ARAŞTIRMA

Geliş Tarihi: 17 Aralık 2024
Kabul Tarihi: 19 Ocak 2025

TEST OTOMASYONUNDA YAŞANAN BÜYÜK ZORLUKLAR VE UZMANLARIN TAVSİYELERİ: SEKTÖRDEN UZMAN DEĞERLENDİRMELERİ

Major Challenges In Test Automation And Expert Recommendations: Expert Reviews From The Industry

DOI: 10.58307/kaytek.1596049

Fatih Mehmet Harmancı / Miraç Emektar/ Salim Öncü / Nuri Gürkan Güngör

Özet

Test otomasyonu, yazılım geliştirme süreçlerinin ayrılmaz bir parçası olarak, yazılım hatalarının erken tespiti, maliyetlerin azaltılması ve geliştirme döngülerinin hızlandırılması gibi birçok avantaj sunmaktadır. Bununla birlikte, test otomasyonunun uygulama ve sürdürülebilirlik aşamalarında çeşitli zorluklarla karşılaşılmaktadır. Bu makalede, test otomasyonu alanındaki uzmanların değerlendirmelerine dayanarak, en sık karşılaşılan zorluklar ve bu zorlukların üstesinden gelmek için önerilen stratejiler ele alınacaktır. Test otomasyon araçlarının entegrasyonundan veri yönetimine, performans testlerinden eğitim ve bilinçlendirmeye kadar geniş bir yelpazede ortaya çıkan bu zorluklar, yazılım projelerinin başarısı üzerinde doğrudan etkiye sahiptir. Uzmanların görüşleri doğrultusunda sunulan çözüm önerileri, test otomasyonunun etkinliğini artırmak ve yazılım projelerinin başarısını güvence altına almak için stratejik bir rehber niteliğindedir.

Anahtar Kelimeler: *Test Otomasyon, Yazılım Testi, Yazılım Hataları, Yazılım Projeleri*

Abstract

Test automation is an integral part of software development processes, offering numerous advantages such as early detection of software defects, cost reduction, and acceleration of development cycles. However, various challenges arise in the implementation and sustainability of test automation. This paper explores the most common challenges faced in test automation, based on insights from industry experts, and discusses the strategies recommended to overcome these obstacles. Challenges ranging from tool integration and data management to performance testing and the need for education and awareness have a direct impact on the success of software projects. The expert-driven solutions presented in this study serve as a strategic guide to enhance the effectiveness of test automation and ensure the success of software projects.

Keywords: *Test Automation, Software Testing, Software Defects, Software Projects*

Fatih Mehmet Harmancı / fatihharmanci@hotmail.com / 0009-0008-8691-9574 / Virgosol Yazılım ve Bilişim AŞ.
Miraç Emektar / mirac.emektar@virgosol.com / 0009-0007-7251-6793 / Virgosol Yazılım ve Bilişim AŞ.
Salim Öncü / salim.oncu@virgosol.com / 0009-0002-8508-0240 / Virgosol Yazılım ve Bilişim AŞ.
Nuri Gürkan Güngör / gurkan.gungor@virgosol.com / 0009-0004-4002-0396 / Virgosol Yazılım ve Bilişim AŞ.

GİRİŞ

Yazılım geliştirme süreçlerinde kalite güvencesi ve test otomasyonu, başarılı projelerin temel taşlarından biri olarak kabul edilmektedir.[1] Geleneksel manuel test yöntemlerinin yetersiz kaldığı noktalarda devreye giren test otomasyonu, yazılım hatalarını erken tespit etme, maliyetleri düşürme ve geliştirme döngülerini hızlandırma gibi pek çok avantaj sunar. [2] Ancak, test otomasyonunun sağladığı bu avantajlara rağmen, uygulama ve sürdürülebilirlik açısından birçok zorlukla karşılaşılmaktadır. Bu zorluklar, test otomasyonunun tam potansiyelinin gerçekleştirilmesini engelleyebilir ve yazılım geliştirme ekiplerinin verimliliğini düşürebilir.

Bu makalede, test otomasyonu alanında uzmanlaşmış profesyonellerin görüşlerine dayanarak, en sık karşılaşılan zorluklar ve bu zorlukların üstesinden gelmek için önerilen stratejiler ele alınacaktır. Test otomasyon araçlarının entegrasyonundan veri yönetimine, performans testlerinden eğitim ve bilinçlendirmeye kadar geniş bir yelpazede ortaya çıkan bu zorluklar, yazılım projelerinin başarısı üzerinde doğrudan etkiye sahiptir.

Bu makalede, test otomasyonunda sıklıkla karşılaşılan zorlukları ve bu zorlukların üstesinden gelmek için uzmanların tavsiyelerini inceleyeceğiz. Bu bilgiler doğrultusunda, test otomasyon süreçlerini geliştirmek ve daha etkin hale getirmek için pratik çözümler sunmayı amaçlıyoruz.

I. YAZILIM TEST OTOMASYONU

Test otomasyonu, modern yazılım geliştirme süreçlerinde kalite güvencesinin vazgeçilmez bir bileşeni haline gelmiştir. Manuel testlerin yetersizlikleri, yazılım geliştirme döngülerinin hızlanması ve sürekli entegrasyon/teslimat (CI/CD) süreçlerinin yaygınlaşması, yazılım testlerinin otomatikleştirilmesini zorunlu kılmıştır.[3] Bu bağlamda, test otomasyonu hem zaman hem de maliyet açısından önemli avantajlar sağlamaktadır. Ancak, bu avantajlara rağmen test otomasyonu süreçlerinde çeşitli zorluklar ve engeller bulunmaktadır.

Geçmişte, yazılım testleri genellikle manuel olarak gerçekleştirilirdi. Manuel testler, belirli test senaryolarının insanlar tarafından adım adım uygulanmasını gerektirir. Bu süreç, zaman alıcı ve maliyetli olmanın yanı sıra, insan hatalarına da açık bir yapıya sahiptir.[4] Manuel testlerin tekrarlanabilirliği ve kapsamlı olması zordur. Ayrıca, her testin aynı şekilde uygulanması ve sonuçların tutarlı olması garanti edilemez. Bu sınırlamalar, yazılım projelerinin kalitesini ve güvenilirliğini olumsuz etkileyebilir.

