BAŞKENT ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
YÖNETİM BİLİŞİM SİSTEMLERİ ANABİLİM DALI
YÖNETİM BİLİŞİM SİSTEMLERİ TEZLİ YÜKSEK LİSANS PROGRAMI

DEVOPS PROJE YÖNETİMİNE YÖNELİK UYARLAMA MODELLERİ: SAVUNMA SANAYİİNDE UYGULAMA

HAZIRLAYAN

Pınar KICIR

YÜKSEK LİSANS TEZİ

TEZ DANIŞMANI

Prof. Dr. Murat Paşa UYSAL

ANKARA – 2025

BAŞKENT ÜNİVERSİTESİ

SOSYAL BİLİMLER ENSTİTÜSÜ

YÜKSEK LİSANS TEZ ÇALIŞMASI ORİJİNALLİK RAPORU

Tarih: 03 / 02 /2025

Öğrencinin Adı, Soyadı : Pınar KICIR

Öğrencinin Numarası : 22220340

Anabilim Dalı : Yönetim Bilişim Sistemleri

Programı : Yönetim Bilişim Sistemleri Tezli Yüksek Lisans

Danışmanın Unvanı/Adı, Soyadı: Prof. Dr. Murat Paşa UYSAL

Tez Başlığı : Devops Proje Yönetimine Yönelik Uyarlama Modelleri: Savunma Sanayiinde Uygulama

Yukarıda başlığı belirtilen Yüksek Lisans tez çalışmamın; Giriş, Ana Bölümler ve Sonuç Bölümünden oluşan, toplam 105 sayfalık kısmına ilişkin, 31 / 01 / 2025 tarihinde tez danışmanım tarafından Turnitin adlı intihal tespit programından aşağıda belirtilen filtrelemeler uygulanarak alınmış olan orijinallik raporuna göre, tezimin benzerlik oranı %2'dir. Uygulanan filtrelemeler:

1. Kaynakça hariç

2. Alıntılar hariç

3. Beş (5) kelimeden daha az örtüşme içeren metin kısımları hariç

"Başkent Üniversitesi Enstitüleri Tez Çalışması Orijinallik Raporu Alınması ve Kullanılması Usul ve Esaslarını" inceledim ve bu uygulama esaslarında belirtilen azami benzerlik oranlarına tez çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Öğrenci İmzası:

ONAY

Tarih: 03 / 02 / 2025

Öğrenci Danışmanı Unvan, Ad, Soyad, İmza:

Prof. Dr. Murat Paşa UYSAL

TEŞEKKÜR

Bu tezin tamamlanmasında destek olan herkese teşekkür ederim.

Özellikle, bilgi ve rehberliğiyle her aşamada bana yol gösteren, değerli önerileri ve sabrıyla çalışmama büyük katkı sağlayan danışmanım Prof. Dr. Murat Paşa Uysal'a teşekkürlerimi sunuyorum.

Tezimin uygulama sürecinde kaynak sağlayan, fikir alışverişine olanak tanıyan ve profesyonel bir çalışma ortamı sunan Türk Havacılık ve Uzay Sanayii A.Ş. (TUSAŞ) ailesine teşekkür ederim. Çalışma arkadaşlarımın desteği ve iş birliği, bu süreci daha verimli hale getirmiştir.

Son olarak, sabır ve sevgileriyle her zaman yanımda olan aileme, eşime ve çocuklarıma teşekkür ederim. Bu çalışma, onların desteği olmadan tamamlanamazdı.

Pınar KICIR

Ankara, 2025

ÖZET

KICIR, Pınar, Devops Proje Yönetimine Yönelik Uyarlama Modelleri: Savunma Sanayiinde Uygulama, Başkent Üniversitesi, Sosyal Bilimler Enstitüsü, Yönetim Bilişim Sistemleri Tezli Yüksek Lisans Programı, 2025

Son yıllarda yazılım, donanım ve teknolojide gözlenen hızlı ve baş döndürücü gelişmeler yazılım geliştirme süreçleri, yöntem ve tekniklerinde de önemli değişiklikleri gündeme getirmiştir. Bunlar arasında kurumsal ve işletme ihtiyaçlarının karşılanması ile yazılım geliştirme, operasyon ve sürekli iyileştirme süreçlerinin hızlandırılması olduğunu söylemek mümkündür. Bu kapsamda, yazılım geliştirme ve operasyon süreçlerinin birleşimine dayanan DevOps (Development and Operations) Proje Yönetim Modelinin ön plana çıktığı gözlenmektedir. DevOps, yazılım geliştirme süreçleri ve bilişim teknolojileri (BT) operasyonları arasındaki iş birliğini artırmayı, süreçlerin otomasyonu ile yazılım teslimat sürelerinin kısaltılmasını hedefleyen bir yazılım geliştirme yaklaşımıdır. Sürekli entegrasyon (CI), sürekli teslim (CD) ve otomasyon temelli çalışma kültürüne dayanan DevOps'un, modern yazılım projelerinin hız, verimlilik, kalite vb. ihtiyaçlarının karşılamada önemli roller üstleneceği değerlendirilmektedir. Ancak, DevOps'un standart yazılım süreçleri ile kurumsal BT operasyonlarına farklı yaklaşımı beraberinde çeşitli güçlükleri de getirmektedir. Söz konusu güçlükleri yazılım geliştirme süreçleri, ihtiyaç yönetimi, takım yönetimi, risk yönetimi, kalite yönetimi, otomasyon vb. alanlarda gruplamak mümkündür. Bu araştırmanın temel amacı, savunma sanayiinde yer alan ve yüksek teknoloji ürünlerini geliştiren bir kurum için DevOps Proje Yönetim Modelinin uyarlanmasına yönelik modeller geliştirmek ve uygulama önerilerini ortaya koymaktır. Çalışma nitel araştırma yaklaşımı doğrultusunda Eylem Araştırması ve doküman incelemesi yöntemleri kullanılarak iki aşamada yürütülmüştür. Birinci aşamada yazılım geliştirme uzmanlarıyla yarı yapılandırılmış mülakatlar gerçekleştirilerek veri toplanmış ve analiz edilmiştir. Mülakatlarda, DevOps kapsamında ihtiyaç yönetimi, takım yönetimi, risk yönetimi, kalite yönetimi ile yapay zekâ kullanımı konuları, karşılaşılan güçlükler, kullanılan yöntem, teknik ve araçlara ilişkin görüşler alınmıştır. Araştırma verisi tümden gelim (dedüktif) ve tüme varım (endüktif) tematik analiz yöntemleri kullanılarak incelenmiştir. İlk aşamada, araştırma soruları ve kuramsal çerçeve temel alınarak dedüktif kodlama yapılmış ve kod katalogları oluşturulmuştur. Daha sonra endüktif tematik analiz yöntemiyle yeni temalar ve desenler

araştırılmıştır. Veri analizi sürecinde dedüktif yaklaşım araştırmanın kuramsal çerçevesini oluştururken, endüktif yaklaşım ise toplanan veriden yeni bulguların ortaya çıkarılmasını sağlamıştır. Araştırmanın ikinci aşamasında, birinci aşamada elde edilen bulgular doğrultusunda UML diyagramları kullanılarak DevOps uyarlama modelleri geliştirilmiştir. Söz konusu uyarlama modelleri, ana temalara (kod kataloglarına), kullanıcıların işlevsel ve işlevsel olmayan ihtiyaçlarına ve UML kurallarına uygunluk bakımından senaryo analizi yöntemi kullanılarak değerlendirilmiştir. Sonuç olarak bu çalışmanın çıktılarının, araştırmanın yapıldığı ilgili kuruma olduğu kadar yazılım mühendisliği ile proje yönetimi uygulama ve araştırma alanlarına da önemli katkılarda bulunabileceği değerlendirilmektedir.

Anahtar Kelimeler: DevOps, Proje Yönetimi, Uyarlama, Uyarlama Modelleri.

ABSTRACT

KICIR, Pınar, Adaptation Models for DevOps Project Management: Application in Defense Industry, Başkent University, Institute of Social Sciences, Master of Management Information Systems With Thesis, 2025

In recent years, rapid and groundbreaking advancements in software, hardware, and technology have brought significant changes to software development processes, methods, and techniques. Among these changes, meeting organizational and business requirements, as well as accelerating software development, operations, and continuous improvement processes, have gained prominence. In this context, the DevOps (Development and Operations) Project Management Model, which integrates software development and operational processes, has emerged as a key approach. DevOps is a software development methodology aimed at enhancing collaboration between software development processes and IT operations, reducing software delivery times through process automation. Based on a culture of continuous integration (CI), continuous delivery (CD), and automation, DevOps plays a critical role in addressing the needs of modern software projects, such as speed, efficiency, and quality. However, DevOps introduces unique challenges due to its different approach to standard software processes and corporate IT operations. These challenges can be categorized into areas such as software development processes, requirements management, team management, risk management, quality management, and automation. The primary goal of this research is to develop adaptation models and propose implementation strategies for the DevOps Project Management Model tailored to a high-technology organization within the defense industry. The study follows a qualitative research approach, employing Action Research and document analysis methods across two phases. In the first phase, data were collected and analyzed through semi-structured interviews with software development experts. The interviews addressed topics such as requirements management, team management, risk management, quality management, and the use of artificial intelligence within the scope of DevOps. Opinions on encountered challenges, methods, techniques, and tools were also gathered. The research data were examined using deductive and inductive thematic analysis methods. Initially, deductive coding was applied based on the research questions and theoretical framework, and codebooks were created. Subsequently, new themes and patterns were explored using inductive thematic analysis.

During the data analysis process, the deductive approach established the theoretical foundation of the research, while the inductive approach facilitated the discovery of new findings from the collected data. In the second phase of the research, DevOps adaptation models were developed using UML diagrams based on the findings from the first phase. These adaptation models were evaluated using scenario analysis to ensure alignment with the main themes (codebooks), users' functional and non-functional requirements, and UML standards. Ultimately, it is expected that the outcomes of this study would provide significant contributions not only to the organization where the research was conducted but also to the fields of software engineering and project management, both in practice and research.

Keywords: DevOps, Project Management, Adaptation, Adaptation Models.

İÇİNDEKİLER

TEŞEKKÜR i
ÖZET ii
ABSTRACT iv
TABLOLAR LİSTESİ ix
ŞEKİLLER LİSTESİ x
SİMGELER VE KISALTMALAR LİSTESİ xii
1. GİRİŞ 1
1.1. Problem 2
1.2. Araştırmanın Amacı 4
2. KURAMSAL ÇERÇEVE 5
2.1. Gereksinim Yönetimi 5
2.2. Yazılım Kalite Mühendisliği 7
2.3. Risk Yönetimi 9
2.3.1. Risk yönetiminde analiz teknikleri 11
2.3.2. Teorik çerçeve ve yaklaşımlar 11
2.4. Takım Yönetimi 12
3. İLGİLİ ÇALIŞMALAR 15
4. YÖNTEM 24
4.1. Araştırma Modeli 24
4.2. Araştırmanın Çıktıları 25
4.3. Evren ve Örneklem 26
4.4. Mülakat 26
4.5. Veri Toplama 26
4.6. Veri Analizi 27
4.7. Geçerlilik ve Güvenirlik 29
4.8. Sınırlılıklar 29
5. BULGULAR 30
5.1. Tümdengelim Tematik Analiz Bulguları 31
5.1.1. DevOps süreç yönetimi kod kataloğu 32
5.1.2. Gereksinim yönetimi kod kataloğu 32
5.1.3. Kalite yönetimi kod kataloğu 33

5.1.4. Risk yönetimi kod kataloğu 34
5.1.5. Takım yönetimi kod kataloğu 35
5.1.6. Yapay zeka yönetimi kod kataloğu 35
5.2. Tümevarım Tematik Analiz Bulguları 36
5.2.1. Problem sahaları tematik analizi 36
5.2.2. Uyarlama tematik analizi 42
5.2.3. Takım yönetimi tematik analizi 47
5.2.4. Gereksinim yönetimi tematik analizi 53
5.2.5. Risk yönetimi tematik analizi 58
5.2.6. Kalite yönetimi tematik analizi 63
5.2.7. Güçlükler tematik analizi 67
5.2.8. Yapay zekâ tematik analizi 71
5.2.9. Yapay zeka entegrasyonu tematik analizi 74
5.3. Mülakat Sonuçları ve Bulguların Modellerle Gösterimi 79
5.3.1. DevOps süreç yönetim modeli 79
5.3.2. DevOps kalite yönetimi modeli 80
5.3.3. DevOps risk yönetimi modeli 80
5.3.4. DevOps gereksinim yönetimi modeli 81
5.3.5. DevOps takım yönetimi modeli 82
6. UYARLAMA MODELLERİ 83
6.1. Model Geliştirme 83
6.1.1. DevOps süreç yönetimi uyarlama modeli 83
6.1.2. Kalite yönetimi uyarlama modeli 84
6.1.3. Risk yönetimi uyarlama modeli 85
6.1.4. Gereksinim yönetimi uyarlama modeli 86
6.1.5. Takım yönetimi uyarlama modeli 87
6.1.6. Yapay zekâ yönetimi uyarlama modeli 88
6.2. Uyarlama Modellerini Geçerleme ve Doğrulama 89
6.2.1. Senaryo analizi 89
6.2.1.1. Senaryonun tanımlanması 89
6.2.1.2. Model geçerleme ölçütlerinin tanımlanması 90
6.2.1.3. Senaryonun çalıştırılması 91
6.2.1.4. Bulgular ve değerlendirme 92
6.2.1.4.1. Senaryo çalıştırma ve değerlendirme bulguları 92

6.2.1.4.2. Matris değerlendirme sonuçları 94
6.2.1.4.3. Genel değerlendirme 94
6.3. Uyarlama Modellerinin Güncellenmesi 95
7. TARTIŞMA 98
7.1. İletişim ve İşbirliği Eksiklikleri 99
7.2. Teknolojik Entegrasyon Zorlukları 99
7.3. Süreç Yönetimi ve Otomasyon 99
7.4. Kültürel Değişim 100
8. SONUÇ VE ÖNERİLER 101
8.1. Teknik Düzeyde İyileştirme ve Otomasyon Stratejileri 101
8.2. Organizasyonel ve Kültürel Dönüşüm Stratejileri 102
8.3. Rol ve Sorumlulukların Belirlenmesi 102
8.4. Güçlükler ve Çözüm Önerileri 103
8.4.1. İhtiyaç yönetimi 103
8.4.2. Takım yönetimi 103
8.4.3. Risk yönetimi 104
8.4.4. Kalite yönetimi 104
8.4.5. Yapay zekâ kullanımı 104
KAYNAKLAR 106
EKLER
EK 1: Yarı Yapılandırılmış Mulakat Soruları

TABLOLAR LİSTESİ

Sayfa

Tablo 5.1. Görüşmecilerin Cinsiyetlerine Göre Dağılımı 30
Tablo 5.2. Görüşmecilerin Yaş Aralığına Göre Dağılımı 30
Tablo 5.3. Görüşmecilerin İş Tecrübesi Aralığına Göre Dağılımı 30
Tablo 5.4. Görüşmecilerin Yazılım İş Tecrübesi Aralığına Göre Dağılımı 31
Tablo 5.5. Görüşmecilerin DevOps Tecrübesi Aralığına Göre Dağılımı 31
Tablo 5.6. Soru 2'e Ait Kodlama 32
Tablo 5.7. Soru 4'e Ait Kodlama 33
Tablo 5.8. Soru 6'ya Ait Kodlama 34
Tablo 5.9. Soru 5'e Ait Kodlama 34
Tablo 5.10. Soru 3'e Ait Kodlama 35
Tablo 5.11. Soru 8'e Ait Kodlama 36
Tablo 5.12. Soru 1'e Ait Kodlama 37
Tablo 5.13. Soru 1'e Ait Tematik Analiz 41
Tablo 5.14. Soru 2'ye Ait Kod Frekansı 42
Tablo 5.15. Soru 2'ye Ait Tematik Analiz 47
Tablo 5.16. Soru 3'e Ait Kod Frekansı 48
Tablo 5.17. Soru 3'e Ait Tematik Analiz 52
Tablo 5.18. Soru 4'e Ait Kod Frekansı 53
Tablo 5.19. Soru 4'e Ait Tematik Analiz 57
Tablo 5.20. Soru 5'e Ait Kod Frekansı 58
Tablo 5.21. Soru 5'e Ait Tematik Analiz 62
Tablo 5.22. Soru 6'ya Ait Kod Frekansı 63
Tablo 5.23. Soru 6'ya Ait Tematik Analiz 67
Tablo 5.24. Soru 7'ye Ait Kod Frekansı 68
Tablo 5.25. Soru 7'ye Ait Tematik Analiz 71
Tablo 5.26. Soru 8'e Ait Tematik Analiz 72
Tablo 5.27. Soru 8'e Ait Kod Frekansı 73
Tablo 5.28. Soru 9'a Ait Tematik Analiz 75
Tablo 5.29. Soru 9'a Ait Kod Frekansı 76
Tablo 6.1. Model Geçerleme Ölçütleri 90
Tablo 6.2. Senaryo Değerlendirme Matrisi 92

ŞEKİLLER LİSTESİ

Sayfa

Şekil 4.1. Araştırma Modeli 25
Şekil 5.1. QDA Miner Lite Soru 1'e Ait Kod Bulutu Görseli 39
Şekil 5.2. QDA Miner Lite Soru 1'e Ait Kod Dağılımı Grafiği 40
Şekil 5.3. QDA Miner Lite Soru 2'ye Ait Kod Dağılımı Grafiği 45
Şekil 5.4. QDA Miner Lite Soru 2'ye Ait Kod Bulutu Görseli 46
Şekil 5.5. QDA Miner Lite Soru 3'e Ait Kod Dağılımı Grafiği 50
Şekil 5.6. QDA Miner Lite Soru 3'e Ait Kod Bulutu Görseli 51
Şekil 5.7. QDA Miner Lite Soru 4'e Ait Kod Dağılımı Grafiği 56
Şekil 5.8. QDA Miner Lite Soru 4'e Ait Kod Bulutu Görseli 56
Şekil 5.9. QDA Miner Lite Soru 5'e Ait Kod Dağılımı Grafiği 61
Şekil 5.10. QDA Miner Lite Soru 5'e Ait Kod Bulutu Görseli 61
Şekil 5.11. QDA Miner Lite Soru 6'ya Ait Kod Dağılımı Grafiği 65
Şekil 5.12. QDA Miner Lite Soru 6'ya Ait Kod Bulutu Görseli 66
Şekil 5.13. QDA Miner Lite Soru 7'ye Ait Kod Dağılımı Grafiği 69
Şekil 5.14. QDA Miner Lite Soru 7'ye Ait Kod Bulutu Görseli 70
Şekil 5.15. QDA Miner Lite Soru 8'e Ait Kod Dağılımı Grafiği 73
Şekil 5.16. QDA Miner Lite Soru 8'e Ait Kod Bulutu Görseli 74
Şekil 5.17. QDA Miner Lite Soru 9'a Ait Kod Dağılımı Grafiği 77
Şekil 5.18. QDA Miner Lite Soru 9'a Ait Kod Bulutu Görseli 78
Şekil 5.19. DevOps Süreç Yönetimi Modeli - Use Case Diyagram 79
Şekil 5.20. DevOps Kalite Yönetimi Modeli - Use Case Diyagram 80
Şekil 5.21. DevOps Risk Yönetimi Modeli - Use Case Diyagram 80
Şekil 5.22. DevOps Gereksinim Yönetimi Modeli - Use Case Diyagram 81
Şekil 5.23. DevOps Takım Yönetimi Bileşeni - Use Case Diyagram 82
Şekil 6.1. DevOps Süreç Yönetimi 83
Şekil 6.2. DevOps Kalite Yönetimi 84
Şekil 6.3. DevOps Risk Yönetimi 85
Şekil 6.4. DevOps Gereksinim Yönetimi 86
Şekil 6.5. DevOps Takım Yönetimi 87
Şekil 6.6. DevOps Yapay Zeka Yönetimi 88

Sayfa

Şekil 6.7. DevOps Kalite Yönetimi Güncellenen Model 95
Şekil 6.8. DevOps Risk Yönetimi Güncellenen Model..... 95
Şekil 6.9. DevOps Gereksinim Yönetimi Güncellenen Model 96
Şekil 6.10. DevOps Takım Yönetimi Güncellenen Model..... 97

SİMGELER VE KISALTMALAR LİSTESİ

AI	Artificial Intelligence
BizDevOps	Business, Development, and Operations Integration
CD	Continuous Deployment
CI	Continuous Integration
CPPS	Cyber-Physical Production Systems
CPTM	Comprehensive Process-Tailored Model
DevOps	Development and Operations
DevSecOps	Development, Security, and Operations Integration
GitOps	Git-Based Operations
ISO/IEC 250nn (SQuaRE)	Systems and Software Quality Requirements and Evaluation
ISO/IEC/IEEE 29148	Systems and Software Engineering - Life Cycle Processes - Requirements Engineering
LLM	Large Language Model (Büyük Dil Modeli)
ML	Machine Learning
PMBOK® Guide	A Guide to the Project Management Body of Knowledge
PMI	Project Management Institute
PMI-ACP®	Project Management Institute - Agile Certified Practitioner
SCOPUS	Abstract and Citation Database for Peer-Reviewed Literature

1. GİRİŞ

Günümüzde yazılım geliştirme, teknolojik ilerlemelerin hız kazandığı bir dönemde, organizasyonlar için stratejik bir öncelik haline gelmiştir (Luz ve diğerleri, 2019). Dijital dönüşüm süreçlerinin yoğun şekilde yaşandığı günümüzde, teknolojik yeniliklere ayak uydurabilen organizasyonlar rekabet avantajı elde ederken, yazılım teslimat süreçlerinin hızlı, esnek ve güvenilir bir şekilde yönetilmesi, işletmelerin başarısını doğrudan etkileyen bir faktör haline gelmiştir (Mishra ve Otaiwi, 2020). Geleneksel yazılım geliştirme yöntemleri, geliştirme ve operasyon ekipleri arasında net bir ayrım yaparak iş süreçlerinde katı sınırlar oluşturmuş, bu durum hızlı değişikliklere uyum sağlama ve sorun çözüm süreçlerinde önemli kısıtlamalara yol açmıştır (Maroukian ve diğerleri, 2020). Geleneksel yöntemlerin bu sınırlamaları, yazılım geliştirme süreçlerinde daha dinamik ve bütüncül yaklaşımların benimsenmesini gerektirmişti (Bolscher ve diğerleri, 2019).

Bu bağlamda, yazılım geliştirme ve BT operasyonları arasındaki bariyerleri kaldırmayı amaçlayan DevOps (Development and Operations) yaklaşımı, modern yazılım mühendisliğinde bir paradigma değişikliği yaratmıştır (Plant ve diğerleri, 2024).

DevOps, sürekli entegrasyon (CI), sürekli teslimat (CD) ve altyapı otomasyonu gibi modern tekniklerle yazılım teslimatını hızlandırmayı ve operasyonel verimliliği artırmayı hedeflemektedir (Toh ve diğerleri, 2021). DevOps kültürü, yalnızca teknik araçlar ve süreçler değil, aynı zamanda organizasyonlar içerisinde iş birliği ve iletişim kültürünü de yeniden şekillendirir (Marnewick ve Langerman, 2020). Ancak, DevOps'un bu hızlı adaptasyonu sürecinde birçok organizasyon hem teknik hem de organizasyonel düzeyde önemli zorluklarla karşı karşıya kalmaktadır (Pérez-Sánchez ve diğerleri, 2024).

DevOps'un uygulama süreçlerinde karşılaşılan zorluklar, yazılım geliştirme projelerinin farklı alanlarında kendini göstermektedir (Rafi ve diğerleri, 2021). İhtiyaç yönetimi, takım dinamikleri, risk yönetimi, kalite yönetimi ve otomasyon gibi temel bileşenlerde yaşanan sorunlar, bu alanlara yönelik özel çözümler geliştirilmesini zorunlu kılmaktadır (Marnewick ve diğerleri, 2020). Literatürde, DevOps'un benimsenmesi sürecinde ekipler arası uyum eksikliği, iletişim problemleri ve geleneksel iş yapma yöntemlerinden kaynaklanan kültürel direnç gibi faktörler öne çıkmış, bu sorunları ele alan

çözüm önerilerinin yetersiz olduğu belirtilmiştir (Khattak ve diğerleri, 2023; Krey, 2022). Bunun yanında, savunma sanayi gibi yüksek hassasiyet ve karmaşıklık gerektiren sektörlerde, bu sorunların doğru bir şekilde ele alınması kritik bir önem taşımaktadır. Bu sektörlerde, yazılım geliştirme süreçlerinde kalite, güvenlik ve verimlilik gibi faktörler öncelikli hale gelmekte ve DevOps'un bu özelliklere uygun bir şekilde uyarlanması gerekmektedir (Mousaei ve Gandomani, 2020). Savunma sanayinde, yazılım geliştirme süreçlerinin yüksek güvenlik, hata toleransı ve sistem entegrasyonu gereksinimlerine uygun şekilde yönetilmesi, sektörel başarı için kritik bir zorunluluk haline gelmiştir.

Bu araştırma, savunma sanayinde faaliyet gösteren ve yüksek teknoloji ürünleri geliştiren bir kurumun yazılım geliştirme süreçlerine DevOps Proje Yönetim Modelinin uyarlanmasına yönelik bir yaklaşım geliştirmeyi amaçlamaktadır. Çalışma kapsamında, nitel bir araştırma metodolojisi benimsenerek, iki aşamalı bir süreç izlenmiştir. Birinci aşamada, yazılım geliştirme uzmanlarıyla yapılan yarı yapılandırılmış mülakatlar aracılığıyla ihtiyaç yönetimi, takım yönetimi, risk yönetimi, kalite yönetimi ve yapay zeka kullanımı gibi temel konularda veriler toplanmış ve bu veriler, dedüktif ve endüktif tematik analiz yöntemleriyle incelenmiştir. İkinci aşamada ise, elde edilen bulgular doğrultusunda UML diyagramları kullanılarak DevOps uyarlama modelleri geliştirilmiş ve bu modeller, işlevsel ve işlevsel olmayan gereksinimlere uygunluk açısından senaryo analizi yöntemiyle değerlendirilmiştir.

Bu tez çalışması, yalnızca savunma sanayi gibi yüksek hassasiyet gerektiren sektörlerde DevOps'un uygulanabilirliğini artırmayı değil, aynı zamanda yazılım mühendisliği ve proje yönetimi alanlarına da katkı sağlamayı hedeflemektedir. Araştırma bulgularının, DevOps uygulamaları sırasında karşılaşılan güçlüklerin üstesinden gelmek için yeni stratejiler sunacağı ve yazılım geliştirme süreçlerinin daha verimli bir şekilde yönetilmesine olanak sağlayacağı öngörülmektedir.

1.1. Problem

DevOps, yazılım geliştirme süreçlerinde verimliliği artırma ve hızlı teslimatı sağlamada büyük bir potansiyele sahip olmasına rağmen, bu yaklaşıma geçiş süreci birçok organizasyon için büyük zorluklar yaratmaktadır (Karunarathne ve diğerleri, 2024). DevOps'un gerektirdiği teknik araçların entegrasyonu ve bu yeni çalışma modelinin

organizasyon içinde benimsenmesi, hem teknik hem de organizasyonel açıdan çeşitli engelleri beraberinde getirmektedir (Toh ve diğerleri, 2019). Özellikle sürekli entegrasyon ve sürekli teslimat süreçlerinin uygulanması, altyapı otomasyonu, yazılım hatalarının hızlı tespit ve çözümü gibi teknik zorluklar, organizasyonların DevOps uygulamalarını tam anlamıyla benimsemesini engellemektedir (Krey, 2022). Ayrıca, organizasyonel düzeyde ekipler arası iş birliği eksikliği, iletişim problemleri ve geleneksel iş yapma yöntemlerinden DevOps'a geçişteki kültürel direnç önemli sorunlar olarak ortaya çıkmaktadır (Hamza ve diğerleri, 2023). Bu problemlerin çözümü, hem teknik hem de organizasyonel stratejilerle mümkün olacaktır, ancak literatürde bu zorlukları ele alan kapsamlı çözümler sınırlıdır.

Literatürde DevOps uygulamaları üzerine yapılan çalışmalar genellikle genel sektörler için geliştirilmiş yaklaşımlara odaklanmakta ve savunma sanayi gibi yüksek güvenlik, karmaşıklık ve hassasiyet gerektiren sektörler için sınırlı bilgi sunmaktadır. Bu bağlamda, savunma sanayinde DevOps uygulamalarının uyarlanabilirliği ve uygulanabilirliği konusundaki eksiklikler dikkat çekmektedir. Bu tez çalışması, bu sektöre özel ihtiyaçlara uygun DevOps adaptasyon modelleri geliştirerek, bu alandaki önemli bir boşluğu doldurmayı hedeflemektedir.

DevOps'un ihtiyaç yönetimi, takım dinamikleri, risk yönetimi ve kalite yönetimi gibi temel bileşenlerine yönelik literatürdeki mevcut çalışmalar, bu alanlarda karşılaşılan sorunların kapsamlı çözümlerle ele alınmasında yetersiz kalmaktadır. Özellikle, bu bileşenlerde ortaya çıkan sorunlara ilişkin sektörel ve uygulamaya dayalı çözüm önerilerinin eksikliği, savunma sanayi gibi karmaşık sektörlerde DevOps'un benimsenmesini zorlaştırmaktadır. Bu çalışmada, bu temel bileşenlere odaklanılarak, uygulamaya dönük çözüm önerileri ve sektör ihtiyaçlarına özgü yöntemler geliştirilmiştir.

DevOps'un yazılım geliştirme süreçlerine entegrasyonu üzerine yapılan çalışmalar genellikle teorik düzeyde kalmakta ve bu süreçlerin görsel temsili için UML gibi modelleme araçlarının kullanımına yeterince yer verilmemektedir. Bu tez, UML diyagramları kullanılarak geliştirilen DevOps uyarlama modelleri aracılığıyla teorik ve uygulamalı yaklaşımlar arasında bir köprü kurmayı amaçlamaktadır. Bu kapsamda, süreçlerin görsel olarak temsil edilmesi, önerilen modellerin hem anlaşılırlığını hem de uygulanabilirliğini artırmaktadır.

Eylem araştırması yöntemi, yazılım mühendisliği bağlamında pratik çözüm önerileri geliştirmek için etkili bir araç olsa da, savunma sanayinde DevOps adaptasyonuna ilişkin çalışmalarda bu yöntemin sınırlı bir şekilde kullanıldığı görülmektedir. Bu tez, eylem araştırması ve senaryo analizi yöntemlerini bir araya getirerek, DevOps'un savunma sanayine özgü uyarlanabilirliğini değerlendirirken uygulamalı ve çözüm odaklı bir yaklaşım sunmaktadır.

Son olarak, DevOps'un organizasyonel düzeyde adaptasyonu sürecinde karşılaşılan ekipler arası uyum eksikliği, kültürel direnç ve geleneksel yöntemlere bağlılık gibi sorunlar literatürde sıklıkla dile getirilmesine rağmen, bu sorunlara yönelik sektörel çözümlerin eksikliği dikkat çekmektedir. Bu tez, özellikle savunma sanayinde bu sorunları ele alan stratejiler geliştirerek, teknik ve kültürel zorlukların aşılmasına katkı sağlamayı amaçlamaktadır. Bu yönüyle çalışma, hem akademik hem de pratik düzeyde literatüre değerli bir katkı sunmaktadır.

1.2. Araştırmanın Amacı

Bu araştırmanın temel amacı, DevOps uygulamaları sırasında karşılaşılan teknik ve organizasyonel zorlukları kapsamlı bir şekilde analiz etmek ve bu zorlukların aşılmasına yönelik uygulanabilir çözüm önerileri sunmaktır. Bu bağlamda, DevOps'un yazılım geliştirme süreçlerine entegrasyonu sırasında ortaya çıkan temel engellerin belirlenmesi, bu engellerin hangi aşamalarda ve neden oluştuğunun tespit edilmesi ve literatürdeki mevcut çalışmaların yanı sıra yapılan analizler doğrultusunda çözüm önerilerinin geliştirilmesi hedeflenmektedir.

2. KURAMSAL ÇERÇEVE

Kurumsal DevOps uygulamalarında proje yönetimi, yalnızca teknik süreçlerin entegrasyonunu değil, aynı zamanda organizasyonel süreçlerin, takım dinamiklerinin ve yazılım geliştirme döngüsündeki stratejik yaklaşımların da uyumlu bir şekilde çalışmasını gerektirmektedir (Grande ve diğerleri, 2024). Bu bağlamda, DevOps süreçlerinin etkin yönetimi, farklı mühendislik disiplinleri ile proje yönetim yaklaşımlarının bir araya getirilmesini zorunlu kılmaktadır (Macarthy ve diğerleri, 2020). Çevik ve hızlı teslimat döngülerine uyum sağlarken aynı zamanda kalite, güvenlik ve sürdürülebilirlik hedeflerini gerçekleştirmek, disiplinler arası bir yaklaşımı gerekli kılmaktadır (Mousaei ve Gandomani, 2020).

