inference of mobile applications with dynamic state abstraction., 177-193. https://doi.org/10.1007/978-3-319-23509-7_13

Selvam, R., & Karthikeyani, V. (2011). Mobile Software Testing- Automated Test Case Design Strategies. International Journal on Computer Science and Engineering, 3(4), 1450-1461.

Sunyaev, A., Dehling, T., Taylor, P., & Mandl, K. (2014). Availability and quality of mobile health app privacy policies. Journal of the American Medical Informatics Association, 22(e1), e28-e33. https://doi.org/10.1136/amiajnl-2013-002605

Wang, P., Wu, D., Chen, Z., & Wei, T. (2018). Field experience with obfuscating million-user ios apps in large enterprise mobile development. Software Practice and Experience, 49(2), 252-273. https://doi.org/10.1002/spe.2648

Zein, S., Salleh, N., & Grundy, J. (2023). Systematic reviews in mobile app software engineering: A tertiary study. Information and Software Technology, 164, 107323. https://doi.org/10.1016/J.INFSOF.2023.107323