Gartner Akran Topluluğu tarafından yazılım geliştirme yöneticileri üzerinde yapılan bir araştırmada, kuruluşların otomatik yazılım testini benimseme durumu incelenmiştir. Bulgular özetle şöyledir[5] :

- **Yaygın Benimseme**: Katılımcıların %40'ı, geliştirme döngüleri boyunca sürekli olarak otomatikleştirilmiş testleri kullanmaktadır. En yaygın otomatik yazılım testi türleri arasında API testi (%56), entegrasyon testi (%45) ve performans testi (%40) bulunmaktadır.

- **Kalite ve Hız Artışı**: Katılımcıların %60'ı, kuruluşlarının yazılım testini otomatikleştirme nedenleri arasında ürün kalitesinin iyileştirilmesini belirtmiştir. %58'i ise dağıtım hızını artırma isteğini vurgulamıştır.
- **Faydalar**: Katılımcıların %43'ü, testlerin otomatikleştirilmesinden bu yana daha yüksek test doğruluğu elde ettiklerini belirtmiştir. Diğer önemli faydalar arasında artan çeviklik (%42) ve daha geniş test kapsamı (%40) yer almaktadır.
- **Zorluklar**: Otomatik yazılım testi dağıtımında en sık karşılaşılan zorluklar arasında uygulamadaki zorluklar (%36), otomasyon beceri boşlukları (%34) ve yüksek ön maliyetler (%34) bulunmaktadır.

Test otomasyonu, yazılım test süreçlerinin bu zorluklarını aşmak amacıyla geliştirilmiştir. Otomatik testler, yazılım hatalarını erken tespit etme, test süreçlerini hızlandırma, testlerin tekrarlanabilirliğini ve tutarlılığını artırma gibi faydalar sağlar. Otomatik testler, önceden tanımlanmış test senaryolarının yazılım araçları tarafından sürekli olarak çalıştırılmasını içerir. Bu sayede, manuel testlerin zaman alıcı doğası ortadan kalkar ve test süreçleri daha verimli hale gelir.

Test otomasyonu, yazılım test süreçlerinin bu zorluklarını aşmak amacıyla geliştirilmiş olmasına karşın, bu bulgular otomatik testlerin yaygın olarak benimsendiğini, ürün kalitesi ve dağıtım hızını iyileştirdiğini, ancak bazı uygulama zorlukları olduğunu göstermektedir.

II. YÖNTEM

A. Araştırma Tasarımı

Bu çalışmanın amacı, test otomasyonunda karşılaşılan zorlukları ve bu zorluklara yönelik uzmanların önerdiği çözümleri belirlemektir. Bu doğrultuda, nitel araştırma yöntemi benimsenmiştir. Veri toplama aracı olarak görüşme tekniği kullanılmıştır. Alanında uzman kişilerin görüşlerini toplamak ve analiz etmek amacıyla yarı yapılandırılmış görüşmeler gerçekleştirilmiştir. Görüşme soruları, test otomasyonu ile ilgili çeşitli konuları kapsayacak şekilde hazırlanmıştır.

B. Katılımcıların Seçimi ve Özellikleri

Katılımcılar, yazılım geliştirme süreçlerinde deneyime sahip uzmanlar arasından seçilmiştir. Uzmanların pozisyonları ve deneyimleri aşağıdaki kriterlere göre belirlenmiştir. Pozisyon türlerine göre katılımcılar Tablo 1'de belirtilmiştir.

Tablo 1. *Pozisyon Türlerine Göre Katılımcılar*

Pozisyon	Kişi Sayısı
Test Uzmanı / Test Otomasyon Mühendisi	17
Developer	6
Test Lead/Manager	3
DevOps	5
Toplam	**31**

Katılımcıların kıdem sevilerine göre deneyimleri 0-2 yıl, 2-5 yıl, 5 ve üzeri yıl ve diğer olarak kategorize edilmiştir. Kıdem seviyelerine göre katılımcılar Tablo 2'de belirtilmiştir.

Tablo 2. *Kıdem Seviyelerine Göre Katılımcılar*

Kıdem Seviyesi	Kişi Sayısı
Junior (0-2 yıl)	8
Mid-Level (2-5 yıl)	13
Senior (5+ yıl)	10
Toplam	**31**

Farklı sektörlerdeki deneyimlerin araştırmaya dahil edilmesi amacıyla Bankacılık ve Finans, Bilişim ve Teknoloji, Dijital Yayın Platformları (Eğlence), Telekomünikasyon, Sağlık ve Tıp, E-Ticaret gibi çeşitli sektörlerden uzmanlar çalışmaya katılmıştır. Bazı katılımcıların birden fazla sektörde deneyimi bulunmaktadır. Sektörlere göre katılımcı dağılımı Tablo 3'te gösterilmektedir.

Tablo 3. *Sektörlere Göre Katılımcılar*

Sektör	Kişi Sayısı	Sektör	Kişi Sayısı
Bankacılık ve Finans	8	Havayolu	2
Bilişim ve Teknoloji	7	Otel Rezervasyonu, Uçak bileti Rezervasyonu & Filo Yönetimi	2
Dijital Yayın Platformu	3	Eğitim	3
Telekomünikasyon	4	Elektrikli Araç Şarj Etme	2
Sağlık ve Tıp	3	Kozmetik	2
E-Ticaret	4	Savunma Sanayi	2
Otomotiv	3	İthalat-İhracat İşlemleri	2
Toplam			47

C. Veri Toplama ve Analiz

Çalışmanın araştırma soruları aşağıda belirtilmiştir.

1. Test otomasyonunda sıklıkla karşılaştığınız zorluklar ve aldığınız aksiyonlar nelerdir?
2. Test otomasyonuna yeni başlayan şirketler ve kişilere ne tavsiyelerde bulunursunuz? Başarı elde etmek için hangi stratejileri önerirsiniz? Ayrıca, otomasyon projelerinde başarılı olmak için hangi pratikleri uyguluyorsunuz?
3. Test otomasyonuna yapılan yatırımların şirketiniz veya proje performansı üzerindeki etkileri ve getirileri nelerdir? Test otomasyonuna yatırım yapmakta zorlanıyor musunuz ve bu zorlukların ana nedenleri nelerdir? Bu konudaki stratejileriniz nelerdir?
4. Şirketinizde test otomasyonunun verimliliği ve etkinliği konusunda ne gibi sorunlar yaşıyorsunuz? Bu sorunları çözmek için neler yapıyorsunuz ve gelecekte hangi önlemleri almayı planlıyorsunuz? Ayrıca, yeterli test otomasyonu becerilerine sahip kalifiye personel bulmakta zorlanıyor musunuz ve bu durumun etkileri nelerdir? Bu sorunu çözmek için ne tür adımlar atmaktasınız?