Bu çerçevede kuramsal temel, tezde geliştirilen bütünleşik uyarlama modelinin yapı taşlarını oluşturan gereksinim yönetimi, yazılım kalite mühendisliği, risk yönetimi ve takım yönetimi konularını kapsamaktadır. Her bir bileşen, yazılım projelerinde kalite, güvenlik, verimlilik ve ekip iş birliğini optimize etmek amacıyla teorik temeller ve pratik uygulamalar sunmaktadır. Aşağıda bu bileşenler detaylı olarak ele alınmıştır.

2.1. Gereksinim Yönetimi

İhtiyaç mühendisliği, sistem veya yazılım geliştirme yaşam döngüsünde, bir sistemin veya yazılım bileşeninin gereksinimlerini tanımlayan ve yöneten disiplinler arası bir süreçtir (Uysal, 2022a). ISO/IEC/IEEE 29148 standardına göre gereksinim, bir sistem veya sistem bileşeninin bir sözleşme, standart veya başka resmi belgeleri karşılamak için yerine getirmesi gereken bir durum veya kapasite olarak tanımlanmaktadır. Gereksinimler, sistemin geliştirilmesi ve tamamlandıktan sonra kabulü için temel bir dayanak oluşturmaktadır (Giray, 2021).

İhtiyaç mühendisliği, kullanıcı gereksinimlerini fonksiyonel gereksinimler (functional requirements) ve fonksiyonel olmayan gereksinimler (non-functional requirements) olarak iki ana kategoriye ayırmaktadır (Amershi ve diğerleri, 2019). Fonksiyonel gereksinimler, bir sistemin gerçekleştirmesi gereken işlevleri tanımlarken, fonksiyonel olmayan gereksinimler, sistemin bu işlevleri hangi performans ve kalite standartlarında gerçekleştireceğini

belirtmektedir (Uysal, 2022b). Gereksinimlerin doğru bir şekilde tanımlanması, yazılımın hem teknik hem de kullanıcı beklentilerini karşılama yeteneği açısından kritik öneme sahiptir (Nalchigar ve Keshavjee, 2021).

İhtiyaç mühendisliği süreci, ISO/IEC/IEEE 29148 standardına göre aşağıdaki aşamaları içermektedir (Uysal, 2022b):

1. Gereksinim Toplama (Requirements Elicitation): Paydaşlarla etkileşim kurarak ihtiyaçların belirlenmesi ve ilgili belgelerden, düzenlemelerden veya mevcut sistemlerden bilgi alınması.
2. Gereksinim Analizi (Requirements Analysis): Problem ve çözüm alanlarının yapısının anlaşılması ve bunların doğasının ve ilişkilerinin analiz edilmesi.
3. Gereksinim Modelleme (Requirements Modeling): Sistem bileşenlerinin basit ve bütüncül bir şekilde anlaşılmasını sağlamak için modelleme yapılması.
4. Gereksinim Müzakeresi (Requirements Negotiation): Çakışan veya çelişen gereksinimlerin tespit edilerek uzlaşma sağlanması.
5. Gereksinim Önceliklendirme (Requirements Prioritization): En kritik ve değerli gereksinimlerin belirlenmesi.
6. Gereksinim Doğrulama (Requirements Validation): Gereksinimlerin paydaşlar tarafından tanımlandığı şekilde doğru olup olmadığının kontrol edilmesi.
7. Gereksinim Geçerleme (Requirements Verification): Doğrulanmış gereksinimlerin ele alındığını, karşılandığını ve doğru bir şekilde tanımlandığını garanti edilmesi.
8. Gereksinim İncelemesi (Requirements Review): Gereksinimlerin geçerliliğinin, tutarlılığının ve doğruluğunun gözden geçirilmesi.
9. Gereksinim Belirtimi (Requirements Specification): Üzerinde mutabık kalınmış gereksinimlerin belgelenmesi.
10. Sürekli Süreç (Continuous Process): Gereksinimlerin bir sistemin, ürünün veya hizmetin yaşam döngüsü boyunca tanımlanmasını, belgelenmesini, korunmasını, iletilmesini ve izlenmesini sağlayan faaliyetleri içeren bir süreçtir.

Gereksinim yönetimi, problem alanı (kullanıcı ihtiyaçları ve kullanım senaryoları) ile çözüm alanı (sistem tasarımı ve geliştirme) arasında köprü kurmaktadır (Lee, 2019). Paydaşların gereksinimlerini karşılamak amacıyla sistem gereksinimlerine dönüştürülmesi

süreci, sistemin tasarım ve geliştirme aşamalarını desteklemektedir (Uysal, 2021). İyi tanımlanmış gereksinim özellikleri arasında tamlık, geçerlilik, doğrulanabilirlik, tutarlılık, belirsizlikten uzaklık, değiştirilebilirlik ve izlenebilirlik bulunmaktadır (Unhelkar, 2018).

DevOps süreçlerinde, gereksinim yönetimi yalnızca başlangıç aşamalarında değil, yazılım geliştirme yaşam döngüsünün her aşamasında önemli bir rol oynamaktadır (Kumeno, 2019). Hızlı değişimlere yanıt verebilme yeteneği, gereksinim mühendisliğinin esnek ve sürekli bir yapıya sahip olmasını gerektirmektedir (Bourque ve Richard, 2014). Ayrıca, fonksiyonel olmayan gereksinimlerin, özellikle performans ve güvenlik standartlarının, sürekli entegrasyon (CI) ve sürekli teslimat (CD) süreçlerine entegrasyonu, yazılım kalitesi üzerinde doğrudan etkili olmaktadır (Ishikawa, 2019).

Kullanılan Yaklaşımlar

1. Use Case (Kullanım Senaryoları); gereksinimlerin, sistemin kullanıcılarla nasıl etkileşime geçtiğini betimleyen senaryolarla modellenmesidir.
2. User Story (Kullanıcı Hikayeleri); çevik yöntemlerde kullanılan, gereksinimlerin kısa ve anlaşılır ifadelerle tanımlandığı bir yaklaşımdır.

Gereksinim yönetimi, yazılım projelerinde müşteri memnuniyetini artırmak ve ürünün kaliteli bir şekilde teslim edilmesini sağlamak için temel bir öncül oluşturmaktadır (Villamizar ve Kalinowski, 2021). DevOps ortamında, sürekli değişen gereksinimlerin doğru bir şekilde ele alınması, yazılım geliştirme süreçlerinin başarısı açısından kritik bir gereklilik olmaktadır (Uysal, 2022a).

2.2. Yazılım Kalite Mühendisliği (SQUARE Standardı)

Yazılım kalite mühendisliği, yazılım projelerinde kalite gereksinimlerini belirlemek, bu gereksinimlerin karşılanmasını sağlamak ve yazılımın performansını değerlendirmek için kullanılan bir disiplindir (Uysal ve Mergen, 2015). ISO/IEC 250nn (SQuaRE) - Sistem ve Yazılım Kalite Gereksinimleri ve Değerlendirme, bu alanda uluslararası bir standart sağlayarak yazılım geliştirme süreçlerinde kalite yönetimini desteklemektedir (Esaki ve diğerleri, 2013).

SQuaRE standardı, yazılım kalitesini değerlendirmek ve iyileştirmek için aşağıdaki bileşenleri içermektedir:

1. Kalite Yönetimi (Quality Management): Sistem ve yazılım kalite ihtiyaçlarının belirlenmesi ve yönetimi için rehber ilkeler sunmaktadır.
2. Kalite Modelleri (Quality Models): Yazılımın iç kalite, dış kalite ve kullanım kalitesi gibi farklı boyutlarda değerlendirilmesini sağlamaktadır.
3. Kalite Ölçümü (Quality Measurement): Kalite ölçüm elemanları ve metrikler aracılığıyla kalite gereksinimlerini sayısal olarak değerlendirmektedir.
4. Kalite Gereksinimleri (Quality Requirements): Yazılım kalite ihtiyaçlarının açık bir şekilde tanımlanmasını sağlamaktadır.
5. Kalite Değerlendirme (Quality Evaluation): Yazılımın kalite özelliklerini değerlendirmek için kullanılan yöntemleri belirlemektedir.

SQuaRE çerçevesi, yazılım ürünlerinin iç kalite, dış kalite ve kullanım kalitesine ilişkin kriterleri kapsamlı bir şekilde ele almaktadır:

1. İç Kalite (Internal Quality): Yazılımın kaynak kodu gibi, geliştiricilere yönelik özelliklerini içermektedir.
2. Dış Kalite (External Quality): Kullanıcıların yazılımı test ve deneme ortamlarında kullanımına dayalı özelliklerdir.
3. Kullanım Kalitesi (Quality in Use): Yazılımın gerçek çalışma ortamındaki performansını değerlendirmektedir.

SQuaRE, yazılım projelerinde kalite değerlendirmesi için çeşitli ana ve alt karakteristikler sunmaktadır:

1. Ana Karakteristikler: (Bakım yapılabilirlik (maintainability), taşınabilirlik (portability) ve güvenilirlik (reliability) vb).
2. Alt Karakteristikler: Tekrar kullanılabilirlik (reusability) ve anlaşılırlık (understandability) vb).

Örneğin, bir yazılımın Bakım Yapılabilirlik özelliği, alt karakteristiklerden Tekrar Kullanılabilirlik ile ölçülebilir (Azuma, 2011). Bu kapsamda, Kalıtım Ağacı Derinliği

(Depth of Inheritance Tree - DIT) gibi metrikler, yazılımın iç kalitesini değerlendirmek için kullanılan ölçüm birimlerinden biridir (Komiyama ve diğerleri, 2020).

SQuaRE standardı, yazılım kalite gereksinimlerini sistematik bir şekilde belirlemek ve sürekli entegrasyon (CI) ile sürekli teslimat (CD) süreçlerinde kalite güvencesi sağlamak için kullanılabilmektedir (Esaki ve diğerleri, 2012):

1. Dinamik Ortamlar: DevOps gibi hızlı geliştirme döngüleri içinde kalite kriterlerini sürekli izleme ve değerlendirmektedir.
2. Kalite Güvencesi (QA): SQuaRE, test süreçlerini optimize etmek için kalite metriklerini kullanmaktadır.

SQuaRE standardının DevOps süreçlerine entegrasyonu, yazılım geliştirme döngüsünün her aşamasında kaliteyi korumak ve geliştirmek için etkili bir araçtır (Uysal ve Mergen, 2015). Özellikle savunma sanayi gibi yüksek güvenlik ve hassasiyet gerektiren projelerde, bu standardın uygulanması, yazılımın güvenilirliğini ve sürdürülebilirliğini artırmada kritik bir rol oynamaktadır (Komiyama ve diğerleri, 2020).

2.3. Risk Yönetimi

Risk yönetimi, yazılım projelerinde karşılaşılabilecek belirsizliklerin tanımlanması, değerlendirilmesi, önceliklendirilmesi ve kontrol edilmesi süreçlerini kapsayan sistematik bir yaklaşımdır (Laukkarinen ve diğerleri, 2018). PMI, riski, "gerçekleştiğinde zaman, maliyet, kapsam veya kalite açısından en az bir proje hedefini olumlu ya da olumsuz etkileyen belirsiz olay ve durum" olarak tanımlamaktadır. Bu kapsamda, riskler problemlerden farklı olarak gelecekte oluşabilecek durumların olasılıklarını temsil eder ve bu nedenle hem tehditler hem de fırsatlar içerebilmektedir (Laukkarinen ve diğerleri, 2017).

Riskler, Muñoz ve diğerleri (2019)'ne göre kaynağına ve etkisine göre çeşitli kategorilere ayrılmaktadır:

1. Proje Yönetimi Riskleri: Projenin yönetiminden kaynaklanan yetersiz ekip, gerçekçi olmayan hedefler veya esnek olmayan zaman çizelgeleri gibi faktörleri içermektedir.

2. Teknik Riskler: Teknoloji ile ilgili karmaşıklık, yeni veya bilinmeyen araçlar, yetersiz yazılım veya donanım gibi teknik unsurlardan kaynaklanmaktadır.
3. Kurumsal Riskler: Organizasyonel yapı, yönetim desteği veya proje yönetim kültürü gibi kuruma özgü risklerdir.
4. Dış Kaynaklı Riskler: Yasal düzenlemeler, ekonomik koşullar ve tedarik süreçlerinden kaynaklanan risklerdir.

Risk yönetim süreci, Boehm ve DeMarco (1997)'ye göre proje ömrü boyunca yinelemeli olarak uygulanması gereken beş temel aşamadan oluşmaktadır:

1. Risklerin Belirlenmesi ve Tanımlanması: Projeye etkisi olabilecek tüm risklerin belirlenmesi, önceki projelerden elde edilen deneyimler, SWOT analizi ve uzman değerlendirmesi gibi yöntemlerle yapılmaktadır (Sinha ve diğerleri, 2020).
2. Risklerin Analizi ve Değerlendirilmesi: Risklerin etkilerinin ve gerçekleşme olasılıklarının nitel ve nicel yöntemlerle değerlendirilmesidir. Örneğin, Risk Matrisi ve Beklenen Parasal Değer (BPD) yöntemleri bu aşamada kullanılmaktadır.
3. Risklerin Önceliklendirilmesi: Risklerin, projeye etkileri ve öncelikleri doğrultusunda sıralanmasıdır.
4. Risklerin Planlanması ve Önlemler: Risklerin etkisini azaltmak veya ortadan kaldırmak için alınacak tedbirlerin planlanmasıdır. Bu süreçte riski kabullenme, azaltma, transfer etme veya kaçınma gibi stratejiler uygulanabilmektedir.
5. Risklerin İzlenmesi ve Kontrolü: Proje boyunca risklerin güncellenmesi, izlenmesi ve alınan önlemlerin etkinliğinin değerlendirilmesi için Risk Burndown Chart gibi araçlar kullanılmaktadır.

Scrum proje yönetimi çerçevesi, risklerin erken tespiti ve yönetimi için çeşitli yöntemler sunmaktadır (Hammad ve Inayat, 2018).

1. Daily Scrum ve Sprint Review: Günlük toplantılar ve sprint değerlendirmeleri, projedeki olası problemleri erken teşhis etmek için kullanılmaktadır.
2. Sprint Retrospective: Her sprint sonunda proje yönetim sürecinin sürekli iyileştirilmesini sağlamaktadır.

3. Product Backlog: Risklerin yönetiminde kullanılan ana araçlardan biridir. Riskler, öncelik sırasına göre sıralamak ve etkilerinin azaltılması için sprint planlamalarına entegre etmektir.

2.3.1. Risk yönetiminde analiz teknikleri

1. Nitel Risk Analizi: Risklerin genel terimlerle (çok yüksek, yüksek, orta, düşük vb.) değerlendirilmesini sağlamaktadır ve önceliklendirme amacıyla kullanılmaktadır (Alharbi ve Qureshi, 2014).
2. Nicel Risk Analizi: Matematiksel yöntemlerle risklerin etkilerini sayısal olarak hesaplamaktadır. Beklenen Parasal Değer (BPD) ve Karar Ağacı gibi teknikler bu aşamada kullanılmaktadır (Hossain ve diğerleri, 2009).
3. Simülasyon ve Modelleme: Monte Carlo gibi tekniklerle, risk faktörlerinin olası sonuçlarını tahmin etmek için simülasyonlar gerçekleştirilmektedir (Lunesu ve diğerleri, 2021)

DevOps süreçlerinde risk yönetimi, hız ve kaliteyi dengeleyerek, proje hedeflerinin güvence altına alınmasını sağlamaktır (Mousaei ve Gandomani, 2018). Hızlı teslimat döngülerinde risklerin etkin bir şekilde yönetilmesi, sistem arızaları, güvenlik açıkları ve kullanıcı şikayetlerinin önüne geçmek açısından kritik öneme sahip olmaktadır.

2.3.2. Teorik çerçeve ve yaklaşımlar

1. ISO 31000: Risk yönetimi süreçleri için uluslararası bir standart olarak kullanılan bu çerçeve, risklerin tanımlanması, değerlendirilmesi ve kontrol edilmesi için metodolojik bir yaklaşım sunmaktadır.
2. FMEA (Failure Mode and Effects Analysis): Sistematik bir yöntem olarak FMEA, potansiyel hataların tespit edilmesi ve önceliklendirilmesini sağlamaktadır.

Savunma sanayi projelerinde, güvenlik ve operasyonel risklerin yüksekliği nedeniyle risk yönetimi daha karmaşık ve kritik bir süreçtir. Bu nedenle, risk yönetiminin DevOps süreçlerine entegrasyonu, hem güvenlik hem de operasyonel verimlilik açısından önemli bir avantaj sağlamaktadır.

2.4. Takım Yönetimi

Takım yönetimi, bir projenin başarıyla tamamlanması için gerekli olan insan kaynaklarının etkili bir şekilde yönetilmesini, yönlendirilmesini ve koordinasyonunu içermektedir (Boss, 1983). PMI (Project Management Institute), Proje Yönetimi Bilgi Birikimi Kılavuzu (PMBOK® Guide) çerçevesinde, takım yönetimini proje yönetiminin vazgeçilmez bir bileşeni olarak ele almaktadır ve bunu "kaynak yönetimi" süreç grubu içinde konumlandırmaktadır. Takım yönetimi, yalnızca görevlerin ve rollerin dağıtılması değil, aynı zamanda ekip üyeleri arasında etkili iletişim, motivasyon ve iş birliğinin sağlanmasını da içermektedir (Poston ve Richardson, 2012). Bu süreç, ekip performansını optimize etmek ve projenin belirlenen kapsam, zaman ve bütçe hedeflerine ulaşmasını sağlamak amacıyla stratejik yaklaşımlar gerektirmektedir (Cervone, 2007).

DevOps bağlamında, PMI'ın belirttiği takım yönetimi ilkeleri, geleneksel proje yönetiminden farklılaşan bazı unsurları içermektedir. DevOps, geleneksel silo yapılarının değişmesini ve geliştirici (Dev) ile operasyon (Ops) ekiplerinin birleştirilmesini teşvik etmektedir (Raj ve Sinha, 2015). PMI'a göre bu tür yapısal dönüşümler, organizasyonel süreç varlıklarının yeniden değerlendirilmesini ve paydaş yönetiminin iyileştirilmesini gerektirmektedir (Carturan ve Goya, 2019). Takımlar arasında yüksek düzeyde iş birliği ve uyum sağlanması, yalnızca teknik süreçleri değil, aynı zamanda organizasyonel kültürü de dönüştürmektedir (Wiedemann ve Krcmar, 2019).

Takım yönetiminde aşağıdaki süreçler öne çıkmaktadır:

- Takım Planlaması: Proje yöneticisinin kaynak ihtiyaçlarını belirleyerek ekip üyelerinin seçimi ve görevlendirilmesidir. DevOps bağlamında, bu süreç yalnızca bireysel becerilere değil, aynı zamanda ekip üyelerinin birlikte çalışabilme yetkinliğine odaklanmaktadır.
- Takım Geliştirme: PMI, ekip üyelerinin yetkinliklerinin artırılması ve takım dinamiklerinin güçlendirilmesi için eğitim, mentorluk ve iş birliği etkinliklerini önermektedir. Çevik yaklaşımlarda olduğu gibi DevOps süreçlerinde de iteratif bir şekilde öğrenme ve gelişim ön planda olmaktadır.

- Takım Yönetimi: Takım üyeleri arasındaki çatışmaların yönetilmesi, motivasyonun artırılması ve performansın izlenmesi, PMI'ın kaynak yönetimi sürecinde kritik öneme sahiptir. DevOps kültüründe, sürekli entegrasyon ve sürekli teslimat süreçlerinde takımın uyumlu çalışması, yüksek performansın temel göstergelerinden biridir.

Scrum ve Kanban gibi çevik yöntemler, DevOps süreçleriyle bütünleşerek yüksek performanslı ekiplerin oluşturulmasını desteklemektedir. DevOps süreçlerinde, iterasyonların hızlı bir şekilde tamamlanması ve sürekli teslimat süreçlerinin etkin yönetimi, yalnızca teknolojik araçların değil, aynı zamanda ekiplerin iş birliği düzeyinin artırılmasıyla mümkün olmaktadır (Ambler ve Lines, 2020).

Bu bağlamda, PMI'ın kaynak yönetimi süreçlerine entegre edilen araçlar ve teknikler şu şekilde öne çıkmaktadır:

- İş Birliğini Destekleyen Araçlar: Trello, Confluence gibi proje yönetimi araçları, takım üyeleri arasında bilgi paylaşımını kolaylaştırmakta ve süreçlerin izlenebilirliğini artırmaktadır. PMI, projelerde iletişim ve iş birliğinin, başarının anahtarı olduğunu belirtmektedir.
- Performans Ölçümü: Ekip performansının düzenli olarak ölçülmesi ve iyileştirilmesi için metriklerin tanımlanması, DevOps bağlamında kritik bir unsur olarak karşımıza çıkmaktadır. Bu metrikler, PMI'ın proje yönetişim ilkeleriyle uyumlu bir şekilde belirlenmektedir.

Savunma sanayi gibi yüksek hassasiyet gerektiren sektörlerde, PMI'ın belirttiği takım yönetimi prensiplerinin uygulanması büyük önem taşımaktadır. Bu sektörlerde, süreçlerin karmaşıklığı ve hataya toleransın düşük olması nedeniyle, ekiplerin uyumlu bir şekilde çalışması yalnızca proje başarı oranını artırmaz, aynı zamanda operasyonel etkinliği ve güvenilirliği desteklemektedir. DevOps süreçlerinin savunma projelerine entegrasyonu, PMI'ın önerdiği gibi sıkı bir planlama, etkili bir koordinasyon ve sürekli izleme süreçlerini gerektirmektedir.

Sonuç olarak, PMI'ın takım yönetimi perspektifi, DevOps süreçlerinin etkili bir şekilde uygulanmasında önemli bir rehber sunmaktadır. Takımların iş birliği, koordinasyon ve performansını optimize eden stratejik yaklaşımlar, yalnızca yazılım projelerinin başarısına değil, aynı zamanda uzun vadeli organizasyonel dönüşüme de katkı sağlamaktadır (Banica ve diğerleri, 2017). Bu bağlamda, DevOps ve PMI ilkelerinin birlikte ele alınması, özellikle kompleks ve hassas sektörlerde, projelerin başarısını artıran kritik bir yaklaşım sunmaktadır (Ambler ve Lines, 2020).

3. İLGİLİ ÇALIŞMALAR

DevOps, Çevik ve Bulut Bilişim gibi modern yazılım geliştirme metodolojileri, son yıllarda yazılım mühendisliği alanında hızla popülerlik kazanmıştır (Toh ve diğerleri, 2021). Bu metodolojiler, iş birliğini artırmak, sürüm döngülerini hızlandırmak ve yazılım kalitesini iyileştirmek amacıyla kullanılmaktadır. Özellikle DevOps, yazılım geliştirme süreçlerinde etkinlik, hız ve kaliteyi artırmayı hedefleyerek, yazılım mühendisliğinde önemli bir paradigma değişikliğine öncülük etmiştir (Khan ve diğerleri, 2022).

Mevcut literatürde, DevOps ve ilgili kavramların yazılım mühendisliği üzerindeki etkilerini inceleyen çok sayıda çalışma bulunmaktadır. DevOps, geliştirme (Development) ve operasyon (Operations) ekipleri arasındaki engelleri kaldırmayı hedefleyen bir yaklaşım olarak, yazılım teslimat süreçlerini hızlandırırken daha yüksek bir kalite standardı sağlamayı amaçlamaktadır (Karunarathne ve diğerleri, 2024). Aynı zamanda organizasyonlara daha esnek ve yenilikçi süreçler sunmakta, böylece yazılım geliştirme ve operasyon ekipleri arasındaki iş birliğini güçlendirmektedir (Mousaei ve Gandomani, 2020).

Bu literatür taraması, DevOps'un kültürel boyutları, teknik zorlukları, metodolojik entegrasyonu ve eğitim-öğretimle ilgili sorunlarını kapsamlı bir şekilde ele almaktadır. DevOps'un farklı yönlerini tematik bir yaklaşımla değerlendirerek, bu alandaki bilgi birikimine katkıda bulunmayı hedeflemektedir. Bu çerçevede, kültürel faktörlerden teknik engellere, metodolojilerin entegrasyonundan eğitsel ihtiyaçlara kadar çeşitli boyutlar derinlemesine incelenmiştir.

DevOps'un benimsenmesi sürecinde kültürel faktörler ve insan davranışları kritik bir rol oynamakta olup organizasyonlar için hem fırsatlar hem de zorluklar barındırmaktadır. Khattak ve diğerleri (2023), DevOps'un yazılım mühendisliği süreçlerinde benimsenmesi üzerine gerçekleştirdikleri literatür taramasında, güçlü liderlik, iş birliği ve eğitim programlarının DevOps'un başarılı bir şekilde uygulanması üzerindeki etkilerini araştırmışlardır. Çalışma, liderliğin ve organizasyonel desteğin DevOps'un yaygınlaşmasındaki kritik rolüne vurgu yapmaktadır. Aynı zamanda DevOps'un kültürel bir değişim gerektirdiği ve bu değişimin başarılı bir şekilde yönetilmesi gerektiği belirtilmiştir. Araştırmalar, iş birliğini teşvik eden

stratejilerin organizasyonel değişimi hızlandırabileceğini ve insan odaklı stratejilerin önceliklendirilmesi gerektiğini ifade etmişlerdir.

Benzer şekilde, Rana ve diğerleri (2024), DevOps'un insan faktörleri ile olan karmaşık ilişkisini incelemiş ve bu bağlamda 59 farklı insan faktörünü tanımlayarak, bu faktörlerin DevOps süreçlerini nasıl şekillendirdiğini açıklamışlardır. Bu çalışma, DevOps'un sürdürülebilir bir şekilde uygulanabilmesi için insan odaklı bir yaklaşım benimsenmesi gerektiğini ve ekipler arasında güven ve iletişim eksikliğinin kültürel engellerin başında geldiğini göstermiştir.

DevOps'un uygulanabilirliğini değerlendiren bir diğer çalışmada, Jayakody ve Wijayanayake (2023), DevOps süreç iyileştirme çerçevesi önererek olgunluk modellerinin güçlü ve zayıf yönlerini analiz etmiştir. Çalışma, DevOps'un yazılım teslimatında maliyet verimliliğini artırdığına dikkat çekmiştir (Jayakody & Wijayanayake, 2023).

Toh, Sahibuddin ve Bakar (2021), DevOps uygulamalarının Sürekli Teslimat (CD) süreçleri üzerindeki etkilerini araştırarak, bu süreçlerin yazılım geliştirme döngüsünü hızlandırdığını ve iş birliği düzeyini artırdığını belirtmiştir. DevOps'un Sürekli Teslimat (CD) süreçlerine entegrasyonunda araçlar ve teknoloji, kültür, mimari ve süreç gibi dört temel faktörü tanımlamıştır. Araştırma, DevOps'un benimsenmesinde bu faktörlerin dikkate alınmasının kritik olduğunu vurgulamaktadır (Toh, Sahibuddin ve Bakar, 2021). Bu çalışma, Scopus veritabanından elde edilen 24 makaleyi incelemiş ve DevOps'un CD entegrasyonunda karşılaşılan temel zorlukları vurgulamıştır. Benzer şekilde, Österberg (2020), DevOps'un tanımına dair fikir birliğinin eksik olduğunu, ancak benimsenmesinin yazılım teslim sürelerini kısaltarak kaliteyi artırdığını belirtmiştir.

Khan ve diğerleri (2022), DevOps kültürünü benimsemenin organizasyonel değişim süreçlerinde nasıl etkili olduğunu incelemiştir. Çalışmada, değişime direncin, kültürel adaptasyonun önündeki en büyük engellerden biri olduğu vurgulanmıştır. Ayrıca, liderlik desteğinin ve uygun eğitim programlarının bu süreçlerde kritik öneme sahip olduğu belirtilmiştir.

DevOps'un benimsenmesi sırasında karşılaşılan zorluklar literatürde geniş bir şekilde ele alınmıştır. Krey (2022), DevOps'un benimsenmesindeki engelleri teknik ve kişiler arası

zorluklar olarak ikiye ayırmıştır. Araştırmada, iletişim eksikliği, kültürel direnç ve eski sistemlerin karmaşıklığı gibi sorunlar öne çıkmaktadır (Krey, 2022). Ayrıca, Nayanajith ve Wickramarachchi (2024), gelişmekte olan ülkelerde DevOps'un benimsenmesini engelleyen temel faktörler arasında teknik altyapı eksiklikleri, beceri yetersizliği ve düzenleyici uyumluluk sorunlarını sıralamıştır. Çalışma, küçük ölçekli uygulamaların başlatılmasının ve otomasyonun teşvik edilmesinin bu zorlukların aşılmasına yardımcı olabileceğini önermektedir (Nayanajith & Wickramarachchi, 2024).

DevOps'un benimsenmesi, organizasyonel ve teknik değişimlerin eş zamanlı olarak ele alınmasını gerektiren karmaşık bir süreçtir. Literatürde, liderlik desteği, iş birliğini teşvik eden kültürel stratejiler ve sürekli eğitim programlarının DevOps'un başarıyla uygulanmasını desteklediği vurgulanmaktadır. Ayrıca, otomasyon süreçlerinin geliştirilmesi ve uygun araçların seçimi, DevOps'un etkili bir şekilde benimsenmesi için temel unsurlar arasında yer almaktadır. Gelecekteki çalışmalar, farklı sektörlerdeki DevOps uygulamalarını derinlemesine inceleyerek bu sürecin etkilerini daha geniş bir perspektiften değerlendirebilir.

DevOps'un başarısı, büyük ölçüde organizasyonel kültür ve insan faktörlerine bağlıdır. Pérez-Sánchez ve diğerleri (2024), DevOps ile ilişkili 59 insan faktörünü tanımlayarak, bu faktörlerin DevOps'un benimsenmesindeki etkilerini analiz etmiştir. Araştırma, kültürel dönüşüm ve iş birliğinin DevOps süreçlerinin etkinliği üzerindeki önemini vurgulamaktadır (Pérez-Sánchez et al., 2024). Aynı şekilde, Muyet ve Sahibuddin (2022), liderliğin ve iş birliğinin DevOps'un başarısında belirleyici rol oynadığını ortaya koymuştur (Muyet ve Sahibuddin, 2022).

Leite ve diğerleri (2019), DevOps'un kavramsal çerçevesini ve yazılım geliştirme süreçlerindeki zorlukları sistematik olarak ele almıştır. Araştırma, DevOps'un hızlı teslimat döngüleri ve sürekli entegrasyon avantajlarına rağmen, organizasyonel ve teknik engellerin var olduğunu göstermiştir. Çalışmalarında, DevOps'un organizasyonel entegrasyonu sırasında yaşanan sorunlara değinilmiş ve ekipler arası iletişimi güçlendirmenin önemine vurgu yapılmıştır. El Aouni ve diğerleri (2024), Çevik, Bulut Bilişim ve DevOps entegrasyonunun iş birliği, üretkenlik ve yazılım kalitesi üzerindeki etkilerini araştırmıştır. Bu çalışma, özellikle bu üç metodolojinin bir araya gelmesinin yazılım mühendisliğine sunduğu yeni iş akışları ve fırsatları incelemektedir. Çalışmanın bulguları, doğru araçların

ve süreçlerin seçilmesinin entegrasyonun başarısı için kritik olduğunu göstermektedir. El Aouni ve diğerleri (2024), DevOps'un Çevik ve Bulut metodolojileriyle entegre edilmesinin, yazılım teslim süreçlerini hızlandırdığına ancak altyapı karmaşıklığını artırdığına dikkat çekmiştir. Çalışma, bu zorlukların üstesinden gelebilmek için iş birliğini artırmaya yönelik stratejiler önermektedir.

DevOps kültürü yalnızca teknik süreçlerle sınırlı kalmayıp, zihniyet değişimini de gerektirmektedir. Jha ve diğerleri (2023), DevOps bakış açısını anlamaya yönelik yapılan çalışmalar sonucunda, kültürel değişimin ekiplerin iş birliğini ve verimliliğini artırdığını belirtmiştir. Çalışma, SCOPUS veri tabanı kullanılarak kapsamlı bir literatür taramasıyla desteklenmiştir.

DevOps kültürünün yazılım geliştirme ekiplerinde başarılı bir şekilde benimsenmesi, organizasyonel süreçlerin uyumuna bağlı olmaktadır. Sravan ve diğerleri (2023), yazılım geliştirme ve operasyon ekipleri arasında iş birliğini artırmak için liderlik ve güvenin önemine dikkat çekmiştir.

El Aouni ve diğerleri (2024), Çevik, Bulut ve DevOps entegrasyonunun sağladığı avantajları detaylı bir şekilde ele alarak, iş birliği ve süreç iyileştirme üzerinde odaklanmıştır. Aynı zamanda, Eramo ve diğerleri (2024), sürekli yazılım mühendisliği yaklaşımlarının yazılım performansı üzerindeki olumlu etkilerini vurgulamıştır. Öte yandan, Omer ve diğerleri (2020), mikroservis mimarilerinin DevOps süreçlerine entegrasyonunda karşılaşılan güvenlik, ölçeklenebilirlik ve dayanıklılık gibi teknik engelleri derinlemesine incelemiş ve operasyonel verilerden yararlanılarak bu sorunlara çözüm önerileri geliştirmiştir. Özellikle, gerçek zamanlı verilerin etkin kullanımı ile operasyon süreçlerinin daha verimli hale getirilebileceği vurgulanmıştır.

Mikro servis mimarileri ve GitOps uygulamaları, DevOps süreçlerinin iyileştirilmesinde öne çıkan teknolojilerdir. Koren ve diğerleri (2023), ultra kısa darbe lazer üretim sistemlerinde mikro hizmetlerin etkin kullanımını analiz etmiş ve GitOps'un süreçleri nasıl dönüştürdüğünü göstermiştir. Ayrıca, Antunes ve Tate (2024), BizDevOps'un iş birimlerini ve yazılım geliştirme ekiplerini bir araya getirme konusundaki rolünü incelemişlerdir.

Çok küçük işletmelerde DevOps'un uygulanabilirliği üzerine yapılan çalışmalar, bu süreçlerin ölçeklendirilmesiyle ilgili zorlukları ortaya koymaktadır. Acevedo-Dueñas ve diğerleri (2023), bu zorlukları sistematik bir haritalama metodolojisiyle ele almış ve küçük işletmelere yönelik özel DevOps rehberlerinin geliştirilmesini önermiştir.

Savunma sanayi, yüksek risk ve kalite gereksinimleriyle DevOps'un özelleştirilmiş uygulamalarına ihtiyaç duymaktadır. Literatürde, CPPS (Cyber-Physical Production Systems) gibi alanlarda mikro servislerin kullanımına dair öneriler sunulmuştur (Koren ve diğerleri, 2023). Ayrıca, performans mühendisliği ve kalite kültürü, savunma sanayinde DevOps'un etkili bir şekilde uygulanması için kritik bileşenlerdir (Eramo ve diğerleri, 2024).

Zhang ve diğerleri (2020), makine öğrenimi (ML) tabanlı sistemlerin test edilmesi üzerine sistematik bir literatür taraması gerçekleştirmiştir. Braiek ve Khomh (2020) ise Zhang ve diğerlerinden daha küçük bir veri seti kullanarak, ML sistemlerinin test edilmesinde karşılaşılan zorlukları, mevcut çözümleri ve literatürdeki boşlukları ele almışlardır. Bu çalışmalar, test süreçlerinin DevOps ile entegre edilmesinde teknik engellerin olduğunu göstermektedir. Lwakatare ve diğerleri (2020) ise endüstriyel ortamlarda büyük ölçekli ML tabanlı sistemlerin geliştirilmesi sırasında karşılaşılan zorlukları kapsamlı bir şekilde incelemiştir.

Masuda ve diğerleri (2019) tarafından yapılan çalışmalar, DevOps süreçlerinde güvenlik unsurlarına dikkat çekmiştir. Bu çalışmalar, güvenlik ve emniyet gibi kritik unsurların DevOps süreçlerinde nasıl ele alındığını ve geliştirildiğini incelemiştir.

Yapılan incelemeler sonucunda, DevOps ve Çevik (Agile) metodolojilerinin yazılım geliştirme süreçlerini olumlu yönde etkilediği, ancak uygulanmasında karşılaşılan zorlukların devam ettiği ortaya çıkmıştır. Özellikle güvenlik, süreç yönetimi ve takım iş birliği konularında önemli boşluklar tespit edilmiştir. Toh ve diğerleri (2021), DevOps'un CD süreçlerine entegrasyonunda karşılaşılan zorlukları ele alırken, bu entegrasyonun iş birliği ve yazılım kalitesini artırdığını, ancak güvenlik unsurlarının yeterince dikkate alınmadığını belirtmiştir. Leite ve diğerleri (2019), DevOps'un benimsenmesi sırasında organizasyonel ve teknik zorlukların varlığını vurgulamış ve güçlü liderlik ile iş birliği süreçlerinin önemine dikkat çekmiştir. El Aouni ve diğerleri (2024), Agile ve Bulut

Bilişim'in DevOps ile entegrasyonunun iş birliği ve yazılım kalitesini artırdığını, ancak uygun araç setlerinin seçimi konusunda eksiklikler olduğunu ortaya koymuştur.

DevOps'un etkin bir şekilde benimsenebilmesi için organizasyonların eğitim ve öğretim süreçlerine yatırım yapması gerekmektedir. Leite ve diğerleri (2020), DevOps eğitiminde karşılaşılan materyal eksikliği ve uygulamalı deneyim eksikliği gibi sorunları ele almış ve standardize edilmiş öğretim yöntemleri geliştirilmesi gerektiğini savunmuştur.

Etiyopya yazılım endüstrisi bağlamında yapılan bir başka çalışmada, Abebe ve diğerleri (2022), DevOps'un yerel yazılım geliştirme firmalarındaki uygulanabilirliğini incelemiş ve kültürel faktörlerin yanı sıra teknik eğitimlerin kritik bir rol oynadığını ortaya koymuştur. Araştırma, ekip içi iş birliğini artıran eğitim programlarının yaygınlaştırılmasını önermektedir.

DevOps'un benimsenmesi sırasında liderlik, otomasyon ve iş birliği gibi unsurların etkisi büyüktür. Muyet ve Sahibuddin (2022), mevcut DevOps benimseme modellerini inceleyerek, bu süreçlerin başarılı olabilmesi için kritik faktörleri ortaya koymuştur. Özellikle, liderlik desteğinin ve doğru ölçümleme mekanizmalarının önemine dikkat çekmişlerdir.

DevOps'un Nesnelerin İnterneti (IoT) uygulamalarına entegrasyonu, operasyonel verimliliği artırmak için yeni fırsatlar sunmaktadır. Bijwe ve Shankar (2022), DevOps kültürünün IoT uygulamalarındaki adaptasyonunu ele almış ve Endüstriyel DevOps Olgunluk Modeli (IDMM) adlı bir çözüm önerisi geliştirmiştir.

DevOps süreçlerinde yapay zeka (AI) ve model tabanlı yaklaşımların entegrasyonu, daha dinamik ve etkili çözümler sağlamaktadır. Eramo ve diğerleri (2024), AIDOaRt projesi kapsamında, yapay zekanın DevOps süreçlerine entegrasyonunu detaylandırmıştır. Çalışmalarında, AI tabanlı karar verme süreçlerinin, hata yönetimi ve performans iyileştirme açısından katkı sağladığı belirtilmiştir.

Bulut tabanlı uygulama dağıtımlarını modellemek için yeni yaklaşımlar geliştirilmiştir. Chiari ve diğerleri (2024), DOML (Deployment-Oriented Modeling Language) adlı bir modelleme yöntemini tanıtmış ve bu yöntemin daha karmaşık sistemlerde uygulanabilirliğini incelemiştir. Araştırmaları, DOML'nin dağıtımları daha standart hale getirme potansiyelini göstermektedir.

Yazılım endüstrisinde DevOps uygulamalarını benimsemeye yönelik hazırlık modelleri, organizasyonların olgunluk seviyelerini değerlendirme açısından önemlidir. Rafi ve diğerleri (2021), RMDevOps modelini geliştirerek, güvenlik gereksinimleri gibi unsurların daha iyi yönetilmesini sağlamışlardır.

DevOps, yazılım geliştirme ve operasyon ekiplerini bir araya getiren bir kültür olarak, eğitim süreçlerinde de önemli zorluklar içermektedir. Fernandes ve diğerleri (2020), DevOps eğitimindeki en büyük zorlukları tespit etmiş ve bu zorlukları tematik analiz yöntemiyle kategorize etmiştir. Çalışmada, eğitim materyallerinin yetersizliği ve pratik uygulama fırsatlarının sınırlı olması gibi problemler öne çıkmıştır. DevOps eğitimi veren akademisyen ve profesyonellerin deneyimleri, bu süreçte karşılaşılan zorlukları anlamada kritik bir öneme sahiptir. Fernandes ve diğerleri (2022), eğitimcilerle yapılan kapsamlı mülakatlar sonucunda, DevOps eğitimiyle ilgili sekiz temel sorunu ortaya koymuştur. Bu sorunlar arasında, eğitim içeriklerinin karmaşıklığı ve teknolojik araçların yeterince iyi anlaşılmaması dikkat çekmektedir.

DevOps uygulamalarının eğitim yönetim sistemlerine entegrasyonu, öğrenci performansını artırmada önemli bir rol oynar. Yang ve diğerleri (2020), Doğu Çin Normal Üniversitesi'nde yapılan bir çalışmada, yeni DevOps tabanlı sistemin eski sisteme kıyasla performansı nasıl artırdığını detaylı bir şekilde ele almıştır. Çalışma, DevOps bileşenlerinin uygulanabilirliğini göstermekte ve eğitim yönetiminde DevOps'un değerini vurgulamaktadır.

DevOps eğitimine yönelik öneriler, standart öğretim materyallerinin geliştirilmesinden, eğitimcilere yönelik rehberlerin hazırlanmasına kadar geniş bir yelpazeyi kapsamaktadır. Fernandes ve diğerleri (2020), eğitim materyallerinin yapılandırılmasının eğitim süreçlerini büyük ölçüde iyileştirebileceğini belirtmiştir. Yang ve diğerleri (2020) ise DevOps konseptlerinin daha geniş çapta benimsenmesi için yeni teknolojilerin eğitime entegre edilmesi gerektiğini önermektedir.

DevOps, yazılım geliştirme ve operasyon ekiplerini birleştirerek kalite güvencesini süreçlerin temel bir parçası haline getirir. Ibrahim ve diğerleri (2019), DevOps'un kalite güvencesi ile ilgili karşılaşılan zorluklarını ele almış ve süreçlerin optimize edilmesi için analitik

yaklaşımların önemini vurgulamıştır. Çalışma, yazılım geliştirme ve operasyon ekipleri arasında sürekli iletişim ve iş birliğinin kalite güvencesini nasıl artırabileceğini detaylandırmıştır.

DevOps süreçlerinde veri kalitesini değerlendirmek ve optimize etmek için çok kriterli karar verme mekanizmaları kullanılmaktadır. Rafi ve diğerleri (2020), DevOps ortamında veri kalitesini artırmaya yönelik çok kriterli bir çerçeve sunmuş ve gerçek zamanlı veri analizi ile süreçlerin iyileştirilebileceğini göstermiştir.

DevOps, yazılım geliştirme süreçlerinin hızını artırırken kalite standartlarını korumayı hedefler. Mishra ve Otaiwi (2020), sistematik bir literatür taramasıyla DevOps'un yazılım kalitesine etkilerini incelemiş ve bu uygulamanın kalite farkındalığını artırdığı sonucuna ulaşmıştır. Çalışma, DevOps'un özellikle hata oranlarını düşürme ve teslimat sürelerini optimize etme konusundaki rolünü vurgulamaktadır.

Bulut tabanlı sistemlerde DevOps uygulamalarının kalitenin yanı sıra güvenilirliği de artırdığı belirtilmiştir. Tatineni (2023), DevOps'un bulut sistemlerde uygulanması sırasında sağlanan kalite ve güvenilirlik artışını analiz etmiştir. Çalışma, bulut ortamında yazılım teslimat süreçlerinin daha esnek hale getirildiğini ve müşteri memnuniyetinin artırıldığını ortaya koymuştur.

DevOps'un kalite farkındalığını artıran özellikleri, yazılım geliştirme süreçlerinde kalite beklentilerinin korunmasında kritik bir rol oynamaktadır. Alnafessah ve diğerleri (2021), 2015-2020 yılları arasında yapılan çalışmaları incelemiş ve DevOps'un hızlı teslimat sağlarken kaliteyi koruyabileceğini belirtmiştir. Çalışmada, gelecekte DevOps analitiği ve otomasyon araçlarının daha fazla önem kazanacağı vurgulanmıştır.

DevOps güvenliği, yazılım geliştirme ve operasyon süreçlerinin entegre edilmesi sırasında ortaya çıkan güvenlik zorluklarını ele alır. Rafi ve diğerleri (2020), DevOps güvenlik zorluklarını önceliklendiren bir taksonomi sunmuş ve güvenlik uygulamalarının süreçlere entegre edilmesinin önemini vurgulamıştır. Çalışma, güvenlik analitiklerinin ve otomasyon araçlarının etkin bir şekilde kullanılması gerektiğine dikkat çekmiştir.

DevOps yaşam döngüsünde güvenliği artırmak için yenilikçi yaklaşımlar geliştirilmiştir. Okubo ve Kaiya (2022), süreç madenciliği ve anomali tespiti yöntemlerini kullanarak operasyon

sırasında güvenliği artırmaya yönelik bir çerçeve önermiştir. Bu yöntem, operasyon verilerindeki anomalilerin tespit edilmesi ve hızlı müdahaleyi sağlamaktadır.

DevSecOps, güvenlik prensiplerini DevOps süreçleriyle birleştirerek güvenli yazılım teslimatını hedefler. Zhao ve diğerleri (2024), DevSecOps'un temel boyutlarını analiz etmiş ve bu sürecin küresel yazılım geliştirme projelerindeki mevcut durumunu değerlendirmiştir. Çalışma, CPTM (Comprehensive Process-Tailored Model) modelini önermiştir.

DevOps süreçlerinde dayanıklılık ve güvenlik iyileştirmesi için yeni yaklaşımlar geliştirilmiştir. Sriraman ve Shriram (2024), Slide-Block çerçevesi adını verdikleri, uçtan uca güvenlik sağlayan bir yöntem tanıtmışlardır. Bu çerçeve, mevcut modellerle karşılaştırıldığında daha yüksek performans ve güvenlik sunmaktadır.

DevSecOps süreçlerinin yazılım geliştirme projelerine etkili bir şekilde uygulanması, organizasyonların güvenli yazılım geliştirme olgunluğunu artırmaktadır. Akbar ve diğerleri (2022), DevSecOps süreçlerini uygularken karşılaşılan zorlukları ele almış ve güvenlik politikalarının entegre edilmesi için öneriler sunmuştur.

Bu literatür taraması, DevOps'un yazılım geliştirme ve operasyon süreçlerindeki dönüştürücü etkisini kapsamlı bir şekilde ele almıştır. Özellikle, DevOps'un kültürel, teknik ve eğitsel boyutlarında karşılaşılan zorlukların organizasyonel başarının önündeki en büyük engeller olduğu görülmektedir. Çalışmalar, DevOps'un benimsenmesi sürecinde liderlik desteği, işbirliği stratejileri ve eğitim programlarının güçlendirilmesi gerektiğini açıkça ortaya koymaktadır. Çalışmaların çoğu, sistematik literatür taraması (SLR) veya sistematik haritalama çalışması (SMS) metodolojisi kullanılarak gerçekleştirilmiştir.

Gelecekteki araştırmalar, bu alanlarda daha derinlemesine çözümler sunabilir ve DevOps'un yazılım endüstrisindeki etkisini artırabilir.

4. YÖNTEM

Bu araştırmanın amacı, savunma sanayiinde yer alan ve yüksek teknoloji geliştiren bir kurum için DevOps Proje Yönetim Modelinin uyarlanmasına yönelik modeller geliştirmek ve bunların uygulamalarına yönelik önerileri ortaya koymaktır. Çalışma nitel araştırma yaklaşımı doğrultusunda Eylem Araştırması ve doküman incelemesi yöntemleri kullanılarak iki aşamada yürütülmüştür (Şekil 4.1). Birinci aşamada literatür taraması ve doküman incelemesine ek olarak ilgili kurumda DevOps proje yönetim modelini kullanan yazılım geliştirme uzmanlarıyla yarı yapılandırılmış mülakatlar gerçekleştirilmiş, veri toplanmış ve analiz edilmiştir. Mülakatlarda, DevOps proje yönetiminde ihtiyaç yönetimi, takım yönetimi, risk yönetimi ve kalite yönetimi ile yapay zekâ kullanımı konularına ek olarak karşılaşılan güçlük, kullanılan yöntem, teknik ve araçlara ilişkin katılımcıların görüşleri alınmıştır. İkinci aşamada, birinci aşamada elde edilen bulgular doğrultusunda UML diyagramları kullanılarak uyarlama modelleri ve ilgili bileşenleri geliştirilmiş, çeşitli öneriler sunulmuştur.

Literatür taramasında, elektronik kaynaklarda otomatik arama teknikleri kullanılarak dijital veri tabanları ana arama stratejisi olarak belirlenmiştir. Aşağıdaki veritabanları kullanılmıştır.

- Science Direct
- Google Scholar
- IEEE Xplore
- ResearchGate

Bu veritabanları, bilgisayar bilimleri, yazılım mühendisliği ve organizasyonel yönetim alanındaki en güncel ve güvenilir akademik çalışmalara erişim sağlamaktadır.

4.1. Araştırma Modeli

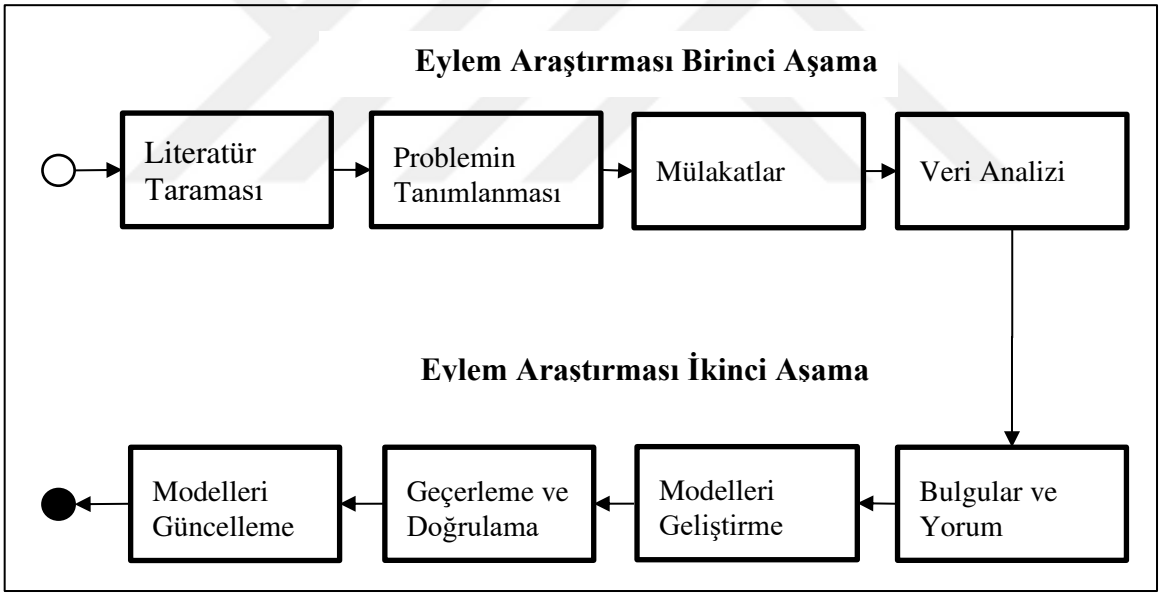
Bu çalışmada benimsenen bütünleşik ve karma araştırma yaklaşımı, araştırma bulgularının nicel ve nitel verilerle desteklenmesi ile araştırmanın kapsam ve derinliğinin artırılmasını sağlamış, daha güçlü bilimsel kanıtların elde edilmesine olanak tanımıştır. Ayrıca bu yaklaşım, araştırma sürecinin hem araştırmacı hem de katılımcıların iş birliği

içinde, sürekli gelişim anlayışıyla gerçekleştirilmesini desteklemiştir. Araştırma soruları ve modeli aşağıda gösterilmiştir:

Araştırma soruları aşağıdaki gibidir:

- **AS-1:** DevOps Proje Yönetim Modelinin uygulanmasında kullanılan yöntem, teknik ve araçlar ile karşılaşılan güçlükler nelerdir?
- **AS-2:** Savunma sanayindeki ilgili kuruma yönelik süreç, ihtiyaç, takım, risk ve kalite yönetimi ile yapay zekâ kullanımını temel alan DevOps uyarlama modelleri nasıl geliştirilebilir?

Birinci araştırma sorusu, literatür taraması ve yarı yapılandırılmış mülakatlar sonucunda elde edilen veriyle; ikinci araştırma sorusu ise UML modelleme dili kullanılarak geliştirilen uyarlama modelleri ve ilgili bileşenleriyle cevaplanmaya çalışılmıştır.



Şekil 4.1. Araştırma Modeli

4.2. Araştırmanın Çıktıları

DevOps Proje Yönetim Modeli'nin uyarlanmasına yönelik uygulama önerilerine ek olarak geliştirilen uyarlama modelleri aşağıdaki bileşenlerden oluşmuştur.

- DevOps Süreç Yönetimi Modeli,

- Gereksinim Yönetimi Modeli,
- Takım Yönetimi Modeli,
- Risk Yönetimi Modeli,
- Kalite Yönetimi Modeli,
- Yapay Zekâ Yönetimi Modelidir.

4.3. Evren ve Örneklem

Araştırmanın evrenini ilgili kurumun DevOps süreçlerinde aktif rol alan ve bilgiye doğrudan erişimi bulunan çalışanlar oluşturmuştur. Örneklem seçiminde amaçlı örneklem yöntemi kullanılarak DevOps süreçleriyle doğrudan ilişkili ve projelerde yer almış kişilerden 17 katılımcı belirlenmiştir. Böylece DevOps süreçlerine ilişkin deneyim ve bilgi düzeylerinin göz önünde bulundurulması ile katılımcı çeşitliliğinin sağlanması hedeflenmiştir.

4.4. Mülakat

Yarı yapılandırılmış mülakat protokolü, beş adımdan oluşan bir süreci izlemiştir: Bunlar; (a) hazırlık aşaması, (b) mülakatın gerçekleştirilmesi, (c) kayıt ve dokümantasyon, (d) kayıtların yazıya dökülmesi ve (e) veri analizi ve raporlamadır. Mülakat soruları, araştırma amacıyla uyumlu olacak şekilde tasarlanmıştır. DevOps süreçleriyle birlikte risk yönetimi, kalite yönetimi, yapay zekâ yönetimi vb. araştırma konuları önceden belirlenmiş temaları ve kod kitaplarını oluşturmuştur. Mülakata katılacak kişilerin seçimi için amaçlı örnekleme yöntemi kullanılmış, proje yönetimi deneyimine sahip katılımcılarından belirlenmiş ve 17 katılımcı mülakata katılmayı kabul etmiştir. Önceden belirlenen temalara ek olarak DevOps süreçlerine yönelik uygulamalar, kullanılan yöntem ve teknikler, araçlar ile kurumsal standartların, bağlamsal faktörler araştırılmıştır. Açık ve dürüst yanıtları teşvik etmek için rahat bir mülakat ortamı sağlanmış, mülakatlar kaydedilmiş ve nitel veri analizi için kayıtlar metne dökülmüştür.

4.5. Veri Toplama

Veri toplama süreci, iki aşamaya göre aşağıdaki şekilde yapılandırılmıştır:

- Birinci aşama: katılımcılarla yarı yapılandırılmış mülakatlar gerçekleştirilmiştir. Mülakatlarda, önceden belirlenen temalar ile katılımcıların DevOps süreçlerinde karşılaştıkları zorluklar, kullandıkları yöntem, teknik ve araçlar ile ihtiyaçları ele alınmıştır. Bu süreçte EK-1 sunulan mülakat formu rehber olarak kullanılmıştır.
- İkinci aşama: İlk aşama sonucunda geliştirilen DevOps bileşenlerine yönelik uyarlama modelleri ve uygulama önerileri uzman katılımcılara tekrar sunulmuş ve geri bildirimler alınmıştır. Ayrıca bu aşamada geliştirilen uyarlama modelleri ile ilgili bileşenlerin işleyişi, doğruluğu ve bütünlüğü, senaryo analizleri doğrultusunda değerlendirilmiş ve gerekli güncellemeler yapılmıştır.

4.6. Veri Analizi

Mülakatlardan elde edilen veri ve içeriği tümden gelim (dedüktif) ve tüme varım (endüktif) veri analiz yöntemleri kullanılarak incelenmiştir (Guest vd., 2012). İlk aşamada, araştırma sorularına ve kuramsal çerçeveye dayalı olarak dedüktif kodlama yapılarak kod kitapları oluşturulmuştur. Daha sonra endüktif veri analizi ile ortaya çıkan yeni temalar ve desenler keşfedilmeye çalışılmıştır. Transkriptor yazılımı (1.2.0 (860)) aracılığıyla toplanan veri yazılı metinlere dönüştürülmüş ve metinler üzerinde yazım ve çeviri hataları düzeltilerek içerik doğruluğu artırılmıştır. Mülakatların transkriptleri, yazılı bir formatta düzenlendikten sonra QDA Miner Lite (v3.0.6) yazılımına aktarılmış ve analiz süreci başlatılmıştır. İlk olarak, her bir transkript üzerinde açık kodlama işlemi gerçekleştirilmiştir; bu süreçte, mülakatlardan elde edilen ifadeler anlamlı alt birimlere ayrılmış ve belirli kodlarla etiketlenmiştir. Daha sonra, yazılımın "kod sıklığı analizi" özelliği kullanılarak, belirli kodların hangi sıklıkta tekrar ettiği analiz edilmiştir. Kodlar arasındaki ilişkileri daha iyi anlamak amacıyla, "kod ağacı" ve "kategori gruplama" araçları kullanılmış ve bu ilişkilerden hareketle temalar oluşturulmuştur. Ayrıca, yazılımın "kod arama ve sorgulama" fonksiyonu kullanılarak, belirli temalarla ilişkilendirilebilecek önemli ifadeler hızlı bir şekilde tespit edilmiştir. Bu süreç, tematik doygunluğun sağlanmasına ve bulguların geçerliliğinin artırılmasına katkı sağlamıştır. QDA Miner Lite'ın arayüzü ve sunduğu görsel analiz araçları, verilerin organize edilmesi ve raporlanmasında araştırma sürecini önemli ölçüde kolaylaştırmıştır.

Tematik analiz adımları, dedüktif-endüktif yaklaşımlarının entegrasyonu ile aşağıdaki şekilde uygulanmıştır:

- Kod Kataloglarını Oluşturma: Araştırma soruları ve kuramsal çerçeve temel alınarak dedüktif kodlama yapılması ve kod kitaplarının oluşturulması,
- Veriyi İnceleme: Veri kuramsal çerçeveye dayalı olarak kod kitapları ve dedüktif yaklaşımla incelenmiş, araştırma soruları doğrultusunda başlangıç kategorileri oluşturulmuştur.
- Başlangıç Kodlarının Oluşturulması: Görüşmecilerin her soru için verdiği cevaplar, dedüktif bir yaklaşımla incelenmiş, hem anlamsal ("semantic") hem de örtük ("latent") içerik analizi gerçekleştirilmiştir.
- Kodların Gruplandırılması ve Temaların Araştırılması: İlişkili kodlar bir araya getirilmiş ve alt temalar altında gruplanmıştır. Alt temalardan ana temalar türetilerek analiz süreci ilerletilmiştir.
- Temaların Gözden Geçirilmesi: Alt ve ana temalar yeniden gözden geçirilerek dedüktif bir çerçevede araştırma sorularına uygunluğu incelenmiştir.
- Raporun Oluşturulması: QDA Miner Lite yazılımı kullanılarak nihai temalar ve alt temalar düzenlenmiş ve raporlama sürecine aktarılmıştır.

Analiz sürecinde, dedüktif yaklaşım teorik çerçeveyi oluştururken, endüktif yaklaşım veriden yeni bulguların ortaya çıkarılmasını sağlamıştır. Bu kombinasyon, DevOps proje yönetim süreçlerinde karşılaşılan zorluklar, kullanılan yöntemler ve ihtiyaçlar konusunda detaylı ve zengin bir veri kümesi elde edilmesine olanak tanımıştır. Elde edilen bulgular ve dedüktif yöntemle oluşturulan kod kitaplarıyla yapılan karşılaştırmalar temel alınarak uyarlama modellerine yönelik UML diyagramları geliştirilmiştir. Oluşturulan uyarlama modellerin işleyişi, doğruluğu ve bütünlüğü senaryo analizleriyle, sözdizimi ("syntax") doğruluğu ise UML doğrulama teknikleriyle değerlendirilmiştir. Bu bütünleşik ve tekrarlı süreç, kuramsal alt yapı ve proje uygulamalarının birlikte ele alınmasını sağlayarak uyarlama modelin kapsamlı bir şekilde geçerleme ve doğrulanmasına olanak tanımıştır.

Geliştirilen DevOps uyarlama modelleri, literatür taranarak oluşturulan ana temalara (kod katalogları), kullanıcıların işlevsel ve işlevsel olmayan ihtiyaçlarına ve UML kurallarına uygunluk bakımında senaryo analizi yöntemi kullanılarak değerlendirilmiştir.

4.7. Geçerlilik ve Güvenirlik

Mülakat, gözlem ve senaryo analizi gibi tekniklerin birlikte kullanımı veri bütünlüğü ve çeşitliliğini artırmıştır. Yarı yapılandırılmış mülakatlarda uygun ortam ile katılımcıların araştırma sorularını rahat biçimde cevaplamaları sağlanarak araştırmanın güvenirliğin artırılması hedeflenmiştir. Dış güvenirliği artırmak için araştırma süreci ayrıntılı olarak planlanmış, veri toplama süreci detaylandırılarak kontrol edilmiş, kayıtlar ayrıntılı ve araştırmacılara açık biçimde Google Drive dizininde paylaşılmıştır. Güvenirliği artırmak amacıyla içerik ve veri analizinde kodlar kullanılmış ve temalar oluşturulmuştur. Tümden gelim ve tüme varım nitel veri analiz yöntemlerinin birlikte kullanımı, uzman görüşlerinin alınması ve araştırmacının sürece aktif katılımı araştırmanın geçerliliği ve güvenirliğini artıran önemli unsurları teşkil etmiştir.

Katılımcılar, ihtiyaç yönetimi, takım dinamikleri, risk yönetimi, kalite yönetimi ve yapay zeka uygulamaları gibi DevOps'un temel bileşenleri üzerinde farklı roller üstlenmiş, çeşitli profesyonel geçmişlere ve uzmanlık alanlarına sahiptir. Katılımcıların belirlenmesinde, maksimum çeşitlilik örnekleme yöntemi benimsenmiş, böylece farklı organizasyonel yapılarda DevOps süreçlerinin uygulanmasına dair geniş bir perspektif elde edilmiştir. Araştırmanın örneklem büyüklüğü, nitel araştırmalarda anlamlı derinlik ve bilgi zenginliği sağlamaya uygun görülmüştür (Patton, 2015). Bu bağlamda, katılımcı sayısı ve profili, araştırmanın güvenirliğini ve bulguların savunma sanayindeki yazılım geliştirme süreçlerine genellenebilirliğini destekleyecek şekilde, temsil edici bir düzeyde tutulmuştur. Bunun yanı sıra, mülakat verilerinin analizi sırasında tematik doygunluk sağlanarak bulguların geçerliliği artırılmıştır.

4.8. Sınırlılıklar

Bu araştırmanın bulguları, elde edilen sonuçlar ve yorumlar araştırmanın gerçekleştirildiği zaman, mekân ve ilgili kurum ile sınırlıdır.

5. BULGULAR

Görüşmeye katılan toplam 17 kişiye ilişkin demografik bilgiler aşağıdaki tablolarda gösterilmiştir. Görüşmecilerin büyük bir bölümü doğrudan Savunma Sanayi ile ilişkili proje yönetimlerinde yer alan kişilerden oluşmuştur.

Tablo 5.1, görüşmecilerin cinsiyet dağılımı verilmiştir. Erkek katılımcıların oranı %64,7 iken, kadın katılımcıların oranı %35,3'tür. Bu tablo, çalışmanın cinsiyet dengesine ilişkin bir özet sunmaktadır.

Tablo 5.1. Görüşmecilerin Cinsiyetlerine Göre Dağılımı

Cinsiyet	Kişi Sayısı	Yüzdesi
Erkek	11	64.7
Kadın	6	35.3
Toplam	17	100

Tablo 5.2, görüşmecilerin yaş aralıklarına göre dağılımı gösterilmektedir. Katılımcıların büyük bir kısmı 25-29 yaş aralığında olup, bu grup toplamın %41,2'sini oluşturmaktadır.

Tablo 5.2. Görüşmecilerin Yaş Aralığına Göre Dağılımı

Yaş Grupları	Kişi Sayısı	Yüzdesi
25-29	7	41.2
30-34	4	23.5
35-39	3	17.6
47-50	3	17.6
Toplam	17	100.0

Tablo 5.3, görüşmecilerin iş tecrübesi aralıklarına göre dağılımı verilmiştir. Katılımcıların %29,4'ü 3-5 yıl ve %29,4'ü 6-8 yıl iş tecrübesine sahiptir, bu da araştırmada çeşitli deneyim seviyelerinin temsil edildiğini göstermektedir.

Tablo 5.3. Görüşmecilerin İş Tecrübesi Aralığına Göre Dağılımı

İş Tecrübesi Aralığı (yıl)	Kişi Sayısı	Yüzdesi
3-5	5	29.4
6-8	5	29.4
10-12	3	17.6
13-15	1	5.9
22-24	1	5.9
25-27	1	5.9
31-33	1	5.9
Toplam	17	100.0

Tablo 5.4, görüşmecilerin yazılım iş tecrübesi aralıklarına göre dağılımı sunulmaktadır. 6-8 yıl yazılım iş tecrübesine sahip katılımcılar %35,3 ile en büyük grubu oluştururken, diğer deneyim aralıkları daha az temsil edilmiştir.