Veri toplama görüşmeler ve bu görüşmelerde not alma şeklinde gerçekleştirilmiştir. Bu kapsamda farklı alanlardan 26 uzman ile görüşme gerçekleştirilmiştir. Görüşmeler ve görüşmelerde not alma, nitel araştırmada yaygın olarak kullanılan veri toplama yöntemlerindendir.[6] Görüşmeler, web konferans araçları aracılığıyla yarı yapılandırılmış olarak gerçekleştirilmiş ve her biri 35-65 dakika sürmüştür. Görüşmelerin kaydedilmesinde iki uzmandan destek alınmıştır. Kaydedilen görüşmeler birinci yazar tarafından gözden geçirilmiş ve doğrulanmıştır. Katılımcılara kayıtların nasıl kullanılacağı ve gizliliğin nasıl korunacağı konusunda açık ve şeffaf olunmuştur. Harici kaydediciler görüşmeyi etkilemeyecek şekilde eğitilmişlerdir. Yazar, kayıtları tarafsız ve önyargısız bir şekilde gözden geçirmiştir.

Analiz birinci yazar tarafından gerçekleştirilmiştir. Analiz sırasında, katılımcıların verdikleri yanıtlar tematik olarak gruplandırılmış ve iteratif bir süreçle zorluklar ve çözüm önerileri belirlenmiştir. Elde edilen veriler, test otomasyonunun çeşitli yönlerini ve uzmanların bu konudaki deneyimlerini kapsamlı bir şekilde ortaya koymak amacıyla kullanılmıştır.

D. Geçerlilik ve Güvenilirlik

Bu çalışmanın geçerliliği ve güvenilirliği, çeşitli yöntemlerle sağlanmıştır: Çalışmanın geçerliliğine yönelik tehditler Runeson ve Höst'in sınıflandırma şemasına göre aşağıda tartışılmıştır [7]:

Kavramsal Geçerlilik: Her görüşmenin başında, görüşmeci görüşme konusuna bir giriş yapmış, görüşmenin amacı ve kullanılan terimlerin tanımları açıklanmıştır. Ek olarak,

görüşmeci bir sorunun yanlış anlaşıldığı görüldüğünde veya görüşmeci tarafından açıklandığında, soru hakkında açıklama yapmıştır.

İç Geçerlilik: Görüşmeci, her görüşmenin yapısını benzer tutmaya çalışmak için bir görüşme rehberi kullanmıştır. Ancak, görüşmeler yarı yapılandırılmış bir şekilde gerçekleştirilmiş, bu da soru sırasının değiştirilmesine veya akışı sağlamak için ek sorular sorulmasına olanak tanımıştır. Soru sırasının değişmesi nedeniyle uzmanların yanıtlarında sıralama etkileri göz ardı edilemez. Mümkün olduğunda, görüşmecilere görüşmenin sonunda serbestçe fikirlerini ifade etmeleri teşvik edilmiş zihinlerinden geçen konuları konuşma fırsatı verilmiştir. Bu, görüşme rehberi tasarımı sırasında potansiyel olarak kaçırılan konuların etkisini sınırlamak amacıyla yapılmıştır.

Dış Geçerlilik: Farklı sektörlerden, şirketlerden ve pozisyonlardan gelen katılımcılar seçilerek, test otomasyonunun farklı bağlamlardaki uygulamaları ve zorlukları hakkında geniş bir perspektif elde edilmiştir.

Güvenilirlik: Görüşmelerin yürütülmesi, yazıya dökülmesi ve analizi yalnızca birinci yazar tarafından yapılmıştır. Tüm adımların tek bir kişi tarafından gerçekleştirilmesi, bu kişinin görüşmeleri derinlemesine bilmesini sağlamış ve belirtilen zorlukları tespit etmeyi kolaylaştırmıştır. Bununla birlikte, görüşmelerde belirtilen tüm zorlukların tespit edildiği garanti edilemez. Ayrıca, görüşmelerden alınan ifadelerin etiketlenmesi ve kategorize edilmesi, yazarın öznel yargılarına bağlıdır ve bu durum çalışmanın güvenilirliği için bir tehdit oluşturabilir.

III. BULGULAR

Test otomasyonu sürecinde firmalar ve uzmanlar çeşitli zorluklarla karşılaşmaktadır. Bu zorluklar hem teknik hem de organizasyonel düzeyde kendini göstermektedir. Aşağıda, test otomasyonunda en sık karşılaşılan zorluklar ve bu zorlukların üstesinden gelmek için kullanılan stratejiler özetlenmiştir.

A. Test Otomasyonu ile İlgili Genel Zorluklar ve Hatalar

Test otomasyonu süreçlerinde karşılaşılan başlıca zorluklar ve hatalar, aşağıda detaylı olarak ele alınmıştır. Bu zorluklar, veri yönetiminden element bulma zorluklarına, ortam özelliklerinden otomasyon araçlarına kadar geniş bir yelpazede incelenmiştir.

1.Senaryo Kaynaklı Hatalar

Test otomasyonunda veri kaynaklı hatalar sıkça yaşanmaktadır. Özellikle veri bozulması veya veri tekrar üretme gerekliliği, test senaryolarının stabil çalışmasını engelleyebilir. Test verilerinin güncelliğinin yitirilmesi veya yanlış veri kullanımı test sonuçlarının güvenilirliğini azaltabilir. Bu sorunlarla başa çıkmak için veri yönetimi stratejilerinin iyileştirilmesi ve sürekli veri gözden geçirme süreçlerinin uygulanması önerilmektedir.

Bir uzman, veri sorunlarını şu şekilde dile getirmiştir: *"Veri setleri her zaman güncel ve doğru olmuyor, bu da testlerin doğruluğunu etkiliyor. Test otomasyonu sürekli veri güncellemeleri gerektiriyor, aksi takdirde yanlış sonuçlar elde edilebilir."*

2. Element Bulma Zorlukları

Dinamik ID'ler. Elementlerin ID'lerinin sık sık değişmesi, testlerin kırılmasına neden olmaktadır ve test senaryolarının tutarlılığını tehdit eder. Bir başka görüşe göre: *"Dinamik ID'ler sürekli değişiyor ve bu, testlerin başarısız olmasına yol açıyor. Her değişiklikte testlerin yeniden güncellenmesi gerekiyor, bu da zaman kaybına yol açıyor."*