Tablo 5.4. Görüşmecilerin Yazılım İş Tecrübesi Aralığına Göre Dağılımı

Yazılım İş Tecrübesi Aralığı (yıl)	Kişi Sayısı	Yüzdesi
3-5	5	29.4
6-8	6	35.3
10-12	2	11.8
13-15	1	5.9
22-24	1	5.9
25-27	1	5.9
31-33	1	5.9
Toplam	17	100.0

Tablo 5.5, görüşmecilerin DevOps tecrübesine ilişkin dağılımı verilmiştir. Katılımcıların %41,2'sinin 5 yıl, %35,3'ünün 3-4 yıl, %23,5'inin ise 2-3 yıl DevOps deneyimi bulunmaktadır. Bu dağılım, çalışmanın DevOps süreçlerinde farklı deneyim seviyelerine sahip bireylerden veri topladığını göstermektedir.

Tablo 5.5. Görüşmecilerin DevOps Tecrübesi Aralığına Göre Dağılımı

DevOps Tecrübesi Aralığı (yıl)	Kişi Sayısı	Yüzdesi
5	7	41.2
3-4	6	35.3
2-3	4	23.5
Toplam	17	100.0

5.1. Tümdengelim Tematik Analiz Bulguları

Bu çalışmada mülakatlarda sorulan araştırma sorularına ve teorik çerçeveye dayalı olarak aşağıdaki kapsamda dedüktif kodlama yapılmış ve soru bazında tablolar oluşturulmuştur.

- DevOps Süreç Yönetimi,
- Gereksinim Yönetimi,
- Kalite Yönetimi,
- Risk Yönetimi,
- Takım Yönetimi,
- Yapay Zeka Yönetimi

Dedüktif analiz yöntemleri kullanılarak gerçekleştirilen bu süreç, hem teorik çerçeveyle uyumlu bulguların doğrulanmasını hem de yeni içgörülerin ortaya çıkarılmasını sağlamıştır.

5.1.1. DevOps süreç yönetimi kod kataloğu

Tablo 5.6, DevOps süreç yönetiminde kullanılan temel bileşenleri ve açıklamalarını sunmaktadır. DevOps, yazılım geliştirme ve operasyon süreçlerinin entegrasyonunu sağlayarak verimlilik ve hız açısından önemli avantajlar sunmaktadır.

Tablo 5.6. Soru 2'e Ait Kodlama

Tema	**Kodlar**
Sürekli Entegrasyon (CI)	Kod testi, entegrasyon testi, CI araçları (Jenkins, TravisCI), yapı otomasyonu, hata tespiti.
Sürekli Teslimat (CD)	CD araçları, dağıtım otomasyonu, sürekli güncelleme, teslimat pipeline, CI/CD entegrasyonu.
Altyapı Otomasyonu	Docker, Kubernetes, Terraform, Ansible, otomasyon betikleri, altyapı yönetimi araçları.
İzleme ve Kayıt Tutma	İzleme araçları (Prometheus, Grafana), log analizi, uyarı sistemleri, hata izleme araçları.
DevOps Araç Zinciri	Jenkins, Git, GitHub Actions, Docker, Kubernetes, Terraform, CI/CD araçları.
Süreç İyileştirme Döngüsü	Süreç ölçümü, geri bildirim döngüsü, retrospektif toplantılar, sürekli iyileştirme.
Kültürel Dönüşüm	Ekipler arası iş birliği, iletişim araçları (Slack, Teams), kültürel değişim yönetimi.
DevOps Standartları ve Uyum	ISO standartları, endüstri standartları, güvenlik politikaları, standartlara uyum araçları.

5.1.2. Gereksinim yönetimi kod kataloğu

Proje yönetimi çerçevesinde gereksinimlerin doğru bir şekilde belirlenmesi, analiz edilmesi, modellenmesi ve doğrulanması, proje hedeflerinin başarıyla gerçekleştirilmesi açısından kritik öneme sahiptir. Tablo 5.7, ihtiyaç yönetiminin farklı aşamalarını ayrıntılı olarak ele almakta ve bu süreçlerin her birinin hangi faaliyetleri içerdiğini özetlemektedir (Uysal, 2022b).

Bu süreçlerin her biri, projenin gereksinim yönetimi bileşeninin bir parçasını oluşturarak projenin başarıya ulaşmasını desteklemektedir. Tablo 5.7'de sunulan temalar, bu

süreçlerin daha sistematik ve etkili bir şekilde yönetilmesine katkı sağlamayı amaçlamaktadır.

Tablo 5.7. Soru 4'e Ait Kodlama

Tema	Kodlar
Gereksinim Ortaya Çıkarma	Gereksinim, ihtiyaç, gözlem, doküman incelemesi, anket, mülakat, paydaş, sistem, yazılım, donanım, alt yapı, takım, organizasyon, atölye çalışması, fonksiyona gereksinim, fonksiyonel olmayan gereksinim, siber güvenlik gereksinimi, kalite gereksinimi, yasal ve mevzuat gereksinimi, standartlara uyum
Gereksinim Analizi	Nicel analiz, nitel analiz, alan (domain) analizi, fonksiyonel analiz, iş analizi, görev analizi, süreç analizi, kullanım durumu (use case), kullanıcı hikayesi (user story), ürün listesi (backlog), odak grup görüşmesi, senaryo analizi, amaç odaklı analiz, problem analizi, problem alanı, kabul ölçütü, olurluk analizi (feasibility), gereksinim sınıflaması, bağımlılık analizi
Gereksinim Modelleme	Çözüm alanı, sistem modelleme, yazılım modelleme, formal yöntemler, nesneye yönelimli modelleme, yapısal modelleme, UML, diyagramlar, veri modelleme, varlık-bağıntı modelleme, veri akışı modelleme, simülasyon, meta model,
Gereksinim Müzakeresi	Problem tanımı, bakış açısı analizi, paydaş yönetimi, müzakere yönetimi, amaçlar, hedefler, çıkar çatışması, çatışma yönetimi, iletişim, iş birliği
Gereksinim Önceliklendirme	Paydaş incelemesi, nicel yöntemler, nitel yöntemler, önceliklendirme, sıralama, zaman, maliyet, iş gücü, organizasyon,
Gereksinim Belirtimi	Gereksinim tanımı, gereksinim dokümanı,
Gereksinimleri Gözden Geçirme	Akran incelemesi, paydaş incelemesi, bakış açısı incelemesi
Gereksinim Geçerleme	Akran doğrulama, paydaş doğrulaması, gereksinim dokümanı
Gereksinim Doğrulama	Gözden geçirme, senaryo analizi, formal yöntemler, izlenebilirlik analizi, test, simülasyon, kontrol listesi, tutarlılık (consistency) kontrolu, bütünlük (completeness) kontrolu,
Gereksinim Yönetimi	Gereksinim ömür devri, izleme, güncelleme, bakım

5.1.3. Kalite yönetimi kod kataloğu

Tablo 5.8, proje yönetimi süreçlerinde kalite yönetimine yönelik kodlamaları içermektedir (ISO/IEC 250nn (SQuaRE)). Kalite yönetimi, projelerin başarıyla tamamlanması için kritik bir bileşen olup süreçlerin ve çıktının belirlenen standartlara uygunluğunu garanti altına almayı hedeflemektedir. Tabloda, kalite yönetiminin temel unsurları ve bunlara ilişkin açıklamalar detaylandırılmıştır. Tablodaki bu temalar, kalite yönetimi sürecinde dikkate alınması gereken temel unsurları sistematik bir şekilde sunmaktadır. Kalite yönetimi süreçlerinin bu unsurlara dayalı olarak ele alınması, projenin genel performansını artıracak ve paydaşların memnuniyetini sağlayacaktır.

Tablo 5.8. Soru 6'ya Ait Kodlama

Tema	Kodlar
Kalite Modeli Sınıflaması	Sistem kalite modeli, yazılım kalite modeli, veri kalite modeli, kalite karakteristik özelliği
Sistem ve Yazılım Kalite Modeli	Fonksiyonel uygunluk, performans verimliliği, uyumluluk (bağdaşma), etkileşim yeteneği, güvenilirlik (reliability), güvenlik (security), bakım yapılabilirlik, esneklik, emniyet (safety)
Kullanım Kalitesi	Etkililik, verimlilik, kullanıcı tatmini, risk içermemek, bağlam kapsamı
Veri Kalitesi	Doğruluk, tamlık, tutarlılık, kredibilite, güncellik, ulaşılabilirlik, uyumluluk, gizlilik, verimlilik, doğruluk, izlenebilirlik, anlaşılabilirlik, taşınabilirlik, kurtarılabilirlik
Kalite Gereksinimi	İç kalite gereksinimi, dış kalite gereksinimi, kullanım kalitesi gereksinimi, veri kalitesi gereksinimi
Kalite Ölçümü	Kalite ölçümü rehber ve modeli, sistem kalite ölçümü, yazılım kalite ölçümü, veri kalite ölçümü, kalite ölçütü, kalite metriği, beyaz kutu yaklaşımı, kara kutu yaklaşımı
Kalite Değerlendirme	Kalite değerlendirme rehberi, kalite değerlendirme referans modeli, kalite değerlendirme planı, kalite değerlendirme süreci
Kalite Yönetimi	Kalite planlaması, kalite yönetim süreci

5.1.4. Risk yönetimi kod kataloğu

Tablo 5.9, proje yönetiminde risk yönetimi sürecine ilişkin temel temaları ve açıklamalarını içermektedir (Chaouch ve diğerleri, 2019). Risk yönetimi, proje başarısını etkileyebilecek potansiyel olumsuz durumları ve fırsatları önceden belirleyerek bunlara karşı uygun stratejiler geliştirmeyi amaçlayan kritik bir süreçtir (Tavares ve diğerleri, 2019). Tablo 5.9'da, risk yönetiminin aşamaları detaylı olarak ele alınmıştır. Bu temalar, risk yönetimi süreçlerinin yapılandırılmış bir şekilde ele alınmasını sağlayarak proje risklerinin etkin bir şekilde yönetilmesine katkıda bulunmaktadır. Risklerin belirlenmesinden kontrol edilmesine kadar olan bu süreçler, projelerin başarıya ulaşmasını ve beklenmeyen durumlara karşı hazırlıklı olunmasını desteklemektedir.

Tablo 5.9. Soru 5'e Ait Kodlama

Tema	Kodlar
Risklerin Belirlenmesi ve Tanımlanması	Risk belirleme, risk tanımlama, risk kayıtları, potansiyel riskler, sistematik analiz.
Risklerin Analizi ve Değerlendirilmesi	Niteliksel analiz, niceliksel analiz, risk olasılığı, risk etkisi, değerlendirme kriterleri.
Risklerin Önceliklendirilmesi	Risk sıralama, kritik riskler, önceliklendirme, olasılık-şiddet matrisleri, risk değerlendirme.
Risklerin Planlanması ve Önlemler	Risk azaltma, risk stratejileri, eylem planları, fırsat artırma, önleyici önlemler.
Risklerin İzlenmesi ve Kontrolü	Risk izleme, risk kontrolü, plan güncelleme, sürekli izleme, uyarlamalı yönetim.

5.1.5. Takım yönetimi kod kataloğu

Tablo 5.10, proje yönetiminde takım yönetimine ilişkin temel temaları ve açıklamalarını içermektedir. Proje ekibinin etkin bir şekilde yönetilmesi, proje hedeflerinin başarıyla gerçekleştirilmesi için kritik bir öneme sahiptir. Tablo 5.10'da, takım yönetimi süreçlerine yönelik başlıca konular ve bu konuların detaylı açıklamaları yer almaktadır. Tablodaki bu temalar, takım yönetimi süreçlerinin yapılandırılmasına katkı sağlayarak ekiplerin verimliliğini ve uyumunu artırmayı hedeflemektedir. Proje ekibinin doğru şekilde yönetilmesi, sadece projenin başarısını değil, aynı zamanda ekip içi memnuniyeti ve sürdürülebilirliği de olumlu yönde etkilemektedir.

Tablo 5.10. Soru 3'e Ait Kodlama

Tema	**Kodlar**
Proje Takımı Planlaması	Kaynak ihtiyaçları, yetkinlik analizi, görev planlaması, iş yükü tahsisi.
Takımın Oluşturulması	Yetenek seçimi, ekip dahil etme, uyumlu çalışma ortamı, iş gücü planlaması.
Takım Rolleri ve Sorumluluklar	Rol tanımları, sorumluluk ataması, iş paylaşımı, görev netliği.
Takım Gelişimi	Sürekli öğrenme, geliştirme programları, ekip sinerjisi, yetkinlik artırma.
Takım Yönetimi ve Liderlik	Etkin iletişim, ekip motivasyonu, liderlik, yönlendirme, takım hedefleri.
Performans ve Gelişim	Performans izleme, değerlendirme, geri bildirim, gelişim planları.
Teknoloji ve Araçların Kullanımı	Verimlilik araçları, teknoloji entegrasyonu, süreç otomasyonu, iş süreçlerini kolaylaştırma.
Sanal ve Hibrit Takımların Yönetimi	Uzaktan çalışma, hibrit ekipler, iletişim yöntemleri, iş birliği araçları.

5.1.6. Yapay zeka yönetimi kod kataloğu

Tablo 5.11, proje yönetiminde yapay zeka (AI) kullanımına ilişkin temel temaları ve açıklamaları içermektedir. Yapay zeka, DevOps süreçlerinde otomasyon, karar destek sistemleri ve veri analitiği gibi alanlarda kritik bir rol oynamaktadır. Bu temalar, AI teknolojilerinin proje yönetimine entegrasyonu sırasında dikkate alınması gereken anahtar unsurları ortaya koymaktadır.

Tablo 5.11. Soru 8'e Ait Kodlama

Tema	Kodlar
AI Destekli Otomasyon	Sürekli entegrasyon, sürekli teslimat, otomasyon araçları, iş akışı otomasyonu, yazılım dağıtımı.
Karar Destek Sistemleri	Büyük veri analitiği, öngörüsel modelleme, karar destek araçları, veri analizi.
Veri Analitiği ve Tahmin	Makine öğrenimi algoritmaları, veri modelleme, tahmin sistemleri, istatistiksel analiz, regresyon.
Yapay Zeka Uygulama Alanları	Proje optimizasyonu, kod analizi, risk tahmini, hataların otomatik tespiti, performans değerlendirme.
AI ve Ekip Yönetimi	İş yükü yönetimi, görev dağıtımı, ekip performans analizi, AI destekli planlama araçları.
Etik ve Güvenlik	Etik algoritmalar, veri gizliliği, güvenlik protokolleri, AI açıklanabilirliği (explainability).
AI Eğitim ve Uygulama	Eğitim materyalleri, AI araç eğitimi, teknik destek, uygulama rehberliği.

5.2. Tümevarım Tematik Analiz Bulguları

Bu çalışmada gerçekleştirilen mülakatlar, DevOps uygulamalarının kurumsal süreçlere entegrasyonu sırasında karşılaşılan organizasyonel ve teknik zorlukları detaylı bir şekilde analiz edilmiştir. Mülakatlardan elde edilen veriler, DevOps'un proje yönetim süreçlerine adaptasyonunda karşılaşılan temel sorunları ve bu sorunların çözümüne yönelik kullanılan yöntemleri kapsamlı bir şekilde ortaya koymaktadır. Dedüktif ve endüktif analiz yöntemleri kullanılarak gerçekleştirilen bu süreç, hem teorik çerçeveyle uyumlu bulguların doğrulanmasını hem de yeni içgörülerin ortaya çıkarılmasını sağlamıştır.

5.2.1. Problem sahaları tematik analizi

Soru: "DevOps Proje Yönetim Çerçevesi doğrultusunda karşılaştığınız ana sorunlar ve ana problem sahaları nelerdir? Bu kapsamda çözüme yönelik kullandığınız ve/veya önerdiğiniz yöntem, teknik ve araçlar nelerdir?"

Amaç:

(a) İlgili kurumdaki ana problem sahalarını ve kullanıldıkları çözümleri belirlemek,
(b) Geliştirilecek DevOps uygulama ve entegrasyon modeline temel oluşturmaktır.

Tablo 5.12, DevOps süreçlerinde karşılaşılan zorlukların kod frekanslarını, katılımcı oranlarını ve kelime oranlarını içermektedir. Kodlama analizi, mülakatlarda belirtilen sorunların dağılımını ve bunların önem derecelerini belirlemek için kullanılmaktadır. Her bir kod, karşılaşılan farklı bir zorluğu temsil eder ve bu zorluklara katılımcıların ne ölçüde odaklandığını göstermektedir.

Tablo 5.12. Soru 1'e Ait Kodlama

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
İletişim ve İş Birliği Eksiklikleri	11	18.30%	7	41.20%	127	6.60%
Otomasyon Eksiklikleri	12	20.00%	9	52.90%	102	5.30%
İzleme ve Kayıt	4	6.70%	4	23.50%	32	1.70%
CI/CD Süreçlerinde Zorluklar	8	13.30%	8	47.10%	93	4.80%
Kültürel Değişim ve Direnç	4	6.70%	3	17.60%	92	4.80%
Güvenlik Entegrasyonu Yetersizliği	7	11.70%	6	35.30%	55	2.80%
Ölçeklenebilirlik ve Performans Sorunları	5	8.30%	5	29.40%	28	1.40%
Konteynerizasyon ve Orkestrasyon Eksikliği	5	8.30%	5	29.40%	37	1.90%
Yavaş Geri Bildirim Döngüleri	4	6.70%	4	23.50%	45	2.30%

Tablo 5.12. Açıklaması:

1. İletişim ve İş Birliği Eksiklikleri:

- Kod Sayısı: 11, toplam kodların %18.30'unu oluşturuyor.
- Katılımcı Oranı: %41.20, bu durumun paydaşlar arasında yaygın bir sorun olduğunu göstermektedir.
- Kelime Oranı: %6.60, bu konunun tartışmalarda geniş yer bulduğunu gösteriyor.

2. Otomasyon Eksiklikleri:

- Kod Sayısı: 12 ile en fazla belirtilen zorluktur (%20.00).
- Katılımcı Oranı: %52.90, katılımcıların yarısından fazlası bu sorunu dile getirmiştir.
- Kelime Oranı: %5.30, tartışmaların önemli bir kısmını oluşturmuştur.

3. İzleme ve Kayıt Eksiklikleri:

- Kod Sayısı: 4, kodların %6.70'ini temsil ediyor.
- Katılımcı Oranı: %23.50, nispeten daha az katılımcının belirttiği bir sorun.

- Kelime Oranı: %1.70, bu konunun tartışmalarda sınırlı bir yer bulduğunu gösteriyor.

4. CI/CD Süreçlerinde Zorluklar:

- Kod Sayısı: 8, kodların %13.30'unu oluşturuyor.
- Katılımcı Oranı: %47.10, bu zorlukla ilgili farkındalığın yüksek olduğunu gösteriyor.
- Kelime Oranı: %4.80, bu konunun mülakatlarda sıklıkla tartışıldığını yansıtıyor.

5. Kültürel Değişim ve Direnç:

- Kod Sayısı: 4, toplamın %6.70'i.
- Katılımcı Oranı: %17.60, katılımcıların daha az bir kısmının dile getirdiği bir konu.
- Kelime Oranı: %4.80, içerik yoğunluğu açısından diğer sorunlarla benzerlik göstermektedir.

6. Güvenlik Entegrasyonu Yetersizliği:

- Kod Sayısı: 7, kodların %11.70'ini oluşturuyor.
- Katılımcı Oranı: %35.30, güvenliğin DevOps süreçlerinde önemli bir konu olduğunu yansıtıyor.
- Kelime Oranı: %2.80, diğer zorluklara göre tartışma yoğunluğu daha düşük.

7. Ölçeklenebilirlik ve Performans Sorunları:

- Kod Sayısı: 5, %8.30 oranında belirtilmiş.
- Katılımcı Oranı: %29.40, orta seviyede katılımcı tarafından dile getirilmiştir.
- Kelime Oranı: %1.40, bu konunun tartışmalarda sınırlı yer bulduğunu gösteriyor.

8. Konteynerizasyon ve Orkestrasyon Eksiklikleri:

- Kod Sayısı: 5, toplamın %8.30'u.
- Katılımcı Oranı: %29.40, önemli bir oranı temsil etmektedir.

- Kelime Oranı: %1.90, bu konuya ayrılan yerin orta seviyede olduğunu gösteriyor.

9. Yavaş Geri Bildirim Döngüleri:

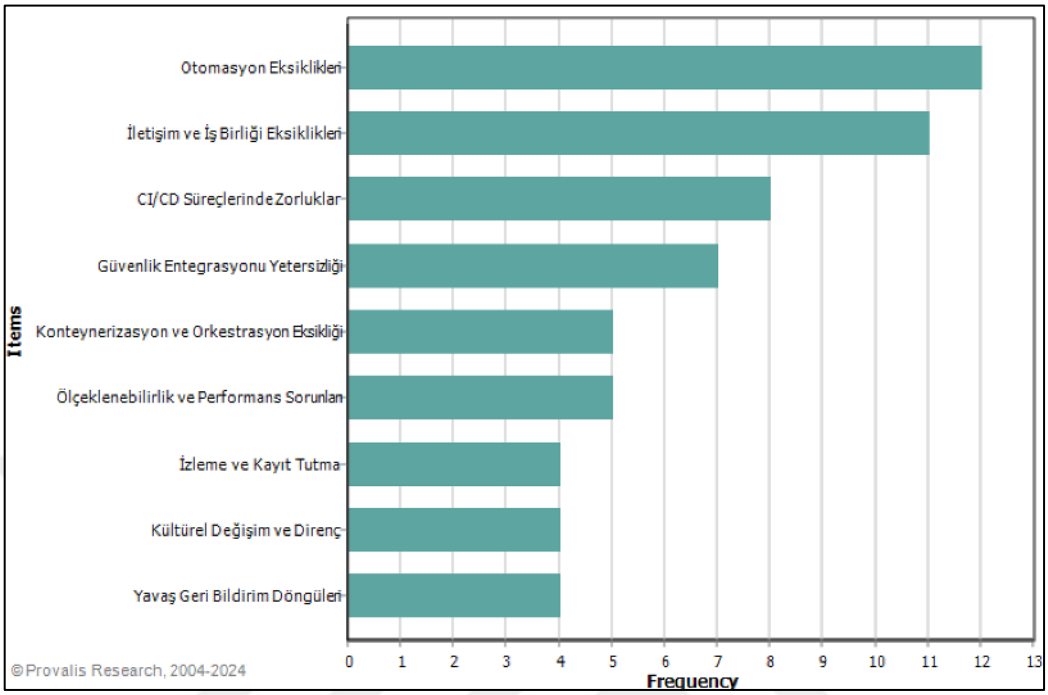
- Kod Sayısı: 4, kodların %6.70'ini oluşturuyor.
- Katılımcı Oranı: %23.50, bazı katılımcılar tarafından dile getirilmiştir.
- Kelime Oranı: %2.30, tartışmalar içinde orta derecede yer almıştır.

Bu analiz, DevOps uygulamalarında karşılaşılan zorlukların önem derecelerini ve bunların katılımcılar üzerindeki etkisini detaylı bir şekilde ortaya koymaktadır. Otomasyon eksiklikleri ve iletişim ile iş birliği sorunları, en fazla dile getirilen konular arasında yer alırken, ölçeklenebilirlik ve konteynerizasyon gibi teknik konular daha az vurgulanmıştır. Bu verilerin, DevOps süreçlerinin iyileştirilmesinde öncelik verilmesi gereken alanları belirlemek için kullanılabileceği değerlendirilmektedir.

Şekil 5.1. QDA Miner Lite Soru 1'e Ait Kod Bulutu Görseli

Şekil 5.1, QDA Miner Lite yazılımı ile yapılan analiz sonucunda ortaya çıkan kod bulutunu temsil etmektedir. Kod bulutu, mülakatlarda sıkça belirtilen temaları ve zorlukları, kodların frekansına bağlı olarak görsel bir şekilde sunmaktadır. Kodların büyüklüğü, ilgili temanın analizdeki önem derecesini ve ne sıklıkla dile getirildiğini ifade etmektedir. Bu kod bulutu, DevOps uygulamalarında karşılaşılan zorlukların sıklığını ve önem derecelerini görselleştirerek, hangi alanlarda iyileştirmeye ihtiyaç duyulduğunu kolayca anlamaya olanak tanımaktadır. Görseldeki temaların büyüklük farkları, otomasyon eksiklikleri ve iletişim sorunlarının öncelikli iyileştirme alanları olduğunu, diğer teknik ve organizasyonel

sorunların ise ikincil düzeyde ele alınması gerektiğini göstermektedir. Bu bulgu, DevOps süreçlerinde odaklanılması gereken öncelikli alanların belirlenmesine katkı sağlamaktadır.



Şekil 5.2. QDA Miner Lite Soru 1'e Ait Kod Dağılımı Grafiği

Şekil 5.2'de yer alan grafik, QDA Miner Lite yazılımı kullanılarak yapılan analizde, DevOps süreçlerinde karşılaşılan zorlukların kod frekanslarını görselleştirmektedir. Grafikte yer alan her bir kod, ilgili sorunun mülakatlarda ne sıklıkla dile getirildiğini temsil etmektedir. Frekans dağılımı, her zorluğun göreceli önem derecesini ve katılımcılar arasında hangi sorunların daha yaygın olduğunu anlamamıza yardımcı olmaktadır. Grafik, DevOps süreçlerinde hangi sorunların daha yaygın olduğunu ve hangi konuların öncelikli olarak ele alınması gerektiğini net bir şekilde ortaya koymaktadır. Özellikle otomasyon eksiklikleri ve iletişim sorunlarının çözülmesi, projelerdeki başarı oranını artırmada önemli bir rol oynayacaktır. Daha düşük frekanslı kodlar ise, dikkat edilmesi gereken ancak görece daha az yaygın olan sorunları yansıtmaktadır. Bu görselin, DevOps süreçlerinin iyileştirilmesi için önceliklendirme yapılırken rehberlik sağlayabileceği değerlendirilmektedir.

Tablo 5.13, DevOps süreçlerinde karşılaşılan zorluklar ve bu zorluklara yönelik çözüm önerilerini içermektedir. DevOps'un etkin bir şekilde uygulanabilmesi için, karşılaşılan operasyonel ve teknik sorunların belirlenmesi ve bu sorunlara uygun araçlarla çözümler

geliştirilmesi gereklidir. Tabloda, her bir zorluk başlığına ilişkin açıklamalar, önerilen çözümler ve kaynaklar sunulmaktadır.

Tablo 5.13. Soru 1'e Ait Tematik Analiz

Tema	Açıklama	Önerilen Çözümler ve Araçlar	Kaynak (APA)
İletişim ve İş Birliği Eksiklikleri	Geliştirici ve operasyon ekipleri arasında iletişim eksikliği, projelerin gecikmesine veya hatalı sonuçlara yol açması.	Slack, Jira gibi iletişim araçları ile ekipler arası iş birliğini artırmak.	Akbar, Naveed ve ark. (2020)
Otomasyon Eksiklikleri	Manuel süreçlerin fazlalığı, hatalara ve zaman kaybına yol açar. CI/CD süreçlerinde otomatizasyon eksikliği.	CI/CD araçları kullanarak süreçlerin otomatikleştirilmesi; Jenkins, GitHub Actions, Terraform gibi araçlar.	Khan ve ark. (2022), El Aouni ve ark. (2024)
İzleme ve Kayıt	Sistem performansının yeterince izlenememesi ve sorunların proaktif olarak tespit edilememesi.	Prometheus, Grafana, ELK Stack gibi monitoring araçları ile sistemlerin izlenmesi ve log yönetimi.	Almeida ve Lopes ve ark. (2022), Gashaw ve Beshah (2022)
CI/CD Süreçlerinde Zorluklar	CI/CD pipeline'larının tam otomatik olmaması, kapsamlı testlerin yapılmaması ve dağıtımların manuel adımlar içermesi.	CI/CD süreçlerini tamamen otomatikleştirerek, Jenkins, GitLab CI, Travis CI gibi araçları kullanmak.	Khan ve Shameem (2020)
Kültürel Değişim ve Direnç	Ekiplerin DevOps'a adaptasyonu sırasında karşılaşılan direnç, geleneksel yöntemlere bağlılık.	Eğitimler ve rehberler ile ekiplerin yeni yöntemlere adaptasyonunu sağlamak; Atlassian Playbook gibi araçlar.	Jayakody ve Wijayanayake (2021), Chiari ve ark. (2023)
Güvenlik Entegrasyonu (DevSecOps)	Güvenliğin erken aşamalarda geliştirme süreçlerine entegre edilmemesi ve güvenlik açıklarının fark edilmemesi.	DevSecOps yaklaşımlarını benimseyerek, OWASP ZAP, SonarQube, Snyk gibi araçlar kullanmak.	Battina, D. S. (2019), Fawzy, Wassif ve Moussa (2023)
Ölçeklenebilirlik ve Performans Sorunları	Artan kullanıcı taleplerine karşı sistemlerin ölçeklenememesi veya performans düşüklükleri.	Kubernetes ve Docker Swarm gibi container yönetim araçları ile ölçeklenebilirliği artırmak.	Eramo ve ark. (2024), Bolscher ve Daneva (2019)
Konteynerizasyon ve Orkestrasyon	Konteyner tabanlı yapıların sağlanamaması veya orkestrasyonun zayıf olması.	Kubernetes, Docker gibi araçlar ile konteynerizasyon ve otomatik dağıtım süreçlerini iyileştirmek.	Ibrahim ve ark. (2019)
Yavaş Geri Bildirim Döngüleri	Geliştirici ve operasyon ekipleri arasındaki geri bildirim döngüsünün yavaş olması ve müşteri taleplerine hızlı yanıt verilememesi.	Otomatik geri bildirim sistemleri ve düzenli retrospektifler ile süreç iyileştirmeleri yapmak.	Cuadra ve ark. (2024), Gokarna ve Singh (2021)

5.2.2. Uyarlama tematik analizi

DevOps proje yönetim çerçevesi süreçlerinin ilgili yazılım projelerine (kurumsal/standart yazılım, görev-kritik, güvenlik-kritik sistem yazılımları), (b) organizasyona ve (c) proje takımına uyarlarken karşılaştığınız güçlükler ve kullandığınız çözümler nelerdir?

Amaç: Kurumda mevcut haliyle kullanılan DevOps proje yönetim çerçevesinin projeye, organizasyona ve proje takımına uyarlanmasıyla ilgili durumların belirlenmesi

Tablo 5,14, DevOps süreçlerinde organizasyonel ve teknik uyarlamalara yönelik karşılaşılan zorlukların kod frekanslarını, katılımcı oranlarını ve kelime oranlarını göstermektedir. Kod analizi, her bir zorluğun önem derecesini ve paydaşlar arasında ne sıklıkla dile getirildiğini anlamaya olanak tanımaktadır.

Tablo 5.14. Soru 2'ye Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Kurumsal/Standart Yazılımlara Uyarlama	6	9.50%	5	29.40%	114	4.80%
Görev-Kritik Yazılımlara Uyarlama	5	7.90%	4	23.50%	121	5.00%
Güvenlik-Kritik Yazılımlara Uyarlama	4	6.30%	4	23.50%	91	3.80%
Organizasyona Uyarlama - Kültürel Direnç	7	11.10%	5	29.40%	108	4.50%
Organizasyona Uyarlama - Mevcut Altyapı ile Uyum	7	11.10%	6	35.30%	161	6.70%
Organizasyona Uyarlama - Siloların Kırılması	10	15.90%	7	41.20%	300	12.50%
Proje Takımına Uyarlama - Ekip Becerileri	7	11.10%	5	29.40%	129	5.40%
Proje Takımına Uyarlama - İletişim ve İş Birliği	8	12.70%	7	41.20%	140	5.80%
Proje Takımına Uyarlama - Çevik Yöntemlere Uyum	9	14.30%	8	47.10%	200	8.30%

Tablo 5.14 Analizi:

1. Kurumsal/Standart Yazılımlara Uyarlama:

- Kod Sayısı: 6 (%9.50).
- Katılımcı Oranı: %29.40.
- Kelime Oranı: %4.80.

Kurumsal yazılımların değişim süreçlerinde yavaşlık ve manuel süreçler, yaygın bir sorun olarak dile getirilmiştir.

2. Görev-Kritik Yazılımlara Uyarlama:

- Kod Sayısı: 5 (%7.90).
- Katılımcı Oranı: %23.50.
- Kelime Oranı: %5.00.

Görev-kritik yazılımlarda hata toleransı düşük olduğundan, güvenilirlik gereksinimi öne çıkmıştır.

3. Güvenlik-Kritik Yazılımlara Uyarlama:

- Kod Sayısı: 4 (%6.30).
- Katılımcı Oranı: %23.50.
- Kelime Oranı: %3.80.

Güvenlik açıklarının sürekli test edilmesi gerekliliği vurgulanmıştır.

4. Organizasyona Uyarlama - Kültürel Direnç:

- Kod Sayısı: 7 (%11.10).
- Katılımcı Oranı: %29.40.
- Kelime Oranı: %4.50.