Shadow DOM. Elementlere erişimi zorlaştıran bir yapı olarak öne çıkar ve test araçlarının iç yapıdaki elementleri bulmasını karmaşıklaştırır. Bir uzman, Shadow DOM ile ilgili şu şekilde yorum yapmıştır: *"Shadow DOM, sayfa yapısının karmaşıklaşmasına neden oluyor ve bu da test senaryolarımızı daha zor hale getiriyor. Test araçları genellikle bu yapıları göz ardı ediyor."*

iFrame. Farklı bir HTML dokümanı içinde yer alan elementlere ulaşmak zordur ve otomasyon testlerinde ekstra zorluklar çıkarabilir. *"iFrame kullanımı testlerde büyük bir engel oluşturuyor. Sayfalar içinde başka sayfalar olduğunda, doğru elemente erişim sağlamak çok daha zor oluyor."*

AngularJS Uygulamaları. Tek sayfalı uygulamalarda (SPA) element bulma zorlukları yaşanmaktadır. Dinamik içeriklerin yönetimi ve elementlerin sürekli değişmesi testlerin etkinliğini azaltabilir. *"AngularJS gibi dinamik uygulamalarla yapılan testler, elementlerin sürekli değişmesi nedeniyle gerçekten karmaşıklaşıyor. Test senaryoları her zaman güncel olmalı, aksi takdirde test geçerli olmuyor."*

3. Ortam Özellikleri

Pop-up'lar ve Bildirimler: Beklenmedik pop-up'lar veya bildirimler test senaryolarını etkileyebilir ve otomasyonun verimliliğini azaltabilir. *"Pop-up'lar sürekli engel teşkil ediyor. Test senaryolarında bu tür beklenmedik durumlarla sık sık karşılaşıyoruz, bu da otomasyonun başarısını engelliyor."*

OTP ve Captcha: Güvenlik amaçlı kullanılan OTP ve Captcha gibi mekanizmalar testlerin otomasyonunu zorlaştırabilir. Bu tür güvenlik katmanları otomasyon testlerinde engeller oluşturabilir. Bir uzman şu şekilde ifade etmiştir: *"Güvenlik katmanları, özellikle OTP ve Captcha, testler için büyük engeller oluşturuyor. Bu tür mekanizmalar otomatik testlerin geçmesini zorlaştırıyor."*

Ödeme Adımları: Ödeme süreçlerinin otomasyonu, güvenlik ve veri gizliliği nedenleriyle hassas bir konudur. Testlerde ödeme adımlarının doğru bir şekilde otomatikleştirilmesi, ek güvenlik önlemleri ve doğru veri yönetimi gerektirir. *"Ödeme adımlarını otomatikleştirmek gerçekten çok zor. Güvenlik ve veri gizliliği her şeyden önemli ve bu konuda dikkatli olmalıyız."*

4. Veri ve Ortam Yönetimi

Veri Kaynaklı Hatalar: Test verilerinin tutarsız veya hatalı olması testlerde başarısızlığa neden olabilir ve testlerin geçerliliğini tehdit edebilir. Bir uzman bu konuda şunları belirtmiştir: *"Veri tutarsızlıkları her zaman karşılaştığımız büyük bir sorun. Testlerin başarısız*

olmasına yol açabiliyor ve bu da projelerin ilerlemesini engelliyor. Verilerin sürekli güncel tutulması çok önemli."

Ortam Stabilitesi: Test ortamının stabil olmaması, test sonuçlarının güvenilirliğini düşürebilir. Test ortamındaki değişiklikler, testlerin tutarlılığını etkileyebilir. Bir başka uzman şu görüşü dile getirmiştir: *"Test ortamları bazen beklenmedik şekilde değişebiliyor ve bu da test sonuçlarımızı doğrudan etkiliyor. Stabil bir test ortamı sağlamak her zaman zor olabiliyor."*

5. Otomasyon Araçları ve Framework'ler (Çerçeveler)

Araçların Sınırlamaları: Bazı otomasyon araçları, belirli teknolojiler veya platformlar için yeterli destek sunmayabilir ve bu durum testlerin yürütülmesini zorlaştırabilir. Bir uzman, araçların sınırlamaları hakkında şu şekilde konuşmuştur: *"Kullandığımız otomasyon aracı bazı platformlarla uyumsuz. Bu nedenle bazı testleri otomatikleştirmek imkansız hale geliyor ve manuel testlere başvurmak zorunda kalıyoruz."*

Framework Seçimi: Senkronizasyon hataları, özellikle farklı uygulamalar arasında veri alışverişinin zamanında gerçekleşmemesi durumunda sıkça karşılaşılan bir sorundur. Servislerde yaşanan yavaşlıklar, senkronizasyon sorunlarına ve zaman aşımı hatalarına neden olabilir. Bir uzman, bu konuda şunları ifade etmiştir: *"Senkronizasyon sorunları çok yaygın. Farklı uygulamalar arasında veri akışında gecikmeler oluyor ve bu da zaman aşımına yol açıyor. Bu tür hatalar testlerin geçerliliğini tehlikeye atabiliyor."*

6. Senkronizasyon ve Zaman Aşımı Hataları

Otomasyon projelerinde senaryoların ve test araçlarının düzenli olarak güncellenmesi ve bakımı, önemli bir zorluk olarak öne çıkmaktadır. Locator'ların (yer imleci) değişmesi, yeni fonksiyonların eklenmesi veya mevcut fonksiyonların değiştirilmesi bakım süreçlerini karmaşık hale getirebilir. Kodun anlaşılır ve bakımı kolay bir şekilde yazılması, düzenli olarak gözden geçirilmesi ve güncellemelerin zamanında yapılması gerekmektedir. Ayrıca, projede kullanılan araçların güncel tutulması bakım zorluklarını azaltabilir. Bu konuda bir uzman şöyle belirtmiştir: *"Otomasyon senaryolarını sürekli güncel tutmak çok zor. Uygulamada yapılan küçük bir değişiklik bile testlerin bozulmasına neden olabiliyor. Özellikle locator'ların değişmesi bakım sürecini çok karmaşık hale getiriyor."*

7. Bakım Zorlukları

Farklı platformlarda test otomasyonu gerçekleştirilirken platformlar arası uyumluluk sorunları yaşanabilir. Aynı senaryonun farklı işletim sistemlerinde veya tarayıcılarda farklı sonuçlar vermesi, testlerin güvenilirliğini sorgulatabilir. Bu tür durumlarla başa çıkmak için cross-platform uyumlu araçlar tercih edilmelidir. Testlerin mümkün olduğunca çeşitli ortamlar üzerinde yürütülmesi sağlanmalıdır. Bir uzman bu durumu şu şekilde açıklamıştır: *"Aynı test senaryosu Chrome'da sorunsuz çalışırken, Safari'de hatalar alıyoruz. Cross-platform testleri gerçekten çok zaman alıyor ve uyum sorunları büyük bir engel."*