Kültürel dönüşüm ve geleneksel yöntemlere direnç, önemli organizasyonel engeller arasında yer almıştır.

5. Organizasyona Uyarlama - Mevcut Altyapı ile Uyum:

- Kod Sayısı: 7 (%11.10).
- Katılımcı Oranı: %35.30.
- Kelime Oranı: %6.70.

Eski sistemlerin DevOps süreçlerine entegrasyonu zorlayıcı bir problem olarak belirtilmiştir.

6. Organizasyona Uyarlama - Siloların Kırılması:

- Kod Sayısı: 10 (%15.90).
- Katılımcı Oranı: %41.20.
- Kelime Oranı: %12.50.

Organizasyonel siloların ortadan kaldırılması, bilgi paylaşımı ve süreçlerin hızlanması için kritik bir zorluk olarak öne çıkmıştır.

7. Proje Takımına Uyarlama - Ekip Becerileri:

- Kod Sayısı: 7 (%11.10).
- Katılımcı Oranı: %29.40.
- Kelime Oranı: %5.40.

DevOps araçlarına yönelik ekip üyelerinin eğitim gereksinimleri sıkça dile getirilmiştir.

8. Proje Takımına Uyarlama - İletişim ve İş Birliği:

- Kod Sayısı: 8 (%12.70).
- Katılımcı Oranı: %41.20.
- Kelime Oranı: %5.80.

Ekipler arası iletişim eksikliği, iş birliğinin artırılması gereken bir konu olarak belirtilmiştir.

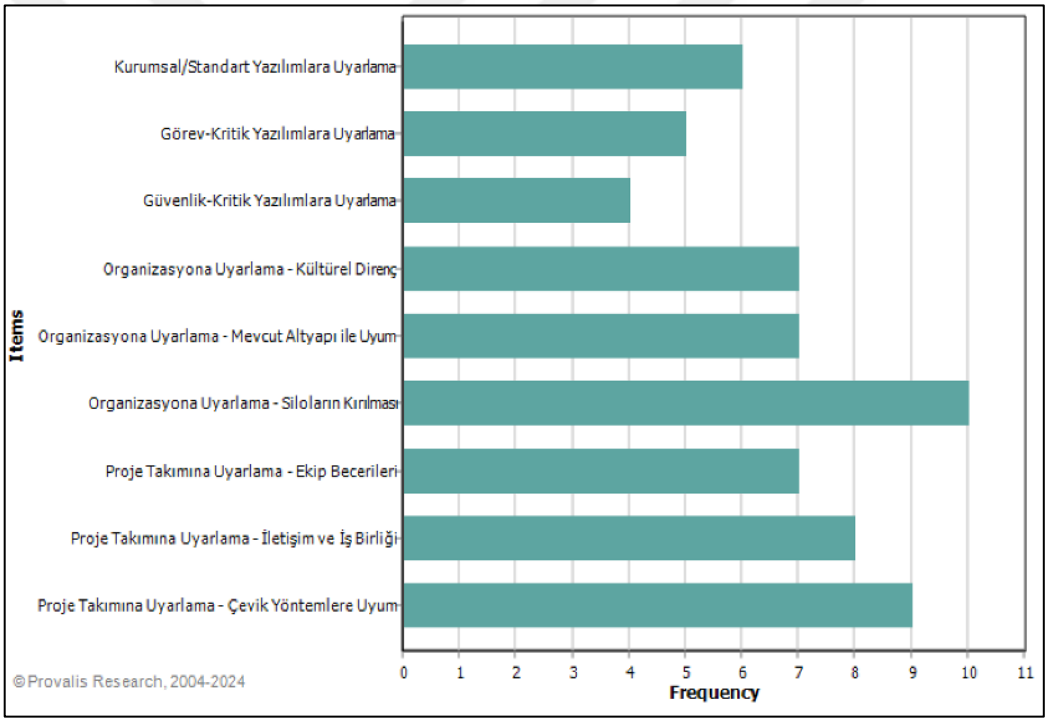
9. Proje Takımına Uyarlama - Çevik Yöntemlere Uyum:

- Kod Sayısı: 9 (%14.30).
- Katılımcı Oranı: %47.10.

- Kelime Oranı: %8.30.

Çevik yöntemlere adaptasyonda yaşanan uyumsuzluklar, proje takımlarında öne çıkan bir zorluk olmuştur.

DevOps uyarlama süreçlerinde farklı seviyelerdeki zorlukları, bunların frekanslarını ve önem derecelerini detaylı bir şekilde sunmaktadır. Organizasyona Uyarlama - Siloların Kırılması ve Proje Takımına Uyarlama - Çevik Yöntemlere Uyum, en sık dile getirilen zorluklar olarak öne çıkmıştır. Bu bulguların, DevOps'un organizasyonel ve teknik uyum süreçlerinde öncelik verilmesi gereken alanları belirlemek açısından yol gösterici olduğu değerlendirilmektedir.



Şekil 5.3. QDA Miner Lite Soru 2'ye Ait Kod Dağılımı Grafiği

Şekil 5.3'deki grafik, QDA Miner Lite yazılımı ile yapılan analiz sonucunda, DevOps süreçlerinde karşılaşılan organizasyonel ve teknik uyarlama zorluklarının kod frekanslarını görselleştirmektedir. Grafik, organizasyonel ve teknik uyarlama zorluklarının önceliklendirilmesi gereken alanlarını net bir şekilde ortaya koymaktadır. Siloların Kırılması, Çevik Yöntemlere Uyum ve İletişim ve İş Birliği, en sık dile getirilen sorunlar olup, bu alanlara öncelik verilmesi gerektiği görülmektedir. Daha düşük frekansa sahip

zorluklar ise, uzun vadede çözüm için ele alınması gereken alanları işaret etmektedir. Bu tür analizler, DevOps uygulamalarının daha etkin hale getirilmesi için kritik veriler sunmaktadır.

Şekil 5.4. QDA Miner Lite Soru 2'ye Ait Kod Bulutu Görseli

Şekil 5.4'de yer alan kod bulutu görseli, DevOps süreçlerinde organizasyonel ve teknik uyarlama zorluklarını temsil eden kodların sıklık ve önem derecesine göre görselleştirilmiş halidir. Kod bulutu, DevOps süreçlerinde organizasyonel ve teknik uyarlama zorluklarının önceliklendirilmesine ışık tutmaktadır. Siloların Kırılması, Çevik Yöntemlere Uyum ve İletişim ve İş Birliği, en büyük ve en sık dile getirilen sorunlar olarak öne çıkmaktadır. Bu bulgular, DevOps uygulamalarının iyileştirilmesi için hangi alanlara öncelik verilmesi gerektiğini net bir şekilde ortaya koymaktadır. Daha küçük kodlar ise, uzun vadede ele alınması gereken ancak daha az yaygın olan zorlukları ifade etmektedir.

Tablo 5.15, DevOps süreçlerinde organizasyonel ve teknik uyarlamalara yönelik karşılaşılan zorlukları, bu zorluklara ilişkin çözüm önerilerini ve kaynakları içermektedir. DevOps'un etkili bir şekilde uygulanabilmesi için organizasyonel yapılar, mevcut altyapılar ve proje takımları gibi farklı bileşenlerde karşılaşılan uyum sorunlarına özel çözümler geliştirilmesi gerekmektedir. DevOps süreçlerine uyum sağlamak için organizasyonel, teknik ve kültürel boyutlardaki zorlukların analizini ve bu zorluklara yönelik somut çözüm önerilerini kapsamaktadır. Özellikle kültürel direnç, mevcut altyapı ile uyum ve görev-kritik sistemler gibi alanlarda geliştirilen çözümler, DevOps'un etkin bir şekilde benimsenmesine katkı sağlayacaktır.

Tablo 5.15. Soru 2'ye Ait Tematik Analiz

Tema	Açıklama	Önerilen Çözümler ve Araçlar	Kaynak
Kurumsal/Standart Yazılımlara Uyarlama	Kurumsal yazılımlarda değişikliklerin yavaş yapılması, manuel süreçlerin yaygın olması ve otomasyona geçişin zor olması.	DevOps süreçlerinin kademeli otomatikleştirilmesi; CI/CD araçları kullanarak süreçleri hızlandırmak.	Eramo, Tucci ve ark. (2024), Fernandes ve ark. (2022)
Görev-Kritik Yazılımlara Uyarlama	Görev-kritik sistemlerde yapılan hataların büyük sonuçlara yol açma riski, yüksek güvenilirlik gereksinimi.	Test Otomasyonu ve Blue-Green Deployment stratejileri ile minimum kesinti süresi sağlamak.	Karunarathne, Wijayanayake ve Prasadika (2024)
Güvenlik-Kritik Yazılımlara Uyarlama	Güvenlik-kritik sistemlerde değişikliklerin ciddi güvenlik riskleri taşıması, güvenlik açıklarının sürekli test edilmesi.	DevSecOps yaklaşımlarını benimseyerek, güvenlik süreçlerini CI/CD'ye entegre etmek ve güvenlik testlerini otomatikleştirmek.	Eramo, Tucci ve ark. (2024)
Organizasyona Uyarlama - Kültürel Direnç	Ekipler arasında DevOps kültürüne adaptasyon sürecinde direnç gösterilmesi ve geleneksel yaklaşımların devam etmesi.	Eğitim ve kültürel değişim yönetimi; ekiplerin DevOps kültürüne adaptasyonunu sağlamak ve liderlerin değişimi teşvik etmesi.	Krey (2022), Bijwe ve Shankar (2022)
Organizasyona Uyarlama - Mevcut Altyapı ile Uyum	Eski sistemlerin DevOps süreçleri ile uyum sağlaması zorlukları, monolitik mimarilerin esneklik eksikliği.	Kapsayıcılar ve Mikroservis Mimarisi kullanarak eski sistemlerin modernizasyonu; Docker, Kubernetes gibi araçlar.	Bijwe ve Shankar (2022), Hemon-Hildgen, Rowe ve Monnier-Senicourt (2020)
Organizasyona Uyarlama - Siloların Kırılması	Ekipler arasında bilgi paylaşımı ve iş birliği eksikliği, süreçlerde gecikmelere yol açması.	Çapraz fonksiyonel ekipler oluşturarak bilgi paylaşımını ve iş birliğini teşvik etmek.	Fernandes ve ark. (2020), Bollieddula (2022)
Proje Takımına Uyarlama - Ekip Becerileri	Ekip üyelerinin DevOps süreçlerine ve otomasyon araçlarına olan deneyimsizliği.	Eğitim programları düzenleyerek CI/CD, konteyner yönetimi ve bulut platformları hakkında beceriler kazandırmak.	Hrusto, Engström ve Runeson (2023), Firdaouss ve ark. (2022)
Proje Takımına Uyarlama - İletişim ve İş Birliği	Ekipler arasında iletişimin zayıf olması ve iş birliğinin zorlaşması.	Agile Pratikler ve ChatOps araçları kullanarak iletişimi ve iş birliğini geliştirmek.	Chiari ve ark. (2023), Azad ve Hyrynsalmi (2023)
Proje Takımına Uyarlama - Çevik Yöntemlere Uyum	Takımların çevik çalışma yöntemlerine geçişte zorlanması ve uyumsuzluklar.	Scrum, Kanban gibi çevik yöntemlerin temel ilkeleri öğretilmeli ve adım adım çevik geçiş yapılmalı.	Almeida, Simões ve Lopes (2022)

5.2.3. Takım yönetimi tematik analizi

Proje Takımı Yönetimi kapsamında DevOps proje yönetim çerçevesi uygularken takım yönetimiyle ilgili konularda nelere dikkat ediyorsunuz? Karşılaştığınız sorunlara çözüm amacıyla kullandığınız yöntem ve teknikler nelerdir (takım kültürü, takımın oluşturulması, rollerin belirlenmesi, eğitim vb.)

Amaç: Proje Takım Yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.

Tablo 5.16, DevOps süreçlerinde takım yönetimiyle ilgili zorlukların kodlama frekanslarını, katılımcı oranlarını ve kelime oranlarını göstermektedir. Her bir kod, takım yönetiminde öne çıkan sorunları temsil ederken, frekans ve oranlar bu zorlukların ne kadar sıklıkla dile getirildiğini yansıtmaktadır.

Tablo 5.16. Soru 3'e Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Takım Kültürü ve İş Birliği	8	12.70%	8	47.10%	200	12.30%
Takımın Oluşturulması ve Roller	8	12.70%	8	47.10%	209	12.90%
Eğitim ve Bilgi Paylaşımı	13	20.60%	11	64.70%	195	12.00%
Takım İçi İletişim ve Koordinasyon	7	11.10%	5	29.40%	204	12.60%
Roller ve Sorumluluk Belirsizliği	9	14.30%	9	52.90%	141	8.70%
Takım Dinamikleri ve Esneklik	8	12.70%	8	47.10%	228	14.10%
Şeffaflık ve Ortak İletişim Dili	8	12.70%	7	41.20%	168	10.40%

Tablo 5.16 Analizi:

1. Eğitim ve Bilgi Paylaşımı:

- Kod Sayısı: 13 (%20.60).
- Katılımcı Oranı: %64.70.
- Kelime Oranı: %12.00.

Bu kod, en yüksek frekansla belirtilen zorluk olarak dikkat çekmektedir. Takım üyelerinin yeni teknolojilere yönelik eğitimi ve bilgi paylaşımı ihtiyacı, DevOps süreçlerinde kritik bir gereksinimdir.

2. Roller ve Sorumluluk Belirsizliği:

- Kod Sayısı: 9 (%14.30).
- Katılımcı Oranı: %52.90.
- Kelime Oranı: %8.70.

Roller ve görev dağılımındaki belirsizlikler, DevOps ekiplerinin etkin çalışmasını olumsuz etkileyen bir diğer önemli konudur.

3. Takım Dinamikleri ve Esneklik:

- Kod Sayısı: 8 (%12.70).
- Katılımcı Oranı: %47.10.
- Kelime Oranı: %14.10.

Takımların esnek ve güvene dayalı bir ortamda çalışması ihtiyacı, takım dinamiklerini iyileştirme açısından sıklıkla vurgulanmıştır.

4. Takım Kültürü ve İş Birliği:

- Kod Sayısı: 8 (%12.70).
- Katılımcı Oranı: %47.10.
- Kelime Oranı: %12.30.

Takımlar arasındaki iş birliği ve sinerjiyi artırma gereksinimi, mülakatlarda öne çıkan bir diğer önemli başlıktır.

5. Takımın Oluşturulması ve Roller:

- Kod Sayısı: 8 (%12.70).
- Katılımcı Oranı: %47.10.
- Kelime Oranı: %12.90.

Ekip üyelerinin rollerinin net tanımlanması ve çapraz fonksiyonel ekiplerin oluşturulması, DevOps ekiplerinin etkinliği için önemli bir gereksinimdir.

6. Takım İçi İletişim ve Koordinasyon:

- Kod Sayısı: 7 (%11.10).
- Katılımcı Oranı: %29.40.

- Kelime Oranı: %12.60.

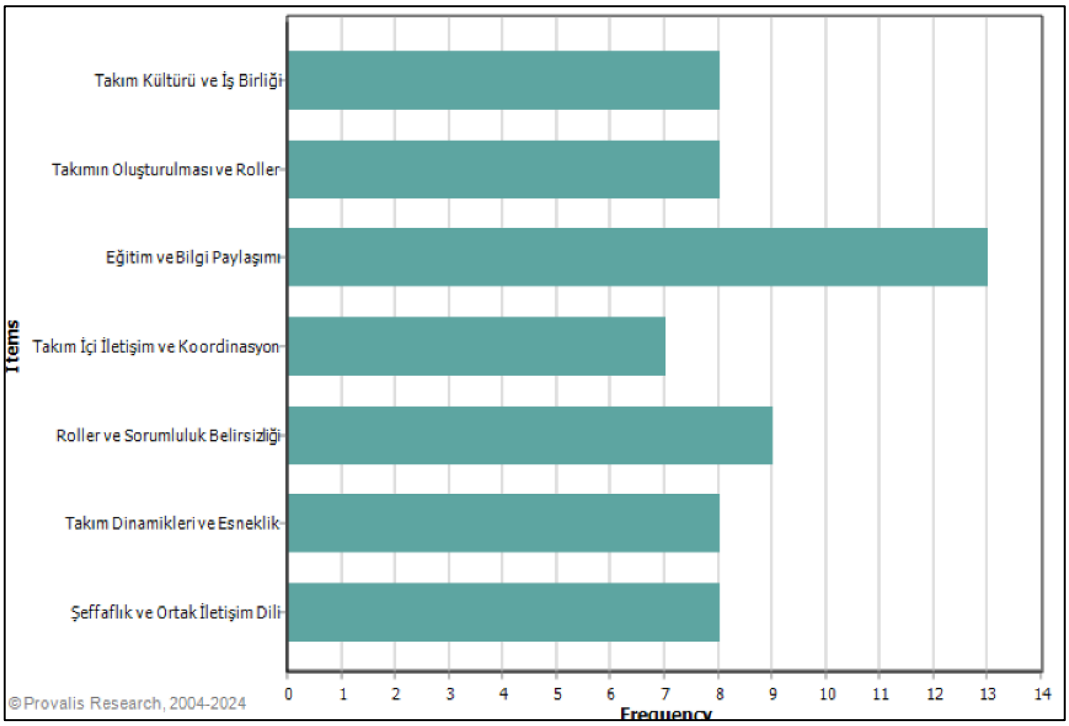
İletişim kanallarının güçlendirilmesi ve koordinasyonun artırılması, diğer zorluklara göre daha az sıklıkla dile getirilmiş, ancak önemli bir konudur.

7. Şeffaflık ve Ortak İletişim Dili:

- Kod Sayısı: 8 (%12.70).
- Katılımcı Oranı: %41.20.
- Kelime Oranı: %10.40.

Takım süreçlerinde şeffaflık sağlanması ve ortak bir iletişim dili geliştirilmesi gerekliliği de sıkça ifade edilmiştir.

Eğitim ve Bilgi Paylaşımı, en sık dile getirilen zorluk olarak dikkat çekerken, Roller ve Sorumluluk Belirsizliği ve Takım Dinamikleri ve Esneklik gibi konular da önemli zorluklar arasında yer almaktadır. Bu veriler, DevOps ekiplerinin etkinliği ve uyumunu artırmak için hangi alanlara öncelik verilmesi gerektiğini belirlemeye yardımcı olmaktadır.



Şekil 5.5. QDA Miner Lite Soru 3'e Ait Kod Dağılımı Grafiği

Şekil 5.5'te yer alan grafik, QDA Miner Lite yazılımı ile yapılan analizde, DevOps süreçlerinde takım yönetimiyle ilgili zorlukların kodlama frekanslarını görselleştirmektedir. DevOps ekiplerinin yönetiminde karşılaşılan zorlukların önceliklendirilmesi için net bir tablo sunmaktadır. Eğitim ve Bilgi Paylaşımı, en sık dile getirilen zorluk olarak öne çıkarken, Roller ve Sorumluluk Belirsizliği ve Takım Kültürü ve İş Birliği gibi temalar da önemini korumaktadır. Bu tür analizler, DevOps ekiplerinin daha verimli çalışması için hangi alanlara odaklanılması gerektiğini göstermektedir.

Şekil 5.6. QDA Miner Lite Soru 3'e Ait Kod Bulutu Görseli

Şekil 5.6'daki kod bulutu görseli, DevOps süreçlerinde takım yönetimi ve iş birliğine yönelik zorlukların, mülakatlarda ne sıklıkta vurgulandığını görsel olarak temsil etmektedir. Bu kod bulutu, DevOps süreçlerinde takım yönetimi ve iş birliğinde karşılaşılan zorlukların önceliklendirilmesini kolaylaştırmaktadır. Eğitim ve Bilgi Paylaşımı, en önemli zorluk olarak öne çıkarken, Roller ve Sorumluluk Belirsizliği, Takım Kültürü ve İş Birliği gibi konular da önemli zorluklar arasında yer almaktadır. Daha küçük kodlar ise, çözülmesi gereken ancak daha az yaygın olan zorlukları temsil etmektedir. Bu bulgu, DevOps ekiplerinin etkinliğini artırmak için odaklanılması gereken temel alanları göstermektedir.

Tablo 5.17, DevOps süreçlerinde takım yönetimi ve iş birliğine ilişkin karşılaşılan zorlukları, önerilen çözümleri ve bu çözümlerin dayandığı kaynakları sunmaktadır. Takım yönetimindeki temel unsurlar, iletişim, rol tanımları, eğitim ve esneklik gibi alanlarda karşılaşılan sorunların çözümüne yönelik somut yaklaşımlar içermektedir. DevOps takım yönetimi ve iş birliğinde karşılaşılan zorluklara yönelik kapsamlı bir çözüm önerisi seti

sunmaktadır. Takım Kültürü ve İş Birliği, Eğitim ve Bilgi Paylaşımı, Roller ve Sorumluluklar, DevOps ekiplerinin etkinliğini artırmak için en sık vurgulanan alanlardır. Kaynaklara dayalı çözüm önerileri, DevOps uygulamalarında karşılaşılan sorunları çözmek için somut adımlar atılmasını sağlamaktadır.

Tablo 5.17. Soru 3'e Ait Tematik Analiz

Tema	**Açıklama**	**Önerilen Çözümler ve Araçlar**	**Kaynak**
Takım Kültürü ve İş Birliği	Takımlar arası iş birliği ve sinerjiyi artırmak, sürekli geri bildirim döngüleri kurarak süreçlerin iyileştirilmesini sağlamak.	Düzenli sprint retrospektifleri, ortak hedef belirleme ve geri bildirim döngüleri. Slack, Jira gibi iş birliği araçları.	Eramo, Tucci ve ark. (2024), Bijwe ve Shankar (2022)
Takımın Oluşturulması ve Roller	Takım üyelerinin rollerinin net belirlenmesi, çapraz fonksiyonel ekipler oluşturmak ve sorumlulukların açıkça tanımlanması.	Rollerin açıkça tanımlandığı bir yapı kurmak, Confluence gibi dokümantasyon araçları ve Jira ile görev yönetimi sağlamak.	Avritzer (2020), Hasselbring ve ark. (2019)
Eğitim ve Bilgi Paylaşımı	DevOps süreçleri ve yeni teknolojiler konusunda düzenli eğitimler sağlamak, bilgi paylaşım oturumları düzenlemek.	Pluralsight, Coursera gibi platformlar ile eğitimler, mentor programları ve CI/CD araçları kullanarak yetkinlik geliştirmek.	Koren ve ark. (2023), Jha ve ark. (2023)
Takım İçi İletişim ve Koordinasyon	Takım içindeki tüm süreçlerde açıklık ve şeffaflık sağlamak, iletişim kanallarının güçlendirilmesi.	Jira, Trello gibi çevik yönetim araçları ve Slack, Teams gibi iletişim araçlarını kullanmak.	Antunes ve Tate (2024), Grande, Vizcaíno ve García (2024)
Roller ve Sorumluluk Belirsizliği	RACI Matrisi kullanarak rollerin netleştirilmesi ve görev dağılımının yapılması.	RACI Matrisi (Responsible, Accountable, Consulted, Informed) yöntemi ile rollerin netleştirilmesi.	Fernandes ve ark. (2020), Acevedo-Dueñas, Muñoz ve Galván-Cruz (2023)
Takım Dinamikleri ve Esneklik	Takım üyelerinin sorumluluk alması ve gerektiğinde esnek bir şekilde çalışması, güven ve adaletin sağlanması.	Güven ve sorumluluğun teşvik edildiği bir kültür oluşturmak, esnek çalışma modelleri uygulamak.	Bolscher ve Daneva (2019)
Şeffaflık ve Ortak İletişim Dili	Ortak bir iletişim dili geliştirilmesi, işlemlerin standart hale getirilmesi.	Ansible ve CI/CD araçları kullanarak işlemleri standart hale getirmek, ortak bir iletişim dili oluşturmak.	Akbar ve ark. (2023)

5.2.4. Gereksinim yönetimi tematik analizi

Soru: “DevOps proje yönetim çerçevesini uygularken İhtiyaç Mühendisliği ve Yazılım Yönetimi kapsamında karşılaştığınız problemler ve bunlara yönelik kullandığınız çözümler nelerdir?” (İhtiyaç mühendisliği: Paydaş yönetimi, ihtiyaçların belirlenmesi, analizi ve önceliklendirilmesi, ihtiyaçların izlenmesi, ihtiyaçların doğrulanması ve geçerlemesi vb.konular)

Amaç: İhtiyaç Mühendisliği ve Yazılım Yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.

Tablo 5.18, yazılım projelerinde gereksinim yönetimi ve entegrasyon süreçlerinde karşılaşılan zorluklara yönelik kodlama frekanslarını, katılımcı oranlarını ve kelime oranlarını göstermektedir. Kodlama analizi, bu zorlukların projelerdeki önem derecesini ve hangi konuların daha fazla tartışıldığını anlamamıza yardımcı olmaktadır.

Tablo 5.18. Soru 4'e Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Paydaş Yönetimi ve İletişim	8	15.40%	8	47.10%	228	12.10%
İhtiyaçların Belirlenmesi ve Önceliklendirilmesi	13	25.00%	12	70.60%	354	18.90%
İhtiyaçların İzlenmesi ve Takibi	9	17.30%	9	52.90%	187	10.00%
İhtiyaçların Doğrulanması ve Geçerlenmesi	9	17.30%	7	41.20%	237	12.60%
Yazılım Yönetimi ve Entegrasyon	5	9.60%	5	29.40%	118	6.30%
Dokümantasyon ve İzlenebilirlik	5	9.60%	3	17.60%	54	2.90%
Değişiklik Yönetimi	3	5.80%	3	17.60%	79	4.20%

Tablo 5.18 Analizi:

1. İhtiyaçların Belirlenmesi ve Önceliklendirilmesi:

- Kod Sayısı: 13 (%25.00).
- Katılımcı Oranı: %70.60.

- Kelime Oranı: %18.90.

Bu konu, en yüksek frekansa ve katılımcı oranına sahiptir. Gereksinimlerin net bir şekilde tanımlanması ve önceliklendirilmesi, projelerin başarısını doğrudan etkileyen bir unsur olarak öne çıkmıştır.

2. Paydaş Yönetimi ve İletişim:

- Kod Sayısı: 8 (%15.40).
- Katılımcı Oranı: %47.10.
- Kelime Oranı: %12.10.

Paydaşlar arasında iletişim eksiklikleri ve beklenti farklılıklarının giderilmesi, önemli bir zorluk olarak ifade edilmiştir.

3. İhtiyaçların İzlenmesi ve Takibi:

- Kod Sayısı: 9 (%17.30).
- Katılımcı Oranı: %52.90.
- Kelime Oranı: %10.00.

Gereksinim değişim süreçlerinin kontrol altına alınması ve izlenmesi gerekliliği, sıkça dile getirilen bir diğer konudur.

4. İhtiyaçların Doğrulanması ve Geçerlenmesi:

- Kod Sayısı: 9 (%17.30).
- Katılımcı Oranı: %41.20.
- Kelime Oranı: %12.60.

Gereksinimlerin doğru şekilde doğrulanması ve test edilmesi ihtiyacı, projelerin doğru yöne ilerlemesi için önemli bir faktör olarak vurgulanmıştır.

5. Yazılım Yönetimi ve Entegrasyon:

- Kod Sayısı: 5 (%9.60).
- Katılımcı Oranı: %29.40.
- Kelime Oranı: %6.30.

Yazılım yönetiminde entegrasyon problemleri ve teknolojik uyumsuzluklar, görece daha az vurgulanan bir konu olmuştur.

6. Dokümantasyon ve İzlenebilirlik:

- Kod Sayısı: 5 (%9.60).
- Katılımcı Oranı: %17.60.
- Kelime Oranı: %2.90.

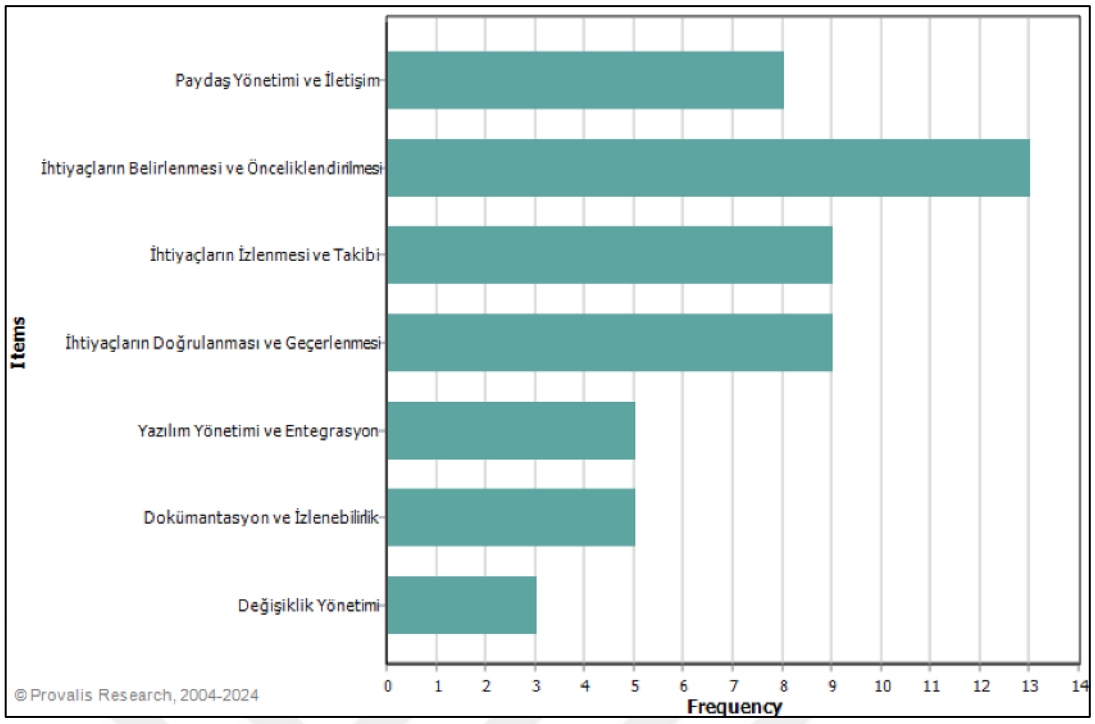
Gereksinim izlenebilirliğinin ve dokümantasyonun yetersizliği, diğer zorluklara göre daha az dile getirilmiş olsa da önemini korumaktadır.

7. Değişiklik Yönetimi:

- Kod Sayısı: 3 (%5.80).
- Katılımcı Oranı: %17.60.
- Kelime Oranı: %4.20.

Değişikliklerin doğru bir şekilde yönetilmesi, projelerin hedeflerden sapmasını önlemek için gereklidir.

İhtiyaçların Belirlenmesi ve Önceliklendirilmesi ve Paydaş Yönetimi ve İletişim, projelerin başarıyla yürütülmesi için en kritik alanlar olarak öne çıkmaktadır. Daha düşük frekansa sahip kodlar ise, uzun vadede ele alınması gereken ancak görece daha az yaygın olan zorlukları temsil etmektedir. Bu veriler, projelerin yönetim süreçlerini iyileştirmek için hangi alanlara odaklanılması gerektiğini göstermektedir.



Şekil 5.7. QDA Miner Lite Soru 4'e Ait Kod Dağılımı Grafiği

Şekil 5.7'deki grafik, yazılım projelerinde gereksinim yönetimi ve entegrasyon süreçlerinde karşılaşılan zorluklara ait kodların frekanslarını görselleştirmektedir. İhtiyaçların Belirlenmesi ve Önceliklendirilmesi, en sık dile getirilen ve çözüm gerektiren en önemli konu olarak öne çıkmaktadır. Bunun yanında, İhtiyaçların İzlenmesi ve Takibi ve Paydaş Yönetimi ve İletişim gibi temalar, projelerin başarısında önemli rol oynayan diğer konular arasında yer almaktadır. Daha düşük frekansa sahip olan kodlar ise, daha az yaygın ancak yine de projelerin başarısını etkileyen zorlukları temsil etmektedir.

Şekil 5.8. QDA Miner Lite Soru 4'e Ait Kod Bulutu Görseli

Şekil 5.8'deki kod bulutu, yazılım projelerinde gereksinim yönetimi süreçlerinde karşılaşılan zorlukları önceliklendirmek için net bir çerçeve sunmaktadır. İhtiyaçların Belirlenmesi ve Önceliklendirilmesi, yazılım projelerinin başarısını doğrudan etkileyen

birincil öneme sahip bir konu olarak öne çıkmaktadır. Bunun yanında, İhtiyaçların Doğrulanması ve Geçerlenmesi ve İhtiyaçların İzlenmesi ve Takibi gibi konular da projelerin etkin yönetimi için kritik unsurlar arasında yer almaktadır. Daha küçük kodlar ise, çözülmesi gereken ancak daha az yaygın olan zorlukları ifade etmektedir. Bu bulgu, projelerde hangi alanlara odaklanılması gerektiği konusunda rehberlik sağlamaktadır.