8. Cross-Platform Uyum Zorlukları

Olgunlaşmamış, stabil hale gelmemiş yazılım ürünlerinde test otomasyonu uygulamak, bir dizi zorlukla birlikte gelir. Sürekli değişen gereksinimler, kararsız mimari, eksik veya hatalı dokümantasyon gibi faktörler, otomasyon senaryolarının sık sık güncellenmesini ve yeniden yazılmasını gerektirir. Bu durum, otomasyonun getireceği faydaları gölgede bırakabilir ve hatta ek maliyetlere yol açabilir. Ayrıca, test verilerinin hazırlanması ve yönetimi de olgunlaşmamış projelerde önemli bir zorluktur. Bu zorlukla ilgili bir uzman şunları söylemiştir: "Ürün henüz tam oturmadığı için sürekli değişiklikler oluyor. Otomasyon senaryolarını sürekli güncellemek zorundayız ve bu çok fazla zaman ve emek harcamamıza neden oluyor. Henüz tamamlanmamış bir üründe otomasyon yapmak gerçekten çok zor."

9. Olgunlaşmamış, Stabil hale Gelmemiş Ürün

Olgunlaşmamış ve stabil hale gelmemiş yazılım ürünlerinde test otomasyonu uygulamak, önemli zorluklar barındırır. Sürekli değişen gereksinimler, kararsız yazılım mimarisi, eksik veya hatalı dokümantasyon gibi etkenler, otomasyon senaryolarının sık sık güncellenmesini ve yeniden yazılmasını gerektirir. Bu durum, otomasyonun sağladığı faydaların azalmasına ve ek maliyetlerin ortaya çıkmasına neden olabilir. Ayrıca, bu tür projelerde test verilerinin hazırlanması ve yönetilmesi de büyük bir zorluk teşkil eder. Bu konuda bir uzman şunları dile getirmiştir: *"Ürün sürekli değişiyor ve bu değişiklikler otomasyon senaryolarımızı doğrudan etkiliyor. Daha ürün oturmadan otomasyon yapmak, sürekli güncelleme ve revizyon gerektiriyor, bu da hem zaman hem de maliyet açısından büyük bir yük oluşturuyor."*

10. Kültürel ve Yönetimsel Dirençler

Otomasyon projelerinin başarısı sadece teknik yeterlilikle sınırlı değildir; organizasyonel ve kültürel faktörler de büyük önem taşır. Kalite kültürünün ekipler arasında benimsenmesi ve yönetimin kalite odaklı yaklaşımı, projelerin başarıya ulaşmasında kritik rol oynar. Özellikle ekipler arası iletişimin güçlendirilmesi, DevOps dönüşümüne uyum sağlanması ve organizasyon genelinde kalite bilincinin artırılması önemlidir. Bu konuda deneyimli bir uzman şöyle demiştir: *"Yönetim desteği olmadan otomasyon projelerinin ilerlemesi çok zor. Kalite odaklı bir bakış açısı tüm ekiplere yayılmadığında, süreçler yavaş ilerliyor ve hedeflenen verimlilik sağlanamıyor. Organizasyon genelinde kalite kültürünü yerleştirmek şart."*

B. Test Otomasyonu İçin Tavsiyeler ve En İyi Uygulamalar

Katılımcılar test otomasyonunun etkinliğini artırmak ve karşılaşılan zorlukları aşmak için bir dizi tavsiye ve en iyi uygulama örnekleri belirtmişlerdir. Aşağıda, kod kalitesinden gelişmiş tekniklere, iyi bir test mimarisinden ekip iletişimine kadar geniş bir yelpazede ele alınan bu tavsiyeleri ve uygulamaları detaylı olarak inceleyeceğiz. Bu bölüm, test otomasyonu süreçlerini iyileştirmek ve optimize etmek isteyen ekipler için değerli bilgiler sunmaktadır.

1. Page Object Model (POM) ve Clean Code Uygulamaları

Verilen yanıtlar içerisinde kod kalitesi yönüyle Clean Code, Page Object Model (POM), OOP (Object Oriented Programming) ve Best Practices öne çıkmıştır. Page Object Model (POM) ve clean code uygulamaları, test otomasyon projelerinde sürdürülebilirliği ve okunabilirliği artırmak için kritik öneme sahiptir. POM, testlerin daha düzenli ve yeniden kullanılabilir olmasını sağlar. Her sayfanın veya bileşenin bir "Page Object" sınıfı tarafından temsil edilmesi, testlerin bakımını ve güncellenmesini kolaylaştırır. Clean code prensipleri, kodun anlaşılabilir ve bakımı kolay olmasını sağlar, bu da uzun vadede proje verimliliğini artırır.

Katılımcıların bu konuda verdiği bazı görüşler:

- "Page Object Model (POM) kullanmak, testlerin daha düzenli ve sürdürülebilir olmasını sağlıyor. Bu yaklaşım ile test senaryolarını kolayca güncelleyebiliyoruz."
- "Clean Code ilkelerine uyarak kod yazmak, zaman içinde kodun okunabilirliğini artırıyor ve bakımını çok daha kolay hale getiriyor."

Ayrıca, otomasyon araçlarının sunduğu en iyi uygulamaları takip etmek önemlidir. Bir katılımcı, *"Otomasyon araçları sürekli güncelleniyor. Bu güncellemeleri takip etmek, yeni özelliklerden faydalanmak ve mevcut hataları çözmek için çok önemli"* şeklinde bir görüş bildirmiştir.

2. Gelişmiş Test Otomasyon Teknikleri ve Stratejileri

Gelişmiş test otomasyon teknikleri bakımından katılımcılar paralel koşum ve cross-platform testing, servis testlerinin öncelikli otomasyonu, dinamik senaryolar ve Generic locator seçimini vurgulamışlardır.

Paralel Koşum ve Cross-Platform Testing. Testlerin paralel olarak koşulması, test süresini önemli ölçüde azaltabilir. Ayrıca, cross-platform testing ile farklı platformlar üzerindeki uyumluluğu test etmek, çeşitli kullanıcı senaryolarını kapsar. Katılımcılar bu tekniklerin etkinliğini şu şekilde belirtmişlerdir: *"Paralel koşum, özellikle çok sayıda testi aynı anda çalıştırmamız gerektiğinde büyük zaman kazancı sağlıyor. Ayrıca cross-platform testing ile, aynı testleri farklı platformlarda da çalıştırabiliyoruz*."

Servis Testlerinin Öncelikli Otomasyonu. Otomasyon sürecine genellikle servis testleri ile başlanır. Bu, büyük sistemlerin otomasyonuna geçmeden önce, daha küçük ve bağımsız testlerin güvence altına alınmasını sağlar. Katılımcılar, servis testlerini önceliklendirmenin yararlarını şöyle dile getirmiştir: *"Servis testlerini otomatikleştirerek, sistemin temel yapı taşlarını test edebiliriz. Bu, büyük sistemlere geçmeden önce temel sorunları tespit etmemizi sağlıyor*."

Dinamik Senaryolar. Test senaryolarının dinamik olması, statik senaryolardan daha esneklik sağlar. Bu, testlerin farklı veri kümesi ve senaryolarla daha kapsamlı bir şekilde test edilmesini sağlar. Katılımcılar bu konuda şunları paylaşmışlardır: *"Dinamik senaryolar kullanarak testlerin kapsamını genişletebiliyoruz. Sabit test senaryolarından daha esnek ve geniş veri kümeleriyle test yapmamıza olanak tanıyor."*

Jenerik Locator Seçimi ve OOP. Locator stratejilerinde jenerik seçimler ve nesneye dayalı programlama (OOP) kullanmak, testlerde karşılaşılabilecek sorunları minimize eder ve kodun yeniden kullanılabilirliğini artırır. Katılımcıların paylaştığı bir görüş ise şu şekildedir: *"Jenerik locator kullanımı ile kodu çok daha modüler hale getirdik. OOP prensipleri sayesinde kodun tekrar kullanımını artırmak oldukça faydalı oldu."*

3. İyi Bir Test Mimarisi ve Planlama

Araştırmada, başarıya ulaşmak için test mimarisinin iyi bir şekilde yapılandırılmasının çok önemi ifade edilmiştir. *"Proje yapısına hakimiyet ve buna uygun bir planlama yapmak, testlerin daha düzenli ve verimli yürütülmesini sağlar."* Ayrıca, test suitlerinin doğru bir şekilde hazırlanması ve kapsamlı manuel testlerin yapılması da vurgulanmıştır.

4. Bakım ve Yönetişim

Kod Bakımı. Kodun anlaşılır ve bakımının kolay olmasını sağlamak için clean code prensiplerine uyulması gerektiği ifade edilmiştir. Ayrıca, locator bulma yöntemlerinin pratik hale getirilmesi ve test raporlarının okunabilirliğinin artırılması gerektiği belirtilmiştir. Bu yaklaşımlar, bakım sürecinde karşılaşılabilecek sorunları minimize eder.

Kapsayıcı Test Senaryoları. Test senaryolarının bağımsız ve kapsamlı olması gerektiği ifade edilmiştir. Bu yaklaşım, testlerin doğruluğunu artırır ve bakım sürecinde ortaya çıkabilecek problemleri azaltır.

5. Ekip İletişimi ve Eğitim

Ekip iletişimi ve eğitim stratejileri arasında bilgi eksikliklerini araştırma, ekip içi destek alma, kod okunabilirliği, scrum ve agile yöntemleri ve eğitim yöntemleri ön plana çıkmaktadır. *"Bilginin yetersiz olduğu durumlarda, ilgili konular hakkında araştırma yapılıyor ve sorun hala çözülemiyorsa ekip arkadaşlarından yardım isteniyor."* Eğitim ve sorumluluk verme yoluyla personelin yetkinlikleri artırılıyor. Günlük toplantılar ve kısa dönemli planlamalar ile süreçler zaman ve kaynak açısından kontrol ediliyor. Ayrıca, zaman yönetimi için işlerin sprinte (iş dönemi) dahil edilmesi ve kaynak eksikliğini gidermek için eğitim verilmesi gerektiği vurgulanıyor. *"Beceri eksiklikleri, uygulamalı eğitim ve mentorluk ile gideriliyor."* Zaman, kaynak ve beceri eksikliği gibi sorunlar, genellikle ekstra efor, detaylı araştırma ve uzmanlardan yardım alarak çözülüyor. Ayrıca, takım içindeki yetenekleri artırmak için en iyi uygulamalardan yararlanılıyor ve yapay zeka destekli çözümler kullanılıyor.

Kod Okunabilirliği ve Eğitim. Kodun anlaşılır olması, ekip üyelerinin testleri daha iyi anlamasını sağladığı ifade edilmiştir. Ayrıca, ekip içindeki test mühendislerinin otomasyon bilgilerini değerlendirmek ve eksik durumlarda eğitim planları oluşturmak gerektiği belirtilmiştir. Bu yaklaşım, ekip üyelerinin yetkinliklerini artırarak test sürecinin verimliliğini yükseltir.

Scrum ve Agile Yöntemleri. Ekipler arası uyum ve iletişimin artırılması için Scrum ve Agile yöntemlerinin uygulanmasının proje yönetimini ve test sürecini iyileştirebileceği ifade edilmiştir. Bu yöntemler, sürekli geri bildirim ve adaptasyon ile daha etkin bir proje yönetimi sağlar.

6. Teknolojik Altyapı ve Araçlar

Otomasyon Araçları ve Güncellemeler. Test araçlarının güncel tutulması ve gelişmiş test araçlarına geçiş yapılmasının otomasyon sürecinin etkinliğini artırdığı ifade edilmiştir. *"Ayrıca, testlerin code coverage'ını (kod kapsama oranı) ölçmek için ilgili araçların kullanılmasının ve anlaşılır log ile HTML raporlarının üretilmesinin önem arz ettiği"* belirtilmiştir. Bu yaklaşımlar, testlerin kapsamını genişletir ve sonuçların daha anlaşılır olmasını sağlar.

Paralel ve Bulut Koşumları. Testlerin bulut platformlarında (örneğin, BrowserStack gibi) paralel olarak koşulmasının test süresini kısaltabileceği ve daha geniş bir cihaz yelpazesi üzerinde test yapılmasını sağlayabileceği ifade edilmiştir. *"Bu yöntemler, testlerin etkinliğini ve kapsamını artırarak, daha hızlı ve kapsamlı sonuçlar elde edilmesine olanak tanır*."