Tablo 5.19, yazılım projelerinde gereksinim yönetimi ve entegrasyon süreçlerinde karşılaşılan zorluklara yönelik kodlama analizi sonuçlarını içermektedir. Yazılım projelerinde gereksinim yönetimi ve entegrasyon süreçlerinde karşılaşılan temel sorunların çözümü için somut öneriler sunmaktadır. Paydaş Yönetimi ve İletişim, İhtiyaçların Belirlenmesi ve Önceliklendirilmesi ve Dokümantasyon ve İzlenebilirlik, projelerin başarısını doğrudan etkileyen konular arasında öne çıkmaktadır.

Tablo 5.19. Soru 4'e Ait Tematik Analiz

Tema	**Açıklama**	**Önerilen Çözümler ve Araçlar**	**Kaynak**
Paydaş Yönetimi ve İletişim	Paydaşlar arasında farklı beklentilerin olması ve iletişim kopuklukları, proje hedeflerinde sapmalara yol açar.	Düzenli paydaş toplantıları ve şeffaf bir iletişim ortamı oluşturmak. Jira, Confluence gibi araçlarla bilgi akışı sağlamak.	Bolscher ve Daneva (2019), Acevedo-Dueñas, Muñoz ve Galván-Cruz (2023)
İhtiyaçların Belirlenmesi ve Önceliklendirilmesi	Gereksinimlerin net olmaması ve önceliklendirme sorunları, projelerde aksamalara neden olabilir.	Kullanıcı hikayeleri ve kabul kriterlerini belirlemek, MoSCoW gibi önceliklendirme teknikleri kullanmak.	Alnafessah ve ark. (2021), Bollieddula (2022)
İhtiyaçların İzlenmesi ve Takibi	Gereksinimlerin değişim sürecinin izlenmesi ve sürüm kontrolü zor olabilir.	Gereksinim takibi için güçlü izleme sistemleri kurmak. Jira veya Trello gibi araçlar ile gereksinimleri izlemek.	Faustino ve ark. (2022), Avritzer (2020)
İhtiyaçların Doğrulanması ve Geçerlenmesi	Yanlış yorumlanan gereksinimler veya eksik doğrulamalar, hatalı geliştirmelere yol açar.	Test otomasyonu ve kullanıcı kabul testlerini yapmak. Selenium, JUnit gibi araçlarla doğrulama sağlamak.	Bijwe ve Shankar (2022)
Yazılım Yönetimi ve Entegrasyon	Yazılım yönetiminde teknolojik uyumsuzluklar veya projede gecikmeler yaşanabilir.	Çevik yöntemler kullanarak sürekli geri bildirim döngüleri oluşturmak. Mikroservis mimarisi ile entegrasyonu kolaylaştırmak.	Khan ve Shameem (2020), Faustino ve ark. (2022)
Dokümantasyon ve İzlenebilirlik	Gereksinimlerin izlenebilirlik raporlarının eksikliği veya düzensiz olması, projelerde kontrol zorluklarına yol açar.	Reqtify veya DOORS gibi araçlar kullanarak gereksinimlerin izlenebilirlik raporlarını otomatik olarak üretmek.	Hamza, Sharifah ve Lee (2023), El Aouni ve ark. (2024)
Değişiklik Yönetimi	Değişikliklerin doğru bir şekilde yönetilmemesi, projenin hedeflerinden sapmalara neden olabilir.	Ortak bir geliştirme ve izleme ortamı kurmak. Değişiklikleri belgelemek ve izlenebilir hale getirmek.	Gashaw ve Beshah (2022), Karunarathne, Wijayanayake ve Prasadika (2024)

5.2.5. Risk yönetimi tematik analizi

DevOps proje yönetim çerçevesini uygularken risk yönetimini nasıl gerçekleştiriyorsunuz ve çeşitli riskleri (proje yönetimi riskleri, teknik riskler, kurumsal riskler ve dış kaynaklı riskler) nasıl yönetiyorsunuz, kullandığınız yöntem, teknik ve araçlar nelerdir? (Risklerin Belirlenmesi ve Tanımlanması, Risklerin Analizi ve Değerlendirilmesi, Risklerin Önceliklendirilmesi, Risklerin Planlaması ve Önlemler, Risklerin İzlenmesi ve Kontrol Edilmesi)

Amaç: Risk yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.

Tablo 5.20, yazılım projelerinde risk yönetimi süreçlerine ait kodlama frekanslarını, katılımcı oranlarını ve kelime oranlarını göstermektedir. Kodlar, risk yönetiminde hangi alanların daha fazla vurgulandığını ve önceliklendirilmesi gerektiğini belirlemek için önemli bir referans sağlamaktadır.

Tablo 5.20. Soru 5'e Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Risklerin Belirlenmesi ve Tanımlanması	12	25.50%	12	70.60%	543	29.20%
Risklerin Analizi ve Değerlendirilmesi	5	10.60%	5	29.40%	209	11.20%
Risklerin Önceliklendirilmesi	4	8.50%	4	23.50%	179	9.60%
Risklerin Planlanması ve Önlemler	8	17.00%	8	47.10%	276	14.80%
Risklerin İzlenmesi ve Kontrol Edilmesi	8	17.00%	7	41.20%	354	19.00%
Teknik ve Güvenlik Önlemleri	7	14.90%	7	41.20%	211	11.40%
Proje Yönetimi Riskleri	3	6.40%	3	17.60%	75	4.00%

Tablo 5.20 Analizi:

1. Risklerin Belirlenmesi ve Tanımlanması:

- Kod Sayısı: 12 (%25.50).
- Katılımcı Oranı: %70.60.
- Kelime Oranı: %29.20.

Bu kod, en yüksek frekans ve kelime oranına sahiptir. Bu da risklerin doğru bir şekilde belirlenmesinin risk yönetimi süreçlerinin temel taşını oluşturduğunu göstermektedir.

2. Risklerin Planlanması ve Önlemler:

- Kod Sayısı: 8 (%17.00).
- Katılımcı Oranı: %47.10.
- Kelime Oranı: %14.80.

Risklerin etkilerini azaltmak veya ortadan kaldırmak için alınan önlemler ve planlama süreçleri, sıkça ifade edilen bir diğer önemli konudur.

3. Risklerin İzlenmesi ve Kontrol Edilmesi:

- Kod Sayısı: 8 (%17.00).
- Katılımcı Oranı: %41.20.
- Kelime Oranı: %19.00.

Risklerin sürekli olarak izlenmesi ve kontrol altında tutulması, projelerde risk yönetiminin etkinliğini artıran kritik bir unsurdur.

4. Teknik ve Güvenlik Önlemleri:

- Kod Sayısı: 7 (%14.90).
- Katılımcı Oranı: %41.20.
- Kelime Oranı: %11.40.

Teknik güvenlik önlemleri, projelerin teknik altyapısını ve güvenliğini sağlamada önemli bir yer tutmaktadır.

5. Risklerin Analizi ve Değerlendirilmesi:

- Kod Sayısı: 5 (%10.60).
- Katılımcı Oranı: %29.40.

- Kelime Oranı: %11.20.

Risklerin olasılık ve etkilerinin değerlendirilmesi, risk yönetimi sürecinde önemli bir aşama olarak öne çıkmıştır.

6. Risklerin Önceliklendirilmesi:

- Kod Sayısı: 4 (%8.50).
- Katılımcı Oranı: %23.50.
- Kelime Oranı: %9.60.

Kritik risklere odaklanmak ve önceliklendirme yapmak, diğer zorluklara kıyasla daha az dile getirilmiştir.

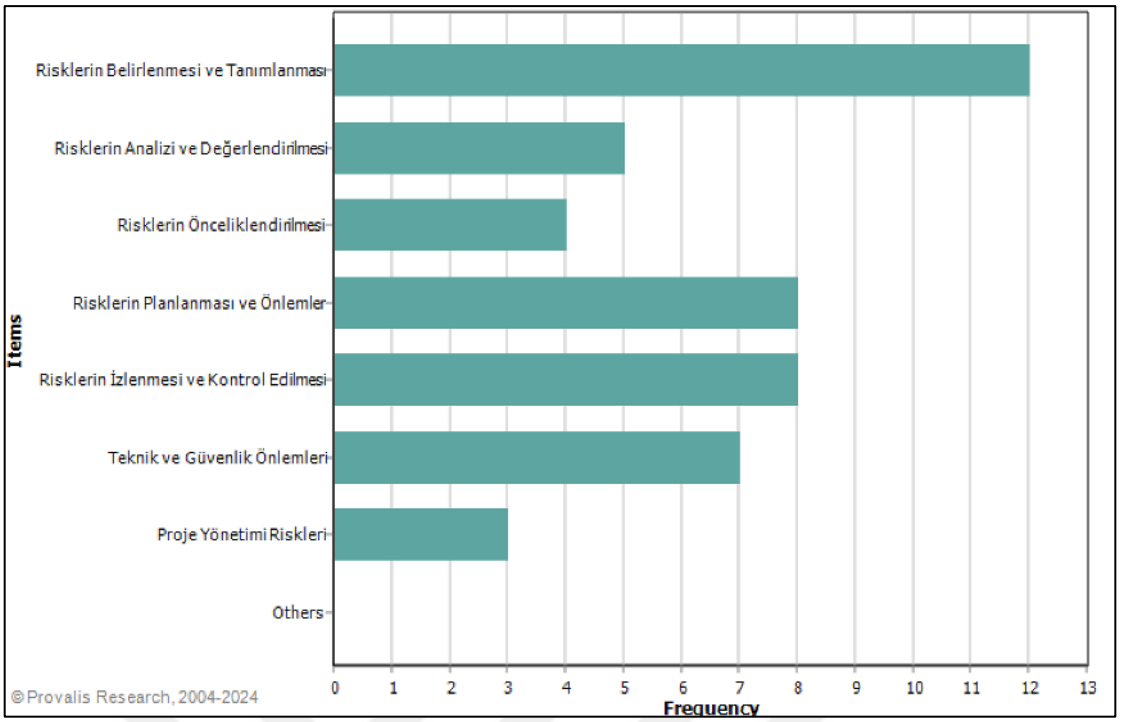
7. Proje Yönetimi Riskleri:

- Kod Sayısı: 3 (%6.40).
- Katılımcı Oranı: %17.60.
- Kelime Oranı: %4.00.

Proje zamanlaması, bütçe ve kaynaklarla ilgili riskler, görece daha az vurgulanan ancak dikkate alınması gereken bir diğer alandır.

Risklerin Belirlenmesi ve Tanımlanması, risk yönetiminin en kritik aşaması olarak öne çıkmaktadır. Risklerin Planlanması ve Önlemler ve Risklerin İzlenmesi ve Kontrol Edilmesi, süreçlerin etkin bir şekilde yönetilmesi için önemli alanlardır. Daha düşük frekansa sahip olan kodlar ise, uzun vadede ele alınması gereken ancak görece daha az yaygın olan zorlukları ifade etmektedir.

Bu bulgular, yazılım projelerinde risk yönetimi süreçlerini geliştirmek için odaklanılması gereken alanları net bir şekilde göstermektedir.



Şekil 5.9. QDA Miner Lite Soru 5'e Ait Kod Dağılımı Grafiği

Şekil 5.9'daki grafik, yazılım projelerinde risk yönetimi süreçlerinde hangi alanlara öncelik verilmesi gerektiğini net bir şekilde göstermektedir. Risklerin Belirlenmesi ve Tanımlanması, açık ara en önemli süreç olarak öne çıkarken, Risklerin Planlanması ve Önlemler ile Risklerin İzlenmesi ve Kontrol Edilmesi de süreçlerin başarılı bir şekilde yönetilmesi için kritik öneme sahiptir. Daha düşük frekansa sahip olan temalar ise, uzun vadede ele alınması gereken ancak görece daha az yaygın olan zorlukları temsil etmektedir. Bu bulgular, yazılım projelerinde risk yönetimini geliştirmek için yol gösterici bir rehber niteliğindedir.

Şekil 5.10. QDA Miner Lite Soru 5'e Ait Kod Bulutu Görseli

Şekil 5.10'daki kod bulutu, risk yönetimi süreçlerinde hangi alanların daha fazla ön plana çıktığını ve hangi konuların daha fazla ilgi gerektirdiğini net bir şekilde göstermektedir. Risklerin Belirlenmesi ve Tanımlanması, en önemli alan olarak öne çıkarken, Risklerin Planlanması ve Önlemler ve Risklerin İzlenmesi ve Kontrol Edilmesi

gibi süreçler, etkin bir risk yönetimi için temel unsurlar arasında yer almaktadır. Daha küçük boyutlu kodlar ise, uzun vadede ele alınması gereken ancak daha az yaygın olan zorlukları temsil etmektedir. Bu görsel, yazılım projelerinde risk yönetimini geliştirmek için odaklanılması gereken kritik alanları özetlemektedir.

Tablo 5.21, yazılım projelerinde risk yönetimi süreçlerinde karşılaşılan temel sorunların çözümüne yönelik kapsamlı bir yaklaşım sunmaktadır. Risklerin Belirlenmesi ve Tanımlanması, Risklerin Analizi ve Değerlendirilmesi ve Risklerin Önceliklendirilmesi gibi temalar, risk yönetimi süreçlerinde kritik öneme sahiptir. Ayrıca, teknik güvenlik önlemleri ve proje yönetimi risklerine yönelik öneriler, projelerin güvenliğini ve başarısını artırmak için somut çözümler sağlamaktadır. Kaynaklara dayalı bu öneriler, risk yönetimi stratejilerinin etkili bir şekilde uygulanmasına rehberlik etmektedir.

Tablo 5.21. Soru 5'e Ait Tematik Analiz

Tema	**Açıklama**	**Önerilen Çözümler ve Araçlar**	**Kaynak**
Risklerin Belirlenmesi ve Tanımlanması	Projeyi etkileyebilecek risklerin beyin fırtınası, geçmiş proje verileri ve SWOT analizi gibi yöntemlerle belirlenmesi.	Jira, Trello gibi proje yönetim araçları ve Confluence gibi dokümantasyon araçları ile risklerin kaydedilmesi.	Jha ve ark. (2023), Gokarna ve Singh (2021)
Risklerin Analizi ve Değerlendirilmesi	Risklerin olasılık ve etkisine göre kalitatif ve kantitatif analizlerle değerlendirilmesi. Risk matrisi kullanılması.	Kalitatif analiz için risk puanlama, kantitatif analiz için Monte Carlo simülasyonu ve risk matrisi.	Giamattei ve ark. (2023)
Risklerin Önceliklendirilmesi	Kritik risklerin MoSCoW ve Pareto gibi yöntemlerle önceliklendirilmesi ve risk puantajlaması yapılması.	MoSCoW yöntemi, Pareto analizi ve risk puantajlama kullanarak en kritik risklere odaklanma.	Hrusto, Engström ve Runeson (2023)
Risklerin Planlanması ve Önlemler	Kaçınma, azaltma, kabul ve aktarma stratejileri ile risklere karşı planlar oluşturulması.	Kaçınma, azaltma, kabul ve aktarma stratejilerinin izlenmesi için risk aksiyon planları ve Jira Roadmaps.	Fawzy, Wassif ve Moussa (2023)
Risklerin İzlenmesi ve Kontrol Edilmesi	Düzenli değerlendirme toplantıları, performans göstergeleri ve izleme araçları ile risklerin kontrol altında tutulması.	Düzenli risk değerlendirme toplantıları ve risk performans göstergeleri ile risk takibi. Grafana, Prometheus ile izleme.	Krey (2022), Jayakody ve Wijayanayake (2023)
Teknik ve Güvenlik Önlemleri	Sunucu güvenliği, şifreleme, yetkilendirme, CI/CD süreçleri ve otomatik testlerle teknik risklerin önlenmesi.	Zabbix, Nagios gibi monitoring araçları; SSL/TLS, şifreleme, CI/CD pipeline'lar ile güvenlik önlemleri.	Jayakody ve Wijayanayake (2021), Khattak ve ark. (2023)
Proje Yönetimi Riskleri	Proje zamanlaması, bütçe ve kaynaklarla ilgili risklerin çevik yöntemlerle ve planlama araçlarıyla yönetilmesi.	Jira ve Trello gibi proje yönetim araçları ile risk planlaması ve izleme. Agile yöntemlerle geri bildirim sağlanması.	Jayakody ve Wijayanayake (2022)

5.2.6. Kalite yönetimi tematik analizi

DevOps proje yönetim çerçevesini uygularken ürün ve proje süreçlerinin kalite yönetimini nasıl gerçekleştiriyorsunuz, kullandığınız yöntem, teknik ve araçlar nelerdir?

Amaç: Kalite yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.

Tablo 5.22, yazılım projelerinde kalite yönetimi süreçlerine ilişkin kodlama frekanslarını, katılımcı oranlarını ve kelime oranlarını göstermektedir. Kodlar, kalite yönetimi süreçlerinde hangi konuların daha sık vurgulandığını ve hangi alanların öncelikli olduğunu anlamak için önemli bir rehber sunmaktadır.

Tablo 5.22. Soru 6'ya Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Kalite Planlaması	4	9.10%	4	23.50%	159	10.00%
Kod Kalitesi Yönetimi	9	20.50%	8	47.10%	345	21.60%
Test Yönetimi	10	22.70%	9	52.90%	295	18.50%
Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD)	10	22.70%	9	52.90%	369	23.10%
Performans ve Güvenlik Testleri	5	11.40%	5	29.40%	171	10.70%
Geri Bildirim ve Sürekli İyileştirme	6	13.60%	6	35.30%	230	14.40%

Tablo 5.22 Analizi:

1. Kalite Planlaması:

- Kod Sayısı: 4 (%9.10).
- Katılımcı Oranı: %23.50.
- Kelime Oranı: %10.00.

Kalite planlaması, diğer kodlara kıyasla daha az dile getirilmiş olmasına rağmen, projelerin başlangıcında kalite hedeflerinin netleştirilmesi açısından önemli bir konudur.

2. Kod Kalitesi Yönetimi:

- Kod Sayısı: 9 (%20.50).
- Katılımcı Oranı: %47.10.
- Kelime Oranı: %21.60.

Kod kalitesini artırmak ve sürdürülebilirliği sağlamak, yazılım projelerinin uzun vadeli başarısı için önemli bir süreç olarak öne çıkmaktadır.

3. Test Yönetimi:

- Kod Sayısı: 10 (%22.70).
- Katılımcı Oranı: %52.90.
- Kelime Oranı: %18.50.

Test yönetimi, yazılımın her aşamasında kaliteyi artırmak için sıkça dile getirilen bir süreçtir. Manuel testlerin azaltılması ve otomatik test süreçlerinin entegrasyonu, bu alandaki temel önerilerdendir.

4. Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD):

- Kod Sayısı: 10 (%22.70).
- Katılımcı Oranı: %52.90.
- Kelime Oranı: %23.10.

CI/CD süreçlerinin otomasyonu, kod değişikliklerinin entegrasyonu ve dağıtımını sağlamak açısından çok sık vurgulanan bir konudur.

5. Performans ve Güvenlik Testleri:

- Kod Sayısı: 5 (%11.40).
- Katılımcı Oranı: %29.40.
- Kelime Oranı: %10.70.

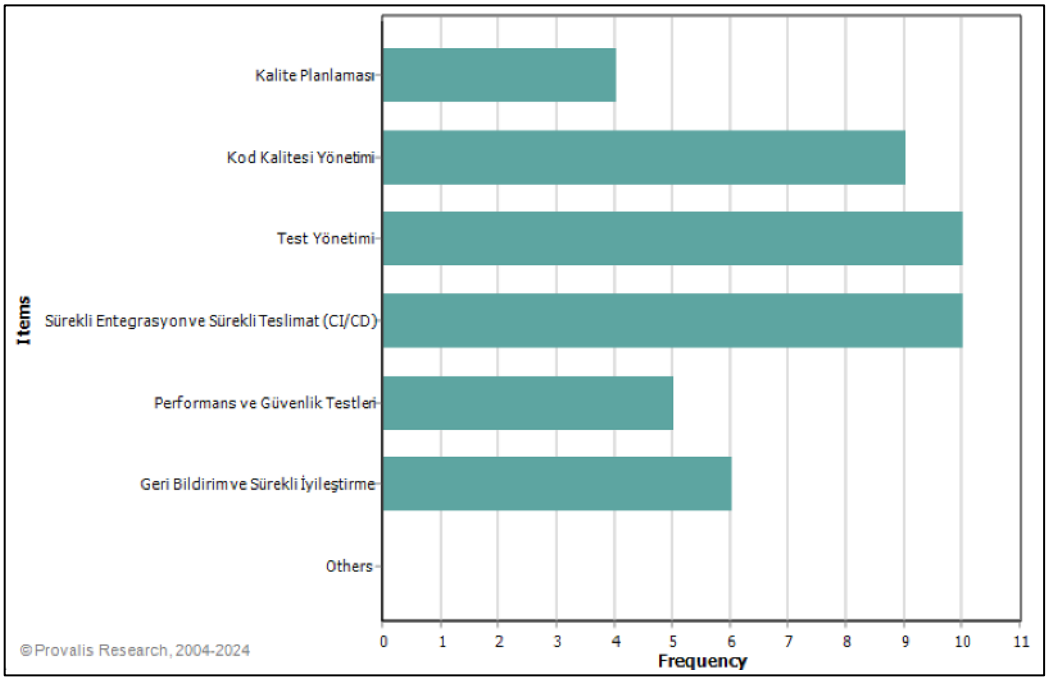
Yazılımın performansını ve güvenliğini artırmaya yönelik test süreçleri, projelerin güvenilirliğini sağlamak için önemlidir.

6. Geri Bildirim ve Sürekli İyileştirme:

- Kod Sayısı: 6 (%13.60).
- Katılımcı Oranı: %35.30.
- Kelime Oranı: %14.40.

Kullanıcı geri bildirimlerinin toplanması ve retrospektif süreçlerin düzenlenmesi, kalite yönetiminde sürekli iyileştirme için önemli bir adımdır.

Test Yönetimi ve Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD), kalite yönetimi süreçlerinde en sık dile getirilen alanlar olarak öne çıkmaktadır. Kod Kalitesi Yönetimi ve Geri Bildirim ve Sürekli İyileştirme, kaliteyi artırma ve sürdürme süreçlerinde önemli bir role sahiptir. Kalite Planlaması ve Performans ve Güvenlik Testleri ise, görece daha az vurgulanmakla birlikte, yazılım projelerinin güvenilirliği ve başarısı için kritik öneme sahiptir. Bu bulgular, kalite yönetimi stratejilerinin geliştirilmesi ve uygulanması için yol gösterici bir çerçeve sunmaktadır.



Şekil 5.11. QDA Miner Lite Soru 6'ya Ait Kod Dağılımı Grafiği

Şekil 5.11'deki grafik, kalite yönetimi süreçlerinde odaklanılması gereken temel alanları açıkça göstermektedir. Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD) ve Test Yönetimi, kalite yönetimi stratejilerinde en öncelikli alanlar olarak öne çıkmaktadır. Kod Kalitesi Yönetimi ve Geri Bildirim ve Sürekli İyileştirme, kaliteyi artırmada önemli bir role sahiptir. Performans ve Güvenlik Testleri ile Kalite Planlaması ise, diğer alanlara kıyasla daha az vurgulansa da yazılım projelerinin başarısı için kritik öneme sahiptir. Bu bulgular, projelerde kalite yönetimi süreçlerinin daha etkili bir şekilde uygulanması için yol gösterici bir çerçeve sunmaktadır.

Şekil 5.12. QDA Miner Lite Soru 6'ya Ait Kod Bulutu Görseli

Şekil 5.12'deki kod bulutu, yazılım projelerinde kalite yönetimi stratejilerinde odaklanılması gereken temel alanları net bir şekilde ortaya koymaktadır. Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD) ve Test Yönetimi, projelerde kaliteyi artırmak için en sık vurgulanan süreçlerdir. Kod Kalitesi Yönetimi ve Geri Bildirim ve Sürekli İyileştirme, kalite yönetiminde önemli bir yere sahiptir. Daha küçük boyutlu kodlar ise, projelerin spesifik ihtiyaçlarına bağlı olarak ele alınması gereken konuları temsil etmektedir. Bu görselin, kalite yönetimi süreçlerinin iyileştirilmesi için öncelikli alanları belirlemede rehberlik edebileceği değerlendirilmektedir.

Tablo 5.23, yazılım projelerinde kalite yönetimi süreçlerini iyileştirmek için gerekli olan araçlar ve yaklaşımları ortaya koymaktadır. Kalite Planlaması ve Kod Kalitesi Yönetimi, proje başlangıcında kaliteyi sağlamada önemli rol oynarken, Test Yönetimi ve Sürekli Entegrasyon ve Teslimat süreçleri, kaliteyi sürdürmek ve iyileştirmek için kritik öneme sahiptir. Performans ve Güvenlik Testleri ile Geri Bildirim ve Sürekli İyileştirme ise yazılımın sürekli gelişimini sağlamak için gereklidir. Bu öneriler, projelerin kalite hedeflerine ulaşmasına yardımcı olacak somut adımları temsil etmektedir.

Tablo 5.23. Soru 6'ya Ait Tematik Analiz

Tema	Açıklama	Önerilen Çözümler ve Araçlar	Kaynak
Kalite Planlaması	Projeye özgü kalite hedeflerinin belirlenmesi ve kalite standartlarının tanımlanması.	Kalite hedefleri belirleme, paydaş iletişimi. Confluence, Google Docs. Kalite hedeflerinin belirlenmesi sürecine tüm paydaşların dahil edilmesi.	Battina (2019)
Kod Kalitesi Yönetimi	Yazılım kodunun kalitesini artırmak ve sürdürülebilirliğini sağlamak.	Kod gözden geçirme, kod standartları belirleme. SonarQube. Kod gözden geçirme süreçlerinin belirlenmesi ve SonarQube gibi araçların etkin kullanılması.	Faustino ve ark. (2022)
Test Yönetimi	Yazılımın her aşamasında test süreçlerini entegre ederek kaliteyi artırmak.	Otomatik testler, Test Tabanlı Geliştirme (TDD). JUnit, Selenium. Otomatik test senaryoları oluşturularak manuel testlerin azaltılması öneriliyor.	Bolscher ve Daneva (2019)
Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD)	Kod değişikliklerinin sürekli entegrasyonunu ve dağıtımını sağlamak.	Otomatik build ve test süreçleri, Rollback planları. Jenkins, GitHub Actions, GitLab CI, Docker, Kubernetes. Rollback planlarının belirlenmesi ve CI/CD süreçlerinin otomatize edilmesi.	Battina (2021)
Performans ve Güvenlik Testleri	Yazılımın performansını ve güvenliğini artırmak.	Yük testleri, güvenlik testleri. JMeter, Amazon CloudWatch. Yük ve güvenlik testlerinin otomatik hale getirilmesi öneriliyor.	Amaro, Pereira ve da Silva (2024)
Geri Bildirim ve Sürekli İyileştirme	Kullanıcı geri bildirimlerini toplamak ve süreçleri geliştirmek.	Kullanıcı testleri, sprint retroları. Jira, Confluence, Slack. Geri bildirim mekanizmalarının düzenli hale getirilmesi ve retrospektiflerin yapılması.	Akbar, Smolander ve ark. (2022)

5.2.7. Güçlükler tematik analizi

DevOps proje yönetim süreçlerinin, araçlarının ve ilkelerinin mevcut kurumsal proje yönetimi yaklaşımlarına, teknik alt yapıya, proje takıma entegrasyonunda karşılaştığınız güçlükler, kullandığınız yöntem ve araçlar ile çözüm önerileriniz nelerdir?

Amaç: DevOps proje yönetim süreçlerinin entegrasyonuyla ilgili sorunları ve çözüm önerilerini belirlemektir.

Tablo 5.24, DevOps adaptasyon sürecinde karşılaşılan zorlukların kod frekanslarını, katılımcı oranlarını ve kullanılan kelime sayılarını özetlemektedir. Tablonun, hangi zorlukların daha sık karşılaşıldığını ve bunların detaylı olarak ne kadar tartışıldığını görselleştirmek için faydalı bir araç olduğu değerlendirilmektedir.

Tablo 5.24. Soru 7'ye Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Kültür ve Direnç	13	33.30%	10	58.80%	415	27.10%
Araç ve Süreç Uyuşmazlığı	14	35.90%	12	70.60%	578	37.70%
Süreç İyileştirmeleri ve Entegrasyon	4	10.30%	3	17.60%	159	10.40%
İletişim ve İşbirliği Sorunları	8	20.50%	7	41.20%	288	18.80%

Tablo 5.24 Analizi:

1. Kültür ve Direnç:

- Sayı: 13 (33.30%).
- Katılımcı Oranı: %58.80 (10 kişi).
- Kelime Oranı: %27.10 (415 kelime).

Kültürel direnç, DevOps adaptasyon sürecinde en sık karşılaşılan zorluklardan biridir. Çalışanların mevcut çalışma kültüründen uzaklaşıp yeni yöntemlere adapte olması zaman alabilir.

2. Araç ve Süreç Uyumsuzluğu:

Sayı: 14 (35.90%).
Katılımcı Oranı: %70.60 (12 kişi).
Kelime Oranı: %37.70 (578 kelime).

Araç ve süreç uyumsuzluğu, en yüksek katılımcı oranına ve kelime kullanımına sahip başlıktır. Bu durum, teknolojik entegrasyonun zorluklarına işaret etmektedir. Kurumsal araçların DevOps ile uyumlu hale getirilmesi gerektiği sıkça vurgulanmıştır.

3. Süreç İyileştirmeleri ve Entegrasyon:

Sayı: 4 (10.30%).
Katılımcı Oranı: %17.60 (3 kişi).
Kelime Oranı: %10.40 (159 kelime).

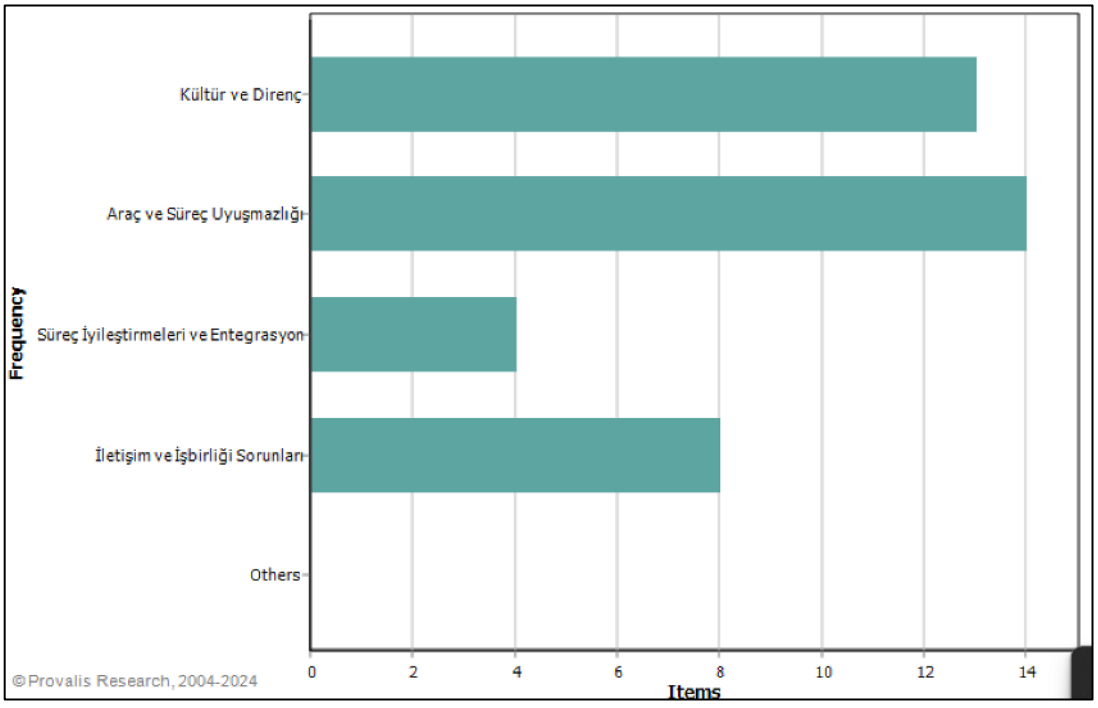
Süreç iyileştirmeleri ve entegrasyon, görece daha az tartışılan ancak önemli bir konudur. Mevcut süreçlerin optimize edilmesi ve DevOps prensiplerine uygun hale getirilmesi gerektiği belirtilmiştir.

4. İletişim ve İşbirliği Sorunları:

Sayı: 8 (20.50%).
Katılımcı Oranı: %41.20 (7 kişi).
Kelime Oranı: %18.80 (288 kelime).

İletişim ve işbirliği sorunları, ekipler arasındaki koordinasyon eksikliklerini temsil etmektedir. Slack, Microsoft Teams gibi araçlarla bu sorunların giderilebileceği belirtilmiştir.

Araç ve Süreç Uyumsuzluğu ve Kültür ve Direnç, adaptasyon sürecinde en sık karşılaşılan problemler olarak öne çıkmaktadır. Süreç İyileştirmeleri ve Entegrasyon ise, çözüm geliştirme süreçlerinin önemli bir parçasıdır ancak daha az sıklıkla dile getirilmiştir. İletişim ve İşbirliği Sorunları, ekipler arası etkileşim ve uyumun artırılması gereken bir alan olarak dikkat çekmektedir. Bu veriler, DevOps adaptasyon sürecini geliştirmek için odaklanılması gereken kritik noktaları ortaya koymaktadır.