7.Olgunlaşmamış Yazılımlarda Test Otomasyonuna Başlarken Alınacak Önlemler

Olgunlaşmamış yazılım ürünlerinde test otomasyonuna başlarken dikkat edilmesi gereken temel unsurlar, sürecin karmaşıklığını yönetmek ve değişen gereksinimlere uyum sağlamak için önemlidir. *"İlk olarak, test otomasyonuna aşamalı bir yaklaşım benimsemek gerekir*." Bu süreçte, başlangıçta daha istikrarlı ve temel işlevlere odaklanarak, basit ve sağlam bir otomasyon çerçevesi oluşturulmalıdır. Bu kapsamda daha net ve stabil hale gelen gereksinimler için hazırlanan senaryoların otomasyona alınmada önceliklendirilmesi önemlidir. Otomasyon altyapısı zamanla genişletilirken, gereksinimlerin daha net ve stabil hale gelmesi beklenmeli ve manuel testler ile ürünün temel işlevlerinin doğru çalıştığından emin olunmalıdır. *"Bu strateji, testlerin verimliliğini artırırken, sık sık yapılan kod değişikliklerinden kaynaklanan uyumsuzlukların önüne geçer*."

Ayrıca, esnek ve modüler bir otomasyon mimarisi tasarlamak, değişikliklere hızlı adapte olmayı sağlar ve uzun vadede otomasyonun sürdürülebilirliğini garantiler. *"Sürekli entegrasyon ve sürekli dağıtım (CI/CD) pratiklerinin erken aşamada uygulanması, her yazılım değişikliğinin otomatik olarak test edilmesini ve hataların hızlıca tespit edilmesini sağlar*." Geliştirme ekibiyle yakın işbirliği içinde çalışarak, test otomasyonunu geliştirme sürecinin ayrılmaz bir parçası haline getirmek ve değişiklikleri önceden öngörmek, proaktif bir yaklaşımı destekler. *"Otomasyon stratejisinin düzenli olarak gözden geçirilmesi ve ürünün olgunlaşma sürecine paralel olarak ayarlanması, test otomasyonunun etkinliğini artırarak yazılım kalitesinin iyileştirilmesine katkıda bulunur*."

C. Test Otomasyonuna Yapılan Yatırımlar, Proje Performansı Üzerindeki Etkileri

Test otomasyonuna yapılan yatırımlar, genellikle yüksek verimlilik, kalite artışı ve zaman tasarrufu sağlıyor. *"Otomasyon, acil durumlarda güven verici ve önemli sistem bölümlerinin test edilmesini mümkün kılıyor*." Kaliteyi artırarak hataları hızla tespit etme ve düzelme imkânı sunuyor, bu da uzun vadede maliyetleri düşürüyor ve piyasaya sürme süresini kısaltıyor. Ayrıca, otomasyonun etkileri arasında ürün kalitesinin artması, kullanıcı memnuniyetinin artması ve sistem güvenilirliğinin sağlanması bulunuyor. *"Bu yatırımlar, yazılımın her sürüm öncesi kontrol edilmesini ve hatasız bir uygulama sunulmasını*

sağlayarak maddi kayıpların önüne geçiyor." Bu süreç, yeni geliştirmelerin eski özellikleri etkileyip etkilemediğini kısa sürede ortaya koyarak zaman ve efor tasarrufu sağlamaktadır. Ayrıca, test otomasyonu sayesinde proje performansı artırılmakta ve canlı çıkan hata sayısı azaltılmaktadır.

Test otomasyonuna yatırım yapmakta yaşanan zorluklar arasında eğitimli personel ihtiyacı, yüksek maliyetler ve yeterli test uzmanı bulamama gibi faktörler öne çıkıyor. *"Eğitimli personel eksikliği, device farm ve sunucu maliyetleri gibi faktörler bütçe sınırlamaları ile birleşiyor*." Ayrıca, deneyimli eleman eksikliği ve otomasyon kültürü eksiklikleri gibi zorluklar da bulunuyor. Bu zorlukları aşmak için şirket içindeki daha küçük projelerde otomasyonun faydaları gösterilerek uzun vadeli yararlar vurgulanıyor. *"Bütçe kısıtlamaları ve kısa vadeli sonuç beklentileri de yatırım yapmayı zorlaştıran unsurlar arasında yer alıyor*."

D. Verimlilik, Etkinlik ve Kalifiye Personel Sorunları

Otomasyon projelerinde zaman, kaynak veya beceri eksikliği gibi zorluklarla başa çıkmanın çeşitli yöntemleri öne çıkmaktadır. İlk olarak, bilgi eksikliklerinde araştırma yapmak ve gerektiğinde ekip arkadaşlarından destek almak önemli bir strateji olarak vurgulanmaktadır. *"Eğitim ve sorumluluk vererek personelin yetiştirilmesi, günlük toplantılar ve kısa süreli planlamalar ile süreçlerin kontrol edilmesi sağlanmaktadır*." Zaman eksikliğinde, proje başlangıcında iyi bir planlama yapılarak ilerleyen süreçlerde olası hataların önüne geçilmesi hedeflenmektedir. Ayrıca, beceri eksikliği durumunda uygulamalı eğitimler ve *"buddy"* sistemleri ile personelin gelişimi sağlanmaktadır.

Kaynak ve beceri eksikliklerıyle başa çıkmada, ekibin mevcut yetkinliklerinin artırılması için ekstra efor harcanmakta ve alanında uzman kişilerden yardım alınmaktadır. *"Zaman ve kaynak eksikliklerinde, önceliklendirme yaparak kritik alanlara odaklanmak ve güncel dokümanlar ile online kurslardan faydalanmak etkili yöntemler arasında yer almaktadır*." Ekstra mesai ve pratik yaparak sorunların üstesinden gelinmekte, ekip içindeki yetkin kişilerin bilgi transferi sağlanmaktadır. AI uygulamaları ve Stack Overflow gibi kaynaklardan faydalanarak zaman ve beceri eksikliklerinin üstesinden gelinmektedir. Ekip çalışması ve yöneticilerle iletişim de bu süreçlerin başarıyla yönetilmesinde önemli rol oynamaktadır.

Test otomasyon uzmanı eksikliklerini gidermek ve şirket içine bilgi ve deneyim transferini sağlamak amacıyla dış kaynak alımına gitmek, birçok fayda sağlar. *"Bu yaklaşım, uzmanlık gerektiren alanlarda deneyimli profesyonellerin hızla devreye girmesini sağlayarak, projelerin kesintisiz ilerlemesine olanak tanır."* Aynı zamanda, dış kaynaklardan gelen uzmanlar, en iyi uygulamaları ve güncel trendleri şirkete taşıyarak, iç ekiplerin bilgi seviyesini artırır. Böylece, şirket içi kaynakların eğitilmesi ve yetkinliklerinin yükseltilmesi sağlanırken, uzun vadede daha güçlü bir test otomasyon ekibi oluşturulur. Dış kaynak kullanımı, projelerin zamanında ve kaliteli bir şekilde tamamlanmasını desteklerken, iç kaynakların stratejik görevlerine odaklanmasına da imkân tanır.