Şekil 5.13. QDA Miner Lite Soru 7'ye Ait Kod Dağılımı Grafiği

Şekil 5.13'de yer alan grafikte, Araç ve Süreç Uyumsuzluğu en sık dile getirilen zorluk olarak öne çıkarken, Kültür ve Direnç ikinci sırada yer almaktadır. İletişim ve İşbirliği Sorunları, ekiplerin koordinasyonunu artırmayı hedefleyen çözümleri gerekli kılarken, Süreç İyileştirmeleri ve Entegrasyon ise adaptasyon sürecinde daha az vurgulanan ancak kritik bir alan olarak göze çarpmaktadır.

Bu grafiğin, DevOps'a geçiş süreçlerini daha etkin planlamak ve bu zorluklara yönelik çözüm geliştirmek için kullanılabileceği değerlendirilmektedir.

Şekil 5.14. QDA Miner Lite Soru 7'ye Ait Kod Bulutu Görseli

Şekil 5.14'de yer alan kod bulutu görseli, DevOps adaptasyonu sırasında dikkate alınması gereken temel zorlukları hızlı bir şekilde kavramaya olanak tanımaktadır. Görseldeki yoğunluk, araçların uyumsuzluğu ve kültürel direnç konularının en kritik zorluklar olduğunu vurgulamaktadır. Bu bulut, araştırmacıların ve karar alıcıların bu alanlara odaklanarak çözüm geliştirmelerini desteklemektedir.

Tablo 5.25, DevOps adaptasyon sürecinde karşılaşılan temel zorlukları dört ana başlık altında toplamaktadır. Kültürel Değişim ve Direnç, çalışanların yeni süreçlere alışması için eğitim ve farkındalık çalışmalarını önerirken, Araç ve Süreç Uyumsuzluğu, teknolojik entegrasyonun önemini vurgulamaktadır. Süreç İyileştirmeleri ve Entegrasyon, süreçlerin optimize edilmesi gerektiğine dikkat çekerken, İletişim ve İşbirliği Sorunları, ekipler arası etkileşimi artırmak için kullanılabilecek araçlara odaklanmaktadır. Bu öneriler, DevOps süreçlerinin daha etkili bir şekilde uygulanmasına rehberlik etmektedir.

Tablo 5.25. Soru 7'ye Ait Tematik Analiz

Tema	Açıklama	Önerilen Çözümler ve Araçlar	Kaynak
Kültürel Değişim ve Direnç	DevOps'un hızlı ve çevik yapısı, geleneksel proje yönetim kültürüyle uyumsuz olabiliyor. Çalışanların alışık oldukları geleneksel yöntemlerden uzaklaşmaları ve yeni süreçlere adapte olmaları zorlaşıyor.	DevOps kültürünün benimsenmesi için organizasyon genelinde eğitim programları, farkındalık çalışmaları ve atölye çalışmaları düzenlenmesi öneriliyor. Bu programlar sayesinde eski ve yeni yöntemler arasındaki farklılıklar açıklanarak ekiplerin adaptasyonu hızlandırılıyor.	Fawzy, Wassif ve Moussa (2023), Chiari ve ark. (2023)
Araç ve Süreç Uyuşmazlığı	Mevcut kurumsal proje yönetimi araçları, DevOps araçlarıyla tam uyumlu olmayabiliyor. Geleneksel proje yönetimi araçları, çevik ve sürekli teslimat süreçlerini desteklemekte yetersiz kalabiliyor.	Araç entegrasyonlarına odaklanarak hibrit bir araç seti oluşturulması gerektiği belirtiliyor. Jira, Trello gibi geleneksel proje yönetimi araçlarının yanı sıra CI/CD araçlarının (Jenkins, GitLab CI) entegrasyonu öneriliyor.	Alnafessah ve ark. (2021)
Süreç İyileştirmeleri ve Entegrasyon	DevOps süreçleri ile mevcut kurumsal süreçlerin uyumlu hale getirilmesi zaman alabiliyor ve bazı zorluklar çıkarabiliyor. Teknik altyapı uyumsuzlukları da süreçleri zorlaştırıyor.	Süreçlerin kademeli olarak DevOps prensiplerine uyumlu hale getirilmesi, sürekli geri bildirim döngüleriyle iyileştirilmesi ve altyapı uyum problemlerinin çözümü için otomasyon sistemlerinin artırılması öneriliyor.	Eramo, Tucci ve ark. (2024)
İletişim ve İşbirliği Sorunları	DevOps ekipleri, geleneksel ekip yapılarının dışında çalıştıkları için, ekipler arası iletişimde kopukluklar yaşanabiliyor. Kapalı ağlar gibi özel çalışma şartları altında çalışan ekipler, ekstra zorluklarla karşılaşabiliyor.	Etkili iletişim ve işbirliği sağlamak için Slack, Microsoft Teams gibi araçlar öneriliyor. Ayrıca, Jira ve Confluence gibi işbirliği araçlarıyla dokümantasyonun yönetilmesi önem kazanıyor.	Gashaw ve Beshah (2022), Hrusto, Engström ve Runeson (2023)

5.2.8. Yapay zekâ tematik analizi

DevOps proje yönetim çerçevesinin yapay zekâ yazılımlarının (makine öğrenmesi, derin öğrenme vb.) geliştirilmesine yönelik görüşleriniz nelerdir?

Amaç: DevOps proje yönetim çerçevesinin yapay zekâ yazımlarının geliştirilmesine yönelik görüşlerin belirlenmesi

Tablo 5.26’daki tematik analiz, yapay zekanın DevOps süreçlerindeki uygulamalarını ve faydalarını vurgulamaktadır. Ayrıca, bu analiz, geliştirici ekipler ve yöneticilerin, süreçlerini daha etkili bir şekilde yönetmeleri için rehber niteliği taşımaktadır.

Tablo 5.26. Soru 8'e Ait Tematik Analiz

Tema	Açıklama	Kaynak
Yapay Zeka ve Otomasyon	Yapay zekanın DevOps süreçlerinde otomasyon, veri analitiği ve sorun tespiti konularında katkı sağladığı ifade ediliyor. Tekrarlayan görevlerin otomatikleştirilmesi ile DevOps ekibinin stratejik işlere odaklanma imkanı artıyor.	Jayakody ve Wijayanayake (2023)
Veri Yönetimi ve İzleme	Yapay zekanın veri yönetim süreçlerini optimize ederek veri kalitesini artırdığı ve model performansını sürekli izlediği belirtiliyor.	Cuadra ve ark. (2024)
Model Geliştirme ve CI/CD	DevOps çerçevesinin CI/CD süreçlerini kullanarak makine öğrenmesi modellerini daha hızlı güncellediği ve yeni veri entegrasyonu sağladığı ifade ediliyor. Model eğitim ve dağıtım süreçlerinde CI/CD süreçlerinin entegrasyonunun önemine vurgu yapılıyor.	Antunes ve Tate (2024)
Performans ve Güvenlik	Yapay zekanın performans ölçümü ve geliştiricilerin verimliliğini artırma potansiyelinin olduğu vurgulanıyor. Geliştirici araçları ile entegre edilen yapay zeka destekli sistemlerin hata çözümü ve performans iyileştirme önerileri sağladığı belirtiliyor.	Díaz ve ark. (2024)
Araçlar ve Teknikler	Belirli araçlar ve teknikler (Jenkins, GitLab CI, MLflow vb.) DevOps süreçlerinde yapay zekanın entegrasyonu için kullanılıyor. Veri ve model süreçlerinin yönetimi için özel araçların (DVC, Prometheus, Kubeflow vb.) önerildiği ifade ediliyor.	Amaro, Pereira ve da Silva (2022)

Tablo 5.27, DevOps süreçlerinde yapay zeka uygulamalarını daha iyi anlamak ve geliştirmek için odak alanlarını belirlemeye yardımcı olmaktadır.

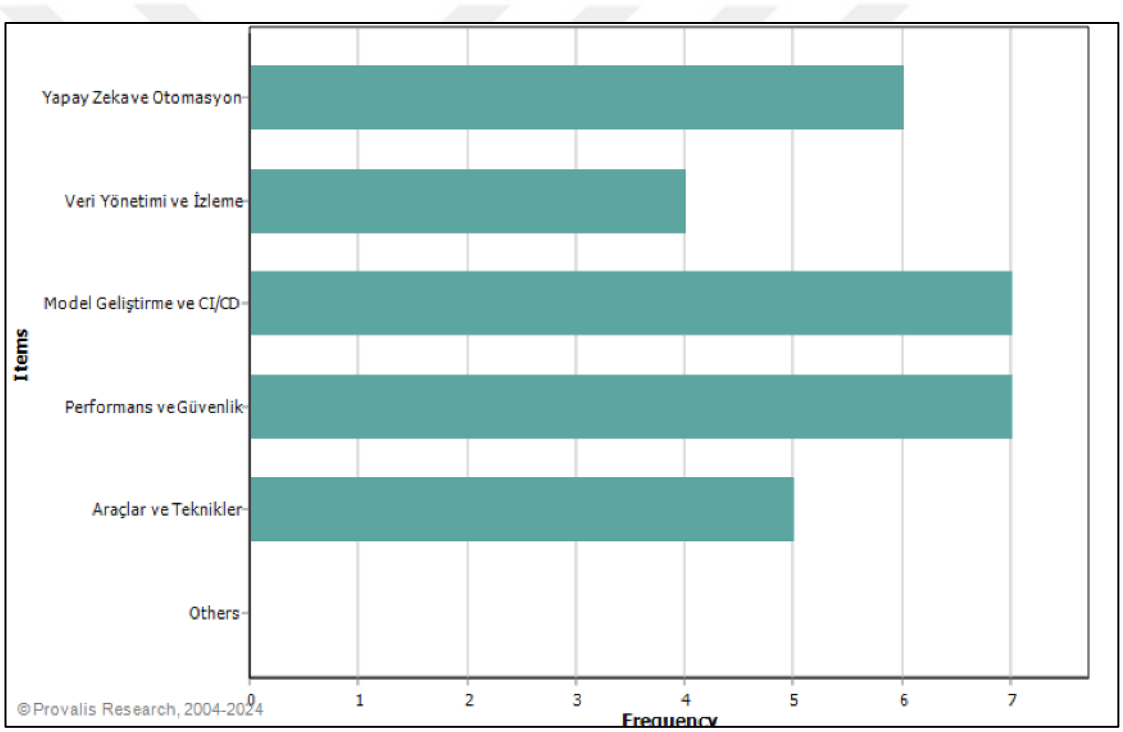
En yüksek frekans, "Model Geliştirme ve CI/CD" ve "Performans ve Güvenlik" başlıklarında gözlenmiştir (%24.10). Bu, katılımcıların yapay zeka destekli sistemlerde model geliştirme ve CI/CD entegrasyonunu, güvenlik ve performans optimizasyonu kadar önemli gördüğünü göstermektedir.

Kelime Oranı açısından, "Yapay Zeka ve Otomasyon" en yüksek oranla %29.90'dır. Bu, katılımcıların yapay zeka otomasyonu üzerinde daha ayrıntılı görüş sunduğunu göstermektedir.

Katılımcı Oranının, "Model Geliştirme ve CI/CD", "Performans ve Güvenlik" ve "Yapay Zeka ve Otomasyon" başlıklarında eşit (%35.30) ve dikkat çekici olduğu değerlendirilmektedir.

Tablo 5.27. Soru 8'e Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Yapay Zeka ve Otomasyon	6	20.70%	6	35.30%	368	29.90%
Veri Yönetimi ve İzleme	4	13.80%	3	17.60%	172	14.00%
Model Geliştirme ve CI/CD	7	24.10%	6	35.30%	288	23.40%
Performans ve Güvenlik	7	24.10%	6	35.30%	235	19.10%
Araçlar ve Teknikler	5	17.20%	5	29.40%	266	21.60%



Şekil 5.15. QDA Miner Lite Soru 8'e Ait Kod Dağılımı Grafiği

Şekil 5.15'de yer alan grafik, yapay zekanın DevOps'ta nasıl uygulandığına dair önemli ipuçları sunmaktadır. Model geliştirme ve güvenlik öncelikli konular olarak dikkat çekerken, araçlar ve veri yönetimi gibi alanlarda daha fazla geliştirme ve odaklanma gerektiği sonucuna varılabileceği değerlendirilmektedir.

Şekil 5.16. QDA Miner Lite Soru 8'e Ait Kod Bulutu Görseli

Şekil 5.16'da yer alan kod bulutu, yapay zeka destekli DevOps süreçlerinde en çok üzerinde durulan alanları net bir şekilde yansıtmaktadır. Model geliştirme, performans ve güvenlik, uygulama süreçlerinde öncelikli olarak öne çıkan başlıklardır. Veri yönetimi gibi alanların ise daha az sıklıkla vurgulanan ancak yine de önem arz eden temalar olduğu değerlendirilmektedir.

5.2.9. Yapay zeka entegrasyonu tematik analizi

Günümüzde LLM vb. yapay zeka yazılımları yazılım geliştirme süreçlerine entegre edilmekte, çeşitli işlevlerin gerçekleştirilmesi, yazılımın geliştirilmesi vb. amaçlarla kullanılmaktadır. Bu konuyla ilgili görüşleriniz nelerdir? Gerek genel olarak ve gerekse kendi kurumunuzda kullanımlarına yönelik neler önerebilirsiniz?

Amaç: Yapay zekâ yazılımlarının yazılım mühendisliği süreçlerinde kullanımıyla ilgili sorunları ve çözüm önerilerini belirlemektir.

Tablo 5.28'de yer alan bulgular, LLM'lerin (Büyük Dil Modelleri) yazılım geliştirme süreçlerinde sağladığı faydalar ve karşılaşılan temel kullanım alanlarını detaylandırmaktadır. Bulgular, literatürden ve uzman görüşlerinden elde edilen verilerle desteklenmiş olup, yazılım geliştirme süreçlerinin farklı aşamalarında LLM'lerin etkinliğine vurgu yapmaktadır.

Tablo 5.28. Soru 9'a Ait Tematik Analiz

Tema	Açıklama	Kaynak
Hız ve Verimlilik	LLM'lerin yazılım geliştirme süreçlerine entegrasyonu, kod üretme ve hata ayıklama gibi görevleri otomatikleştirerek yazılım geliştirme hızını artırır. Kod tamamlama ve hata ayıklama ile geliştiricilere zaman kazandırma. Tekrarlayan işlerin otomatikleştirilmesiyle, geliştiricilerin daha yaratıcı görevlere odaklanabilmesi.	Karunarathne, Wijayanayake ve Prasadika (2024)
Özelleştirilebilirlik ve Öğrenme Desteği	LLM'ler, belirli alanlara veya şirketlere özgü verilerle özelleştirilebilir, böylece geliştiricilere daha spesifik çözümler sunabilir. Kuruma özel verilerle eğitilen modellerin, daha iyi rehberlik ve öneriler sağlaması. Eğitim ve gelişim için yeni diller veya teknolojilere yönelik içerikler sunması.	Firdaouss ve ark. (2022)
Yazılım Belgelerinin Otomatik Üretilmesi	LLM'ler, yazılım belgelerinin otomatik oluşturulması ve güncellenmesi konusunda önemli katkılar sağlayabilir. Kod açıklamaları, API belgeleri gibi yazılım dokümantasyonlarının güncellenmesi ve oluşturulması. Proje yönetimi ve iş takip süreçlerine yapay zeka entegrasyonu ile görev atamaları ve ilerleme raporlamalarının otomatikleştirilmesi.	Koren ve ark. (2023)
Test ve Hata Ayıklama Otomasyonu	LLM tabanlı araçlar, test süreçlerini otomatize edebilir ve hata tahmini veya çözümü konusunda yardımcı olabilir. Test senaryolarının oluşturulması, test sonuçlarının analizi gibi işlevlerin otomatikleştirilmesi. Test süreçlerinde daha hızlı ve kapsamlı bir yaklaşım geliştirilmesi.	Battina (2020)
Güvenlik ve Veri Gizliliği	LLM'lerin kullanımı sırasında veri güvenliği ve gizlilik önlemleri alınmalıdır.	Bollieddula (2022)
Proje Yönetimi ve İş Takibi	LLM tabanlı araçlar, proje yönetimi ve iş takip süreçlerinde iş yükünü azaltmak ve öngörü sağlamak için kullanılabilir. Yapay zeka tabanlı iş yönetimi araçlarının, görev atamalarını ve iş gücü tahminlerini daha isabetli yapabilmesi. Proje yöneticilerine veri analitiği ve tahminlere dayalı destek sağlanması.	Hemon-Hildgen, Rowe ve Monnier-Senicourt (2020)
Özelleştirilmiş Eğitim ve Farkındalık	LLM'ler yazılım geliştiricilerine rehberlik ederek sürekli öğrenme fırsatları sunar. Geliştiricilerin yeniliklere hızlı uyum sağlaması için eğitim ve farkındalık süreçlerine destek. Özelleştirilmiş içeriklerle teknik bilgi seviyesinin artırılması.	Hamza, Sharifah ve Lee (2023), Fernandes ve ark. (2022)

LLM'lerin hız ve verimlilik üzerindeki etkisi, yazılım geliştirme süreçlerini hızlandırarak zaman tasarrufu sağladığını ve geliştiricilerin daha yaratıcı görevlere odaklanmasına imkân tanıdığını göstermektedir. Bunun yanı sıra, özelleştirilebilirlik ve öğrenme desteği ile LLM'lerin kuruma özgü ihtiyaçlara göre adapte edilebilmesi, geliştirme süreçlerinde daha hedefe yönelik çözümler sunulmasına olanak tanımaktadır. Özellikle özelleştirilmiş eğitim ve farkındalık programları, yazılım ekiplerinin bilgi seviyesini artırarak yeni teknolojilere uyum süreçlerini kolaylaştırmaktadır.

Yazılım belgelerinin otomatik üretilmesi, test ve hata ayıklama otomasyonu gibi uygulamalar ise yazılım projelerinde rutin görevlerin otomatikleştirilmesini sağlayarak

süreçlerin etkinliğini artırmaktadır. Ancak, bu süreçlerde veri güvenliği ve gizlilik önlemlerinin alınması gerektiği de literatür tarafından vurgulanmıştır. LLM'lerin proje yönetimi ve iş takibi süreçlerinde de kullanılması, iş süreçlerinin daha öngörülebilir ve düzenli hale getirilmesini sağlamaktadır.

Sonuç olarak, LLM'ler yazılım geliştirme süreçlerine hız, verimlilik ve özelleştirilebilirlik gibi değerler katmakta, ancak bu faydaların sürdürülebilir hale getirilmesi için güvenlik ve etik önlemlerin de dikkate alınması gerekmektedir. Tablo 5.28'de yer alan bulgular, LLM'lerin potansiyel faydalarının yanı sıra kullanım esnasında göz önünde bulundurulması gereken önemli hususları da ortaya koymaktadır.

Tablo 5.29'da yer alan bulgular, LLM'lerin (Büyük Dil Modelleri) yazılım geliştirme süreçlerinde farklı odak alanlarında sağladığı faydaların ve bu faydaların katılımcılar tarafından nasıl algılandığının frekanslarını ortaya koymaktadır.

Tablo 5.29. Soru 9'a Ait Kod Frekansı

Kod	Sayı	Kod Oranı	Katılımcı	Katılımcı Oranı	Kelime Sayısı	Kelime Oranı
Hız ve Verimlilik	12	29.30%	11	64.70%	546	34.40%
Özelleştirilebilirlik ve Öğrenme Desteği	4	9.80%	2	11.80%	106	6.70%
Yazılım Belgelerinin Otomatik Üretilmesi	4	9.80%	4	23.50%	193	12.20%
Test ve Hata Ayıklama Otomasyonu	7	17.10%	7	41.20%	309	19.50%
Güvenlik ve Veri Gizliliği	5	12.20%	4	23.50%	106	6.70%
Proje Yönetimi ve İş Takibi	4	9.80%	3	17.60%	155	9.80%
Özelleştirilmiş Eğitim ve Farkındalık	5	12.20%	5	29.40%	142	8.90%

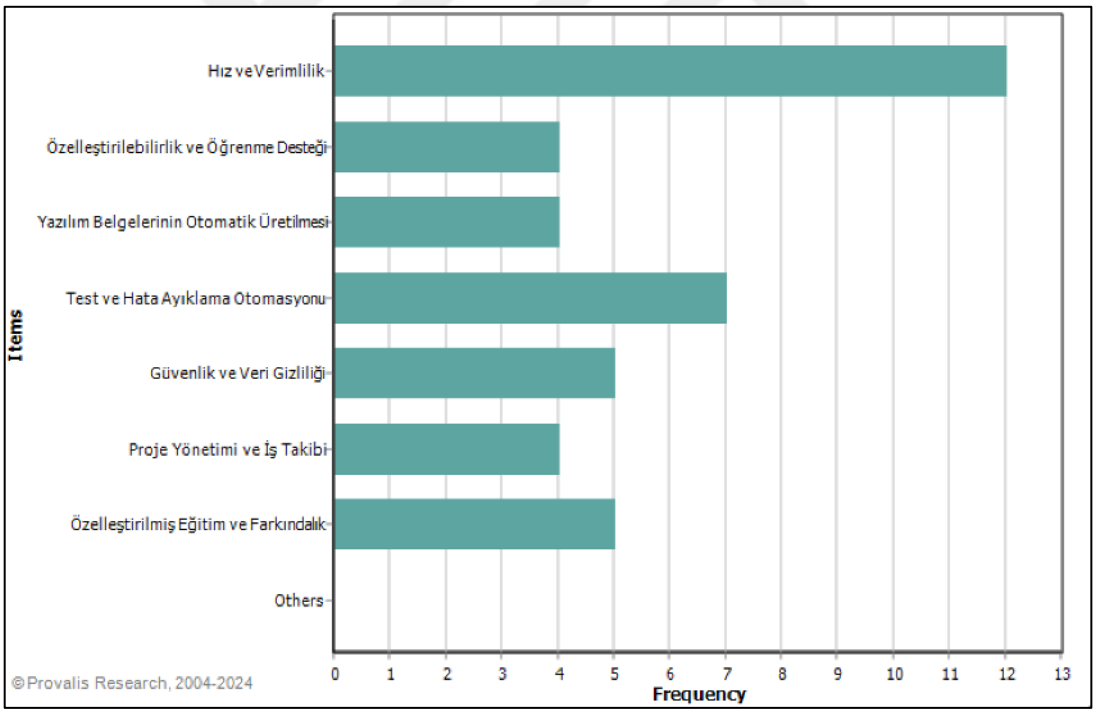
Hız ve Verimlilik en yüksek orana sahip kod olarak, LLM'lerin yazılım geliştirme süreçlerini hızlandırmada ve verimliliği artırmada önemli bir rol oynadığını göstermektedir. Katılımcıların %64,70'i bu konuda geri bildirimde bulunmuş ve toplamda %34,40'lık bir kelime oranı ile bu temanın önemi vurgulanmıştır.

Bunun yanında, Test ve Hata Ayıklama Otomasyonu da önemli bir kategori olarak öne çıkmıştır. LLM'lerin, yazılım test süreçlerini otomatikleştirme ve hataları hızlı bir şekilde

çözme yeteneği, %41,20 katılımcı oranıyla belirtilmiştir. Bu, yazılım geliştirme süreçlerinde manuel iş yükünü azaltan ve süreçleri hızlandıran bir etkisini ortaya koymaktadır.

Güvenlik ve Veri Gizliliği ile Özelleştirilmiş Eğitim ve Farkındalık kategorileri, LLM'lerin hem teknik hem de kültürel uyum süreçlerinde katkı sağladığını göstermektedir. Ancak, özellikle güvenlik ve gizlilik konularında dikkat edilmesi gereken önemli noktalar olduğu vurgulanmıştır.

Sonuç olarak, Tablo 5.29'daki veriler, LLM'lerin yazılım geliştirme süreçlerinde hız, verimlilik, otomasyon, eğitim ve güvenlik gibi temel alanlarda önemli katkılar sağladığını, ancak bu süreçlerin yönetiminde bazı hassasiyetlerin dikkate alınması gerektiğini ortaya koymaktadır. Bu bulgular, LLM'lerin yazılım projelerine entegrasyonu için hem teknik hem de stratejik bir yaklaşım gerektirdiğini göstermektedir. Risk yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.



Şekil 5.17. QDA Miner Lite Soru 9'a Ait Kod Dağılımı Grafiği

Şekil 5.17'de yer alan kod dağılımı, LLM'lerin (Büyük Dil Modelleri) yazılım geliştirme süreçlerine sağladığı katkılar ve bu katkıların kategoriler bazında nasıl

değerlendirildiğini göstermektedir. Grafik, LLM'lerin yazılım projelerindeki farklı kullanım alanlarındaki önem derecesini katılımcıların görüşlerine dayanarak sıralamaktadır.

Grafikte açıkça görüldüğü üzere, Hız ve Verimlilik kategorisi en yüksek frekansa sahiptir ve LLM'lerin yazılım geliştirme süreçlerini hızlandırmada, görevlerin daha kısa sürede tamamlanmasını sağlamada ve verimliliği artırmada önemli bir rol oynadığı katılımcılar tarafından güçlü bir şekilde ifade edilmiştir. Test ve Hata Ayıklama Otomasyonu ise ikinci sırada yer alarak, LLM'lerin yazılım test süreçlerinde ve hata tespitinde etkili bir şekilde kullanıldığını göstermektedir.

Güvenlik ve Veri Gizliliği, Özelleştirilmiş Eğitim ve Farkındalık ve Proje Yönetimi ve İş Takibi gibi kategoriler, LLM'lerin hem teknik hem de organizasyonel süreçlerde sağladığı destekleri temsil etmektedir. Özellikle, güvenlik ve gizlilik konusundaki uygulamalar, katılımcıların dikkat çektiği önemli bir alan olarak öne çıkmaktadır.

Şekil 5.17'deki dağılım, LLM'lerin yazılım geliştirme projelerinde yenilikçi bir çözüm olarak değer sağladığı ancak her kategorideki uygulamaların kapsamının dikkatle ele alınması gerektiğini vurgulamaktadır. Bu bulgular, LLM'lerin entegrasyonuna yönelik stratejik planlamaların yapılmasını desteklemektedir.

Şekil 5.18. QDA Miner Lite Soru 9'a Ait Kod Bulutu Görseli

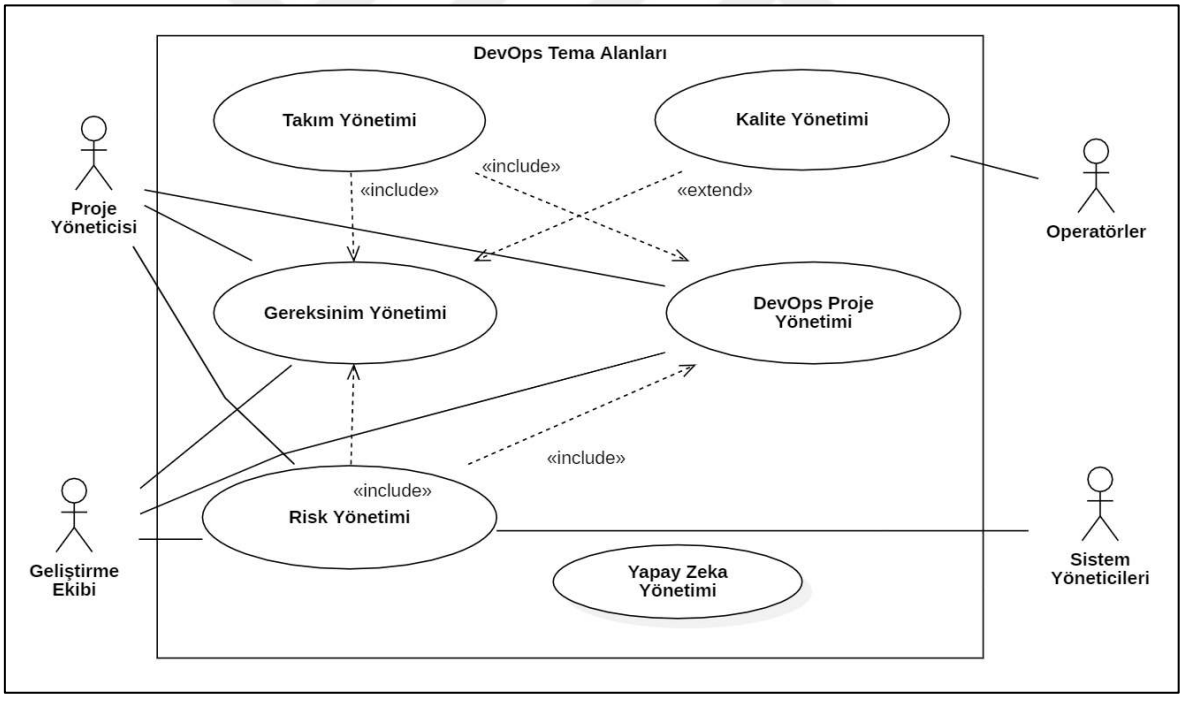
Şekil 5.18'de yer alan kod bulutu görseli, katılımcıların LLM'lerin yazılım geliştirme süreçlerine etkisini değerlendiren temaları görsel olarak özetlemektedir. Kodların boyutları ve renkleri, ilgili temaların önemini ve sıklığını yansıtmaktadır. Hız ve Verimlilik en belirgin tema olarak ön plana çıkarken, katılımcılar bu kategoride LLM'lerin geliştirme süreçlerini hızlandırma ve daha verimli hale getirme potansiyeline vurgu yapmıştır. Bunun yanı sıra,

Test ve Hata Ayıklama Otomasyonu ile Güvenlik ve Veri Gizliliği temaları, teknik süreçlerin iyileştirilmesi ve güvenlik konularında LLM'lerin önemini vurgulamaktadır.

Diğer temalar arasında, Proje Yönetimi ve İş Takibi, Özelleştirilmiş Eğitim ve Farkındalık ve Yazılım Belgelerinin Otomatik Üretilmesi gibi başlıklar yer almakta, bu da LLM'lerin hem teknik hem de organizasyonel süreçlerde sağladığı çok yönlü faydalara dikkat çekmektedir. Bu görsel, LLM'lerin yazılım projelerine sağladığı katkıları anlamak ve ilgili süreçlerde stratejik entegrasyon yöntemleri geliştirmek açısından yol gösterici niteliktedir.

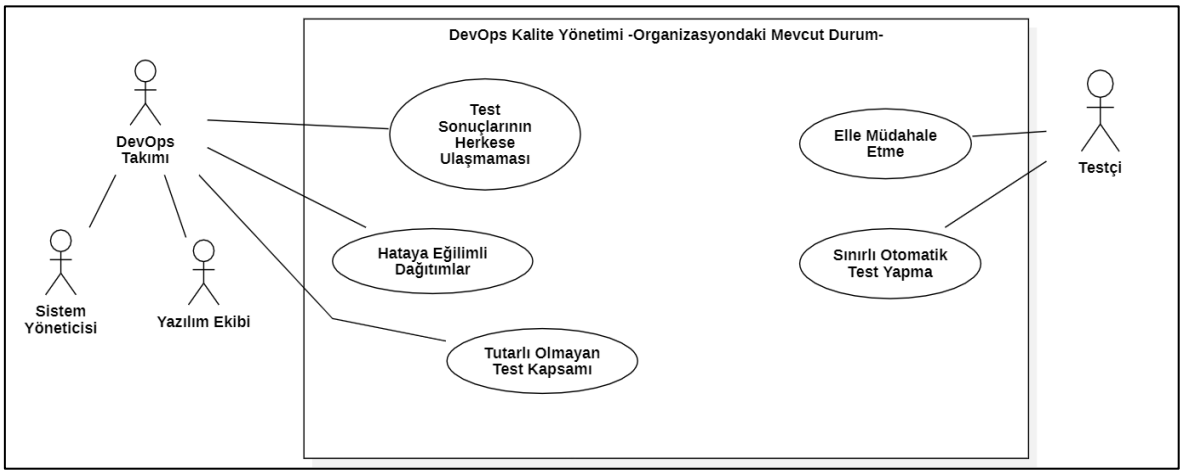
5.3. Mülakat Sonuçları ve Bulguların Modellerle Gösterimi

5.3.1. DevOps süreç yönetim modeli (“use case” diyagramı)



Şekil 5.19. DevOps Süreç Yönetimi Modeli - Use Case Diyagram

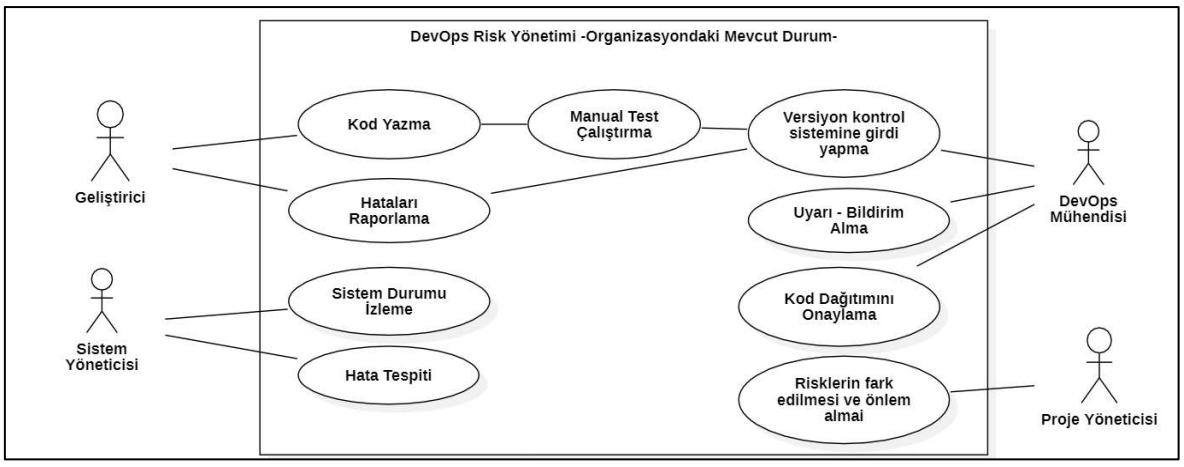
5.3.2. DevOps kalite yönetimi modeli ("use case" diyagram)



Şekil 5.20. DevOps Kalite Yönetimi Modeli - Use Case Diyagram

DevOps projelerinde kalite yönetimi, yapılan görüşmelerdeki tespitler kapsamında mevcut durumda genellikle yetersiz otomasyon ve manuel müdahalelerle sınırlıdır. Test Mühendisi ve DevOps Ekibi gibi aktörler, süreçlerin önemli bölümlerinde manuel müdahaleler yapmak zorunda kalmaktadır. Bu durum, sınırlı test otomasyonu ve sık manuel müdahaleler gibi zorluklarla sonuçlanmaktadır. Ek olarak, gecikmiş geri bildirim döngüsü süreçlerin yavaşlamasına ve hataların üretim ortamında fark edilmesine neden olmaktadır. Bu sorunlar, kalite standartlarının tutturulamamasına ve projelerin hedeflerine ulaşamamasına yol açmaktadır.

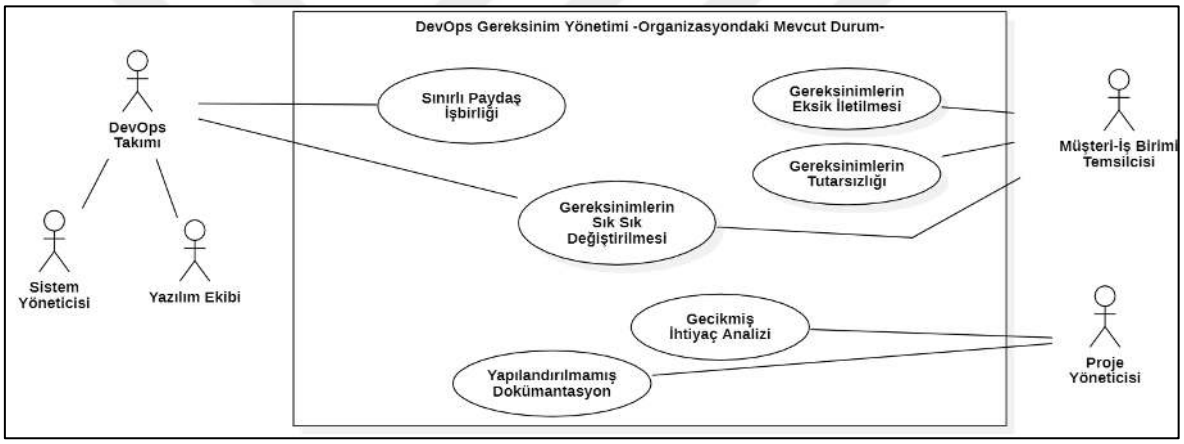
5.3.3. DevOps risk yönetimi modeli ("use case" diyagram)



Şekil 5.21. DevOps Risk Yönetimi Modeli - Use Case Diyagram

Risk yönetiminin yapılan görüşmelerdeki tespitler kapsamında mevcut durumunda, DevOps projelerinde genellikle rastgele ve sistematik olmayan bir yaklaşımla risklerin belirlendiği görülmektedir. Aktörler arasında Proje Yöneticisi ve DevOps Takımı yer alırken, risklerin belirlenmesi ve yönetimi sırasında manuel süreçlerin ağırlıkta olması dikkat çekmektedir. Örneğin, rastgele risk belirleme ve manuel risk önceliklendirme süreçleri, ekipler arasında koordinasyon eksikliğine yol açmaktadır. Ayrıca, gecikmiş risk değerlendirmesi nedeniyle potansiyel sorunlara zamanında müdahale edilemez, bu da üretim sürecinde ciddi aksamalara neden olabilir. Reaksiyonel risk yönetimi ise organizasyonların yalnızca sorun ortaya çıktıktan sonra müdahale etmesini ifade etmektedir.

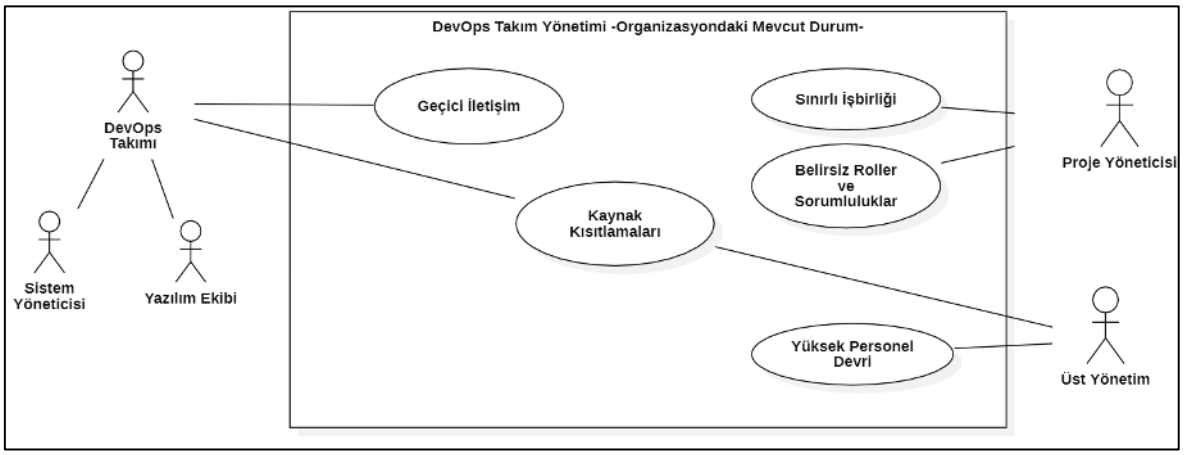
5.3.4. DevOps gereksinim yönetimi modeli ("use case" diyagram)



Şekil 5.22. DevOps Gereksinim Yönetimi Modeli - Use Case Diyagram

Gereksinim yönetiminin yapılan görüşmelerdeki tespitler kapsamında mevcut durumunda, genellikle eksik veya düzensiz bir şekilde yürütülmektedir. Proje Yöneticisi, DevOps Takımı ve Müşteri Temsilcisi, süreç boyunca iş birliği içinde hareket etse de eksik gereksinim toplama ve düzensiz dokümantasyon gibi sorunlar sıkça yaşanmaktadır. Gereksinimlerin zamanında analiz edilmemesi ve sık değişiklikler yapılması projeyi olumsuz etkilemektedir. Bu tür zorluklar, gereksinimlerin doğru bir şekilde anlaşılmasını ve uygulanmasını zorlaştırmaktadır.

5.3.5. DevOps takım yönetimi modeli (“use case” diyagramı)



Şekil 5.23. DevOps Takım Yönetimi Bileşeni - Use Case Diyagram

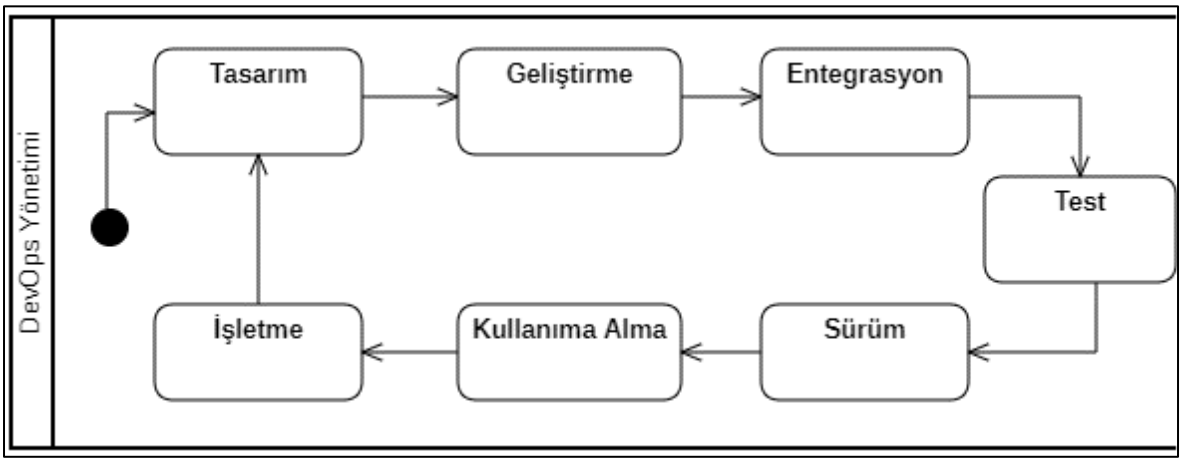
Takım yönetiminin yapılan görüşmelerdeki tespitler kapsamında mevcut durumunda, ekipler arasında zayıf iş birliği ve iletişim eksikliği yaygındır. Proje Yöneticisi, DevOps Takımı ve Üst Yönetim arasında iş birliği zayıf olduğu için sınırlı iş birliği ve düzensiz iletişim gibi sorunlar ortaya çıkmaktadır. Ekiplerin görev ve sorumluluklarının belirsiz olması ve kaynak kısıtlamaları, iş süreçlerini olumsuz etkilemektedir. Bu durum, ekip içinde motivasyon kaybına ve yüksek personel devir oranı gibi problemlere yol açmaktadır.

6. UYARLAMA MODELLERİ

Bu bölünde dedüktif tematik analize göre geliştirilen kod kitapları ile mülakat sonuçlarından hareket edilerek çalışmanın ana amacı olan uyarlama modelleri geliştirilmiştir.

6.1. Model Geliştirme

6.1.1. DevOps süreç yönetimi uyarlama modeli

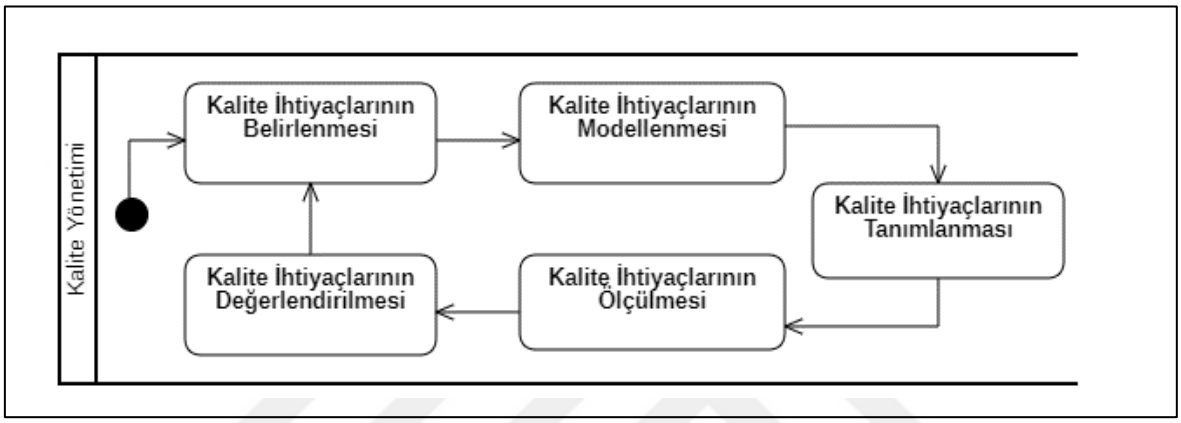


Şekil 6.1. DevOps Süreç Yönetimi

Şekil 6.1'deki model, DevOps yönetim süreçlerini temel alan bir uyarlama yaklaşımını göstermektedir ve yazılım geliştirme yaşam döngüsünün temel aşamalarını döngüsel bir yapı içinde tanımlamaktadır. Süreç, "Tasarım" aşamasıyla başlamakta ve sistem gereksinimlerinin belirlenmesi, hedeflenen işlevselliklerin planlanması ile temeller oluşturulmaktadır. "Geliştirme" aşamasında, belirlenen gereksinimler yazılım koduna dönüştürülmekte ve işlevsellikler hayata geçirilmektedir. Ardından "Entegrasyon" aşamasında, geliştirilen yazılım bileşenleri bir araya getirilerek uyumlu bir şekilde çalışmaları sağlanmaktadır. "Test" aşaması, yazılımın doğruluğunu ve güvenilirliğini değerlendirmek için kritik bir aşamadır ve bu süreçte hatalar tespit edilerek çözüme kavuşturulmaktadır. Başarılı bir test sürecinin ardından "Sürüm" aşamasına geçilmekte ve yazılım son kullanıcıya sunulmaktadır. "Kullanıma Alma" aşamasında, yazılım aktif olarak kullanılmaya başlanmakta ve performansı izlenmektedir. Sürecin son aşaması olan "İşletme"

ise yazılımın bakım ve destek hizmetlerinin sağlanmasını ve sistemin sürekliliğini içermektedir. Model, DevOps'un sürekli iyileştirme anlayışını yansıtarak işletme aşamasından tasarım aşamasına dönülebilecek döngüsel bir yapıya sahiptir. Bu yaklaşım, yazılım geliştirme ve operasyon süreçlerini entegre ederek verimliliği artırmayı, süreçlerdeki hataları erken aşamalarda tespit etmeyi ve kaliteyi yükseltmeyi hedeflemektedir.

6.1.2. Kalite yönetimi uyarlama modeli



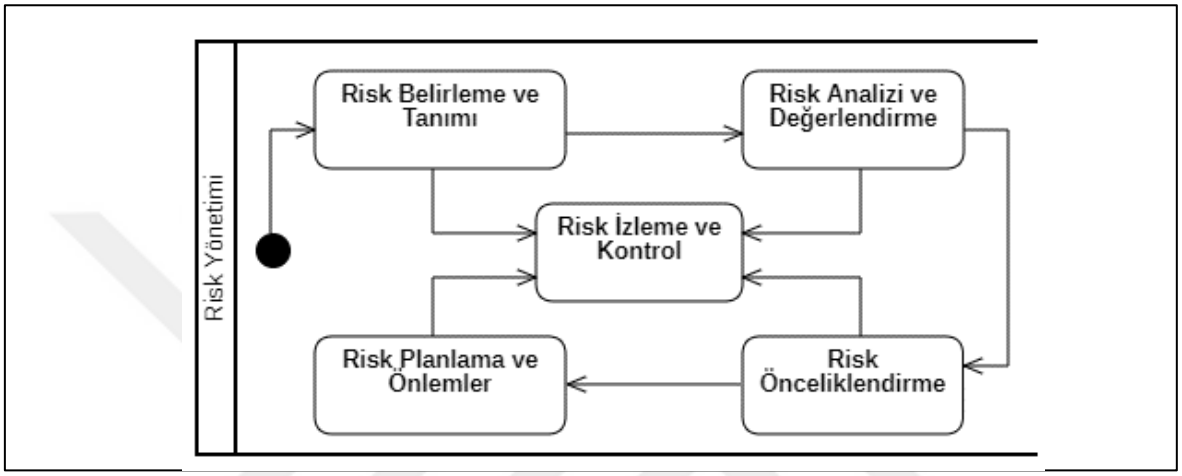
Şekil 6.2. DevOps Kalite Yönetimi

Şekil 6.2'de yer alan model, kalite yönetimi süreçlerine yönelik bir uyarlama yaklaşımını temsil etmektedir ve kalite ihtiyaçlarının yönetimini döngüsel bir yapı içerisinde açıklamaktadır. Süreç, "Kalite İhtiyaçlarının Belirlenmesi" aşamasıyla başlamakta olup, sistemin işlevsel ve işlevsel olmayan gereksinimlerini anlamayı hedeflemektedir. Bu aşamada, kaliteye yönelik beklentiler ve hedefler net bir şekilde tanımlanmaktadır. Belirlenen ihtiyaçlar, "Kalite İhtiyaçlarının Modellenmesi" aşamasında bir çerçeveye oturtulmakta ve sistematik bir yapı kazandırılmaktadır. Modellenen ihtiyaçlar, daha sonra "Kalite İhtiyaçlarının Tanımlanması" aşamasında detaylandırılarak ölçülebilir kriterlere dönüştürülmektedir.

"Kalite İhtiyaçlarının Ölçülmesi" aşamasında, tanımlanan kriterler üzerinden ölçümler yapılmakta ve sistemin kalite performansı değerlendirilmektedir. Bu aşama, kalite yönetiminin somut çıktılar sunmasını ve belirlenen hedeflere ne ölçüde ulaşıldığını göstermektedir. Son olarak, "Kalite İhtiyaçlarının Değerlendirilmesi" aşamasında elde edilen sonuçlar analiz edilmekte ve süreçte iyileştirme yapılması gereken noktalar

belirlenmektedir. Modelin döngüsel yapısı, kalite yönetiminde sürekli iyileştirme ve geri bildirim döngüsünü teşvik etmektedir. Bu model, kalite yönetimi süreçlerinin sistematik bir şekilde ele alınmasını sağlayarak kalite hedeflerinin etkin bir şekilde gerçekleştirilmesine katkıda bulunmaktadır.

6.1.3. Risk yönetimi uyarlama modeli



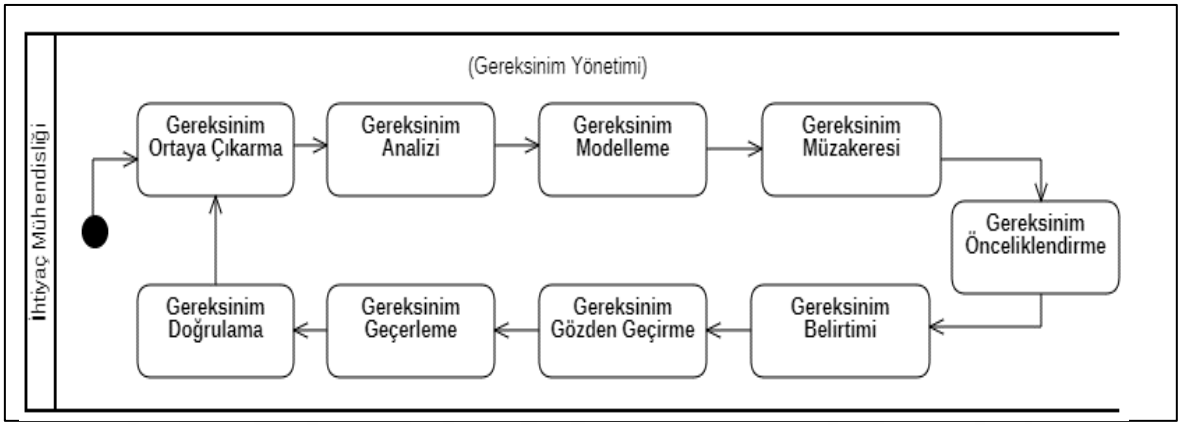
Şekil 6.3. DevOps Risk Yönetimi

Şekil 6.3'deki model, risk yönetimi süreçlerini döngüsel bir yapı içerisinde ele alan bir uyarlama yaklaşımını temsil etmektedir. Süreç, "Risk Belirleme ve Tanımı" aşamasıyla başlamakta olup, olası risklerin tespit edilmesi ve bu risklerin proje üzerindeki etkilerinin tanımlanmasını amaçlamaktadır. Belirlenen riskler, "Risk Analizi ve Değerlendirme" aşamasında detaylı bir şekilde incelenmekte ve projeye olası etkileri bakımından analiz edilmektedir.

"Risk Önceliklendirme" aşamasında, risklerin projeye etkisi ve gerçekleşme olasılığına göre sıralanması sağlanmaktadır. Bu önceliklendirme, risklerin yönetiminde hangi adımlara öncelik verilmesi gerektiğini belirlemek açısından kritik öneme sahiptir. "Risk Planlama ve Önlemler" aşamasında ise, önceliklendirilmiş risklere yönelik önleyici veya düzeltici eylem planları hazırlanmakta ve risklerin etkisini azaltmak için stratejiler geliştirilmekte ve uygulanmaktadır.

Modelin merkezinde yer alan "Risk İzleme ve Kontrol" aşaması, risklerin sürekli olarak izlenmesini ve alınan önlemlerin etkisinin değerlendirilmesini sağlamaktadır. Bu aşama, risklerin değişen proje koşullarına göre yeniden ele alınmasını ve gerekli güncellemelerin yapılmasını mümkün kılmaktadır. Modelin döngüsel yapısı, risk yönetiminde sürekli bir izleme, değerlendirme ve iyileştirme döngüsünü teşvik etmektedir. Bu yaklaşım, risklerin etkin bir şekilde yönetilmesine ve projenin başarıyla tamamlanmasına önemli katkılar sunmaktadır.

6.1.4. Gereksinim yönetimi uyarlama modeli



Şekil 6.4. DevOps Gereksinim Yönetimi

Şekil 6.4'deki model, ihtiyaç mühendisliği kapsamında gereksinim yönetim süreçlerini açıklayan bir uyarlama yaklaşımını temsil etmektedir. Model, gereksinimlerin sistematik bir şekilde ele alınmasını sağlamak için birbirine bağlı adımlardan oluşan döngüsel bir yapıya sahiptir. Süreç, "Gereksinim Ortaya Çıkarma" aşamasıyla başlamakta olup, paydaşların beklentilerinin ve proje hedeflerinin belirlenmesini amaçlamaktadır. Bu aşama, gereksinim yönetiminin temelini oluşturmaktadır.

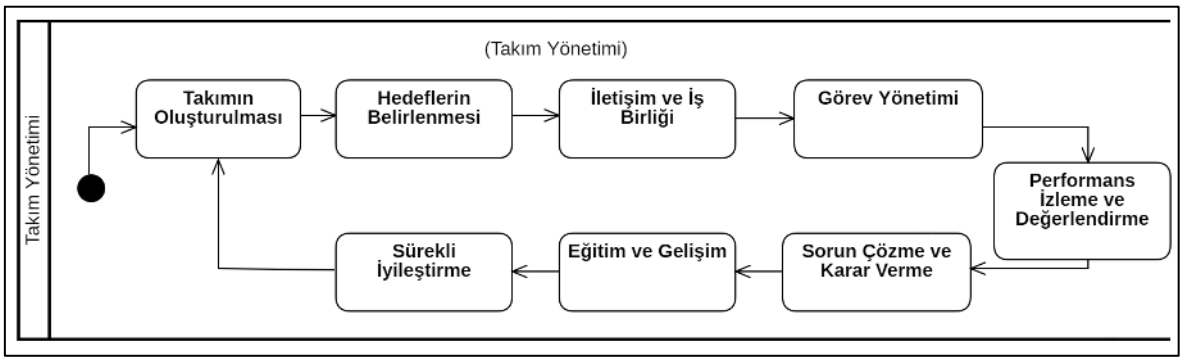
"Gereksinim Analizi" aşamasında, ortaya çıkarılan gereksinimler detaylı olarak incelenmekte ve gereksinimlerin teknik ve işlevsel uygulanabilirliği değerlendirilmektedir. "Gereksinim Modelleme" aşamasında ise, gereksinimler yapılandırılmış bir formatta ifade edilmekte ve sistem tasarımına entegrasyonu kolaylaştırılmaktadır. Modelleme sonrası, "Gereksinim Müzakeresi" adımı, gereksinimlerin paydaşlarla tartışıldığı ve uzlaşmaya

varıldığı bir aşamadır. Bu sürecin, gereksinimlerin projeye uygun hale getirilmesi için önemli olduğu değerlendirilmektedir.

"Gereksinim Önceliklendirme" aşamasında, gereksinimlerin öncelik sıralaması yapılmakta ve hangi gereksinimlere daha fazla önem verilmesi gerektiği belirlenmektedir. Ardından "Gereksinim Belirtimi" aşamasında, gereksinimler yazılı bir biçimde tanımlanarak belge haline getirilmektedir. Bu belge, proje ekiplerinin ortak bir anlayış geliştirmesini sağlamaktadır. "Gereksinim Gözden Geçirme" aşaması, belirlenen gereksinimlerin uygunluk ve tutarlılık açısından kontrol edildiği bir adımı temsil etmektedir.

"Gereksinim Geçerleme" aşamasında, gereksinimlerin doğruluğu ve sistem ihtiyaçlarını karşılama düzeyi değerlendirilmekte, "Gereksinim Doğrulama" aşamasında ise gereksinimlerin doğru bir şekilde uygulandığından emin olunmaktadır. Modelin döngüsel yapısı, gereksinimlerin sürekli izlenmesini ve gerektiğinde yeniden ele alınmasını sağlayarak esneklik ve sürekli iyileştirme imkanı sunmaktadır. Bu yaklaşım, gereksinim yönetim süreçlerinde etkinliği artırmayı ve projelerin başarı şansını yükseltmeyi hedeflemektedir.

6.1.5. Takım yönetimi uyarlama modeli



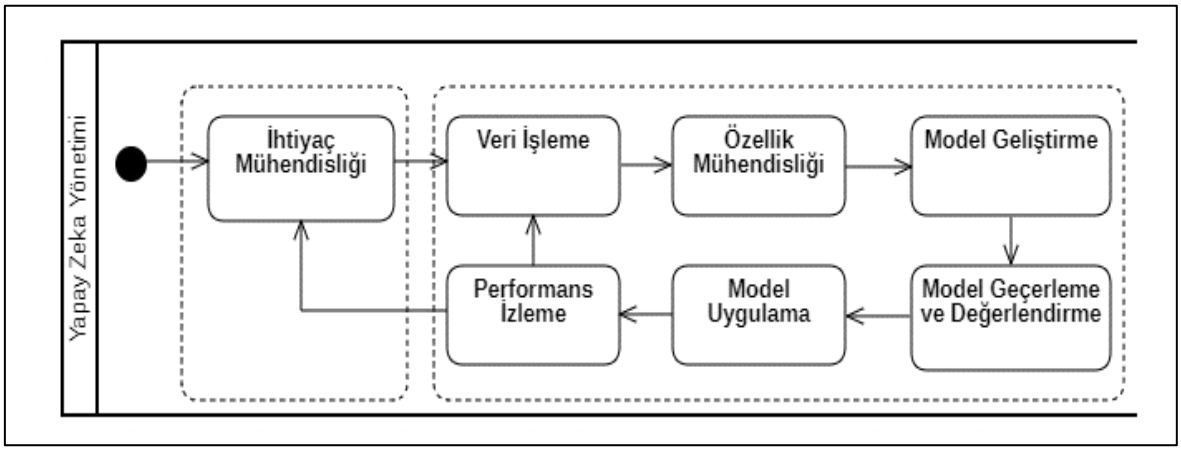
Şekil 6.5. DevOps Takım Yönetimi

Şekil 6.5'de yer alan modelin döngüsel yapısı, takım yönetimi süreçlerinin esnek ve sürekli iyileştirme odaklı bir çerçevede ele alınmasını sağlamaktadır. Özellikle, iletişim ve iş birliğinin vurgulanması, takım üyeleri arasında sinerji yaratırken, görev yönetimi ve

performans izleme aşamaları, süreçlerin ölçülebilir bir yapıda değerlendirilmesine olanak tanımaktadır.

Modelin uygulanması, sadece takım üyelerinin bireysel performansını artırmakla kalmayacak, aynı zamanda organizasyonel hedeflere ulaşılmasında önemli bir araç olarak işlev görecektir. Bu yapı, takım yönetimi süreçlerinin daha etkin bir şekilde planlanması ve yürütülmesi için yol gösterici bir rehber niteliğindedir.

6.1.6. Yapay zekâ yönetimi uyarlama modeli



Şekil 6.6. DevOps Yapay Zeka Yönetimi

Şekil 6.6'da yer alan model, yapay zeka yönetimine yönelik süreçleri temsil eden bir uyarlama yaklaşımını göstermektedir. Model, ihtiyaç mühendisliği ile başlayan ve veri odaklı bir döngü çerçevesinde işleyen bir yapıya sahiptir. Süreç, "İhtiyaç Mühendisliği" aşamasıyla başlamakta olup, yapay zeka sistemine ilişkin gereksinimlerin belirlenmesi ve tanımlanmasını amaçlamaktadır. Bu aşama, sistemin hedeflerine uygun bir yapay zeka çözümü tasarlanması için temel oluşturmaktadır.

Belirlenen ihtiyaçlar doğrultusunda "Veri İşleme" aşamasına geçilir. Bu aşamada, yapay zeka modellerinin eğitimi ve doğruluğu için gerekli olan veriler toplanmakta, temizlenmekte ve işlenmektedir. İşlenmiş veriler, "Özellik Mühendisliği" aşamasında modelin performansını artıracak özelliklerin seçilmesi ve dönüştürülmesi süreçlerine tabi tutulmaktadır. Daha sonra "Model Geliştirme" aşamasında, yapay zeka modelleri bu veriler ve özellikler üzerinden eğitilmekte ve geliştirilmektedir.

Geliştirilen modellerin performansı, "Model Geçerleme ve Değerlendirme" aşamasında test edilmekte ve modellerin ihtiyaçları ne ölçüde karşıladığı değerlendirilmektedir. Doğrulama sürecinde modelin genel başarı düzeyi ve hataları analiz edilerek sonuçların güvenirliği artırılmaktadır. Model, geçerleme aşamasını başarıyla tamamladıktan sonra "Model Uygulama" aşamasında operasyonel süreçlere entegre edilmektedir. Uygulama aşamasını takiben, modelin gerçek dünyadaki performansı sürekli olarak "Performans İzleme" aşamasında takip edilmektedir. Bu izleme süreci, modelin etkinliğini sürdürülebilir kılmayı ve gerektiğinde yeniden eğitilmesini veya güncellenmesini sağlamaktadır.

Modelin döngüsel yapısı, yapay zeka sistemlerinde sürekli iyileştirme, esneklik ve etkinlik sağlamayı amaçlamaktadır. Gereksinim mühendisliğinden performans izlemeye kadar olan tüm aşamalar birbiriyle bağlantılı olup, bu yaklaşım yapay zeka çözümlerinin hem teknik hem de iş hedeflerine uygun şekilde tasarlanmasını ve uygulanmasını desteklemektedir.

6.2. Uyarlama Modellerini Geçerleme ve Doğrulama

6.2.1. Senaryo analizi

6.2.1.1. Senaryonun tanımlanması

Araştırma doğrultusundaki senaryo analizinin kapsamını çalışmaya konu olan savunma sanayii kurumunun pilot eğitimi için geliştirmiş olduğu bir simülatör yazılımı oluşturmaktadır. Simülatör yazılımları, gerçek dünyadaki sistem, süreç veya olayların sanal ortamda eğitimi ile çeşitli problemlerin çözümlerine yönelik modellerin geliştirilmesi ve analiz edilmesi amacıyla geliştirilen yazılımlardır. Bu tür yazılımlar, karmaşık sistemlerin farklı durumlarda nasıl çalışacağını öngörmek, optimize etmek, eğitim sağlamak veya karar destek süreçlerine rehberlik etmek için kullanılmaktadır. Senaryonun kapsamış olduğu ana temalar şunlardır:

- DevOps proje yönetim süreçlerinin simülatör yazılım geliştirme süreçleri,

- Simülatör yazılımının ihtiyaç yönetim süreçleri,
- Simülatör yazılımının takım yönetim süreçleri,
- Simülatör yazılımının risk yönetim süreçleri,
- Simülatör yazılımının kalite yönetim süreçleri,
- Simülatör yazılımının yapay zekâ bileşeni ve yönetimi.

Senaryonun temel aktörlerini mülakatlara katılanlar arasından belirlenen yazılım takımı ile araştırma ekibi (araştırmacı ve danışmanı) oluşturmuştur. Senaryonun tasarımı ve geliştirilmesi araştırma ekibi, çalıştırılması ve değerlendirilmesini ise yazılım ekibi gerçekleştirmiştir.

6.2.1.2. Model geçerleme ölçütlerinin tanımlanması

Tablo 6.1'de gösterildiği gibi geliştirilen uyarlama modelleri (a) içerik ve (b) UML kuralları olmak üzere iki ana başlık altında toplanan ölçütlere göre değerlendirilmiştir:

Tablo 6.1. Model Geçerleme Ölçütleri

	Temalar					
	DevOps Süreçleri	**İhtiyaç Yönetimi**	**Takım Yönetimi**	**Risk Yönetimi**	**Kalite Yönetimi**	**Yapay Zekâ Yönetimi**
Modellerin Kod Kitaplarına Uygunluğu:						
Aktörler	Bütün aktörleri kapsanıyor mu? (İç ve dış paydaşlar, birincil ve ikinci aktörler vb.)					
Kullanım Örnekleri ("Use Case")	Geliştirilen modeller ilgili temadaki kullanım örneklerini içeriyor mu?					
Süreçler	