Kalifiye test otomasyonu personeli bulmakta yaşanan zorluklar arasında, test otomas-

yonunu ve manuel testleri ayrı ayrı görebilen adayların bulunamaması öne çıkıyor. *"Bu sorun, iç eğitimler ve işbirliği yöntemleri ile çözümlenmeye çalışılıyor."* Ayrıca, uzman firmalardan danışmanlık alınarak bu alandaki eksiklikler giderilmeye çalışılıyor. Kalifiye adayların bulunamaması, projelerde eksiklikler yaratabiliyor ve bu sorunun çözülmesi için iş ilanları ve pozisyon bilgilendirmeleri daha açık ve detaylı hale getiriliyor.

Test otomasyonunun verimliliği ve etkinliği konusunda yaşanan sorunlar arasında kalite kültürünün yaygınlaştırılması, maliyet artışları ve teknolojik altyapının desteklenmesi yer almaktadır. *"Test senaryolarının düzenli olarak gözden geçirilmesi, otomasyon araçlarının güncel tutulması ve gelişmiş test araçlarına geçiş yapılması gibi adımlar sorunları çözmede kullanılmaktadır."* Ayrıca, test mimarisinin iyi yapılması, ekip içi uyum ve iletişim, anlaşılır test yazımları ve kod bakımının iyileştirilmesi gibi önlemler de planlanmaktadır.

VII. SONUÇLAR

Bu çalışmada, test otomasyonunda karşılaşılan ana zorluklar ve bu zorluklara yönelik önerilen çözümler kapsamlı bir şekilde ele alınmıştır. Araştırma bulguları, test otomasyonunun birçok avantaj sunmasına rağmen, veri yönetimi, test edilecek uygulamanın otomasyona uygunluğu (güvenlik, OTP, Captcha, istikrarsız test ortamı, elementlerin uygun tanımlanmaması) ve senkronizasyon problemleri gibi çeşitli engeller içerdiğini göstermiştir. Bu zorluklar, yazılım geliştirme süreçlerinde test otomasyonunun etkinliğini sınırlayabilir ve ek maliyetler doğurabilir.

Çalışmanın bulguları, ayrıca test otomasyon araçlarının seçiminin önemini vurgulamaktadır. Uygun araç seçimi, otomasyon sürecinin başarısını doğrudan etkiler. Uzmanlar, test otomasyonunun sürdürülebilirliği için verilerin ve ortamların dikkatli bir şekilde yönetilmesi gerektiğini belirtmektedir.

Ayrıca, test otomasyonu sürecinde karşılaşılan bakım zorlukları da önemli bir konudur. Otomasyon sistemlerinin düzenli olarak güncellenmesi ve bakımlarının yapılması, uzun vadede maliyetleri azaltabilir ve otomasyon süreçlerinin etkinliğini artırabilir. Senkronizasyon ve zaman aşımı problemleri, performans iyileştirmeleri ile çözülebilir. Bu konuda yapılan iyileştirmeler, otomasyon süreçlerinin başarısını destekleyebilir.

Ayrıca, yaşanan zorlukların üstesinden gelmek ve test otomasyonunun tüm potansiyelini ortaya çıkarmak için, şirketlerin test otomasyon uzmanlığı konusunda dış kaynak kullanımı gibi stratejileri değerlendirmesi faydalı olacaktır.

Sonuç olarak, test otomasyonunun başarısı, karşılaşılan zorlukların etkili bir şekilde yönetilmesine bağlıdır. Bu çalışma, test otomasyonunda karşılaşılan zorlukların anlaşılması ve bu zorluklara yönelik stratejilerin geliştirilmesinin önemini ortaya koymaktadır. Gelecek çalışmalar, belirli çerçevedeki uygulamalara özgü (Web, IOS, Android, Desktop, vs) zorlukları daha derinlemesine inceleyerek, test otomasyonunun çeşitli alanlardaki etkinliğini artırabilir.

Etik Beyanı: Yazarlar bu çalışmanın tüm hazırlanma süreçlerinde etik kurallara uyulduğunu beyan eder. Aksi bir durumun tespiti halinde Kamu Yönetimi ve Teknoloji Dergisinin hiçbir sorumluluğu olmayıp, tüm sorumluluk çalışmanın yazarına aittir.

Ethics Statement: The authors declare that the ethical rules are followed in all preparation processes of this study. In the event of a contrary situation, the Journal of Public Administration and Technology has no responsibility and all responsibility belongs to the authors of the study.

Yazar Katkıları: Fatih Mehmet HARMANCI, Miraç EMEKTAR, Salim ÖNCÜ ve Nuri Gürkan GÜNGÖR çalışmanın tamamında birlikte katkı sunmuştur.

Author Contributions: Fatih Mehmet HARMANCI, Miraç EMEKTAR, Salim ÖNCÜ and Nuri Gürkan GÜNGÖR have contributed to all parts and stages of the study.

Çıkar Beyanı: Yazarlar ve herhangi bir kurum/ kuruluş arasında çıkar çatışması yoktur.

Conflict of Interest: There is no conflict of interest among the authors and any institution.

Teşekkür: Yayın sürecinde katkısı olan hakemlere teşekkür ederiz.

Acknowledgement: We would like to thank the referees who contributed to the publication process.

KAYNAKÇA

Black, R., van Veenendaal, E., & Graham, D. (2020). *Foundations of software testing: ISTQB certification* (3rd ed.). ISTQB.

Crispin, L., & Gregory, J. (2014). *Agile testing: A practical guide for testers and agile teams* (2nd ed.). Addison-Wesley.

Dustin, E., Rashka, J., & Paul, J. (2008). *Automated software testing: Introduction, management, and performance* (13th ed.). Addison-Wesley.

Gartner. (2023). Automated software testing adoption trends. Retrieved March 25, 2024, from https://www.gartner.com/peer-community/oneminuteinsights/automated-software-testing-adoption-trends-7d6

Humble, J., & Farley, D. (2010). *Continuous delivery: Reliable software releases through build, test, and deployment automation* (1st ed.). O'Reilly Media.

Morse, J. M. (2016). *Essentials of qualitatively-driven mixed-method designs*. Routledge.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering, 14*(2), 131. https://doi.org/10.1007/s10664-009-9123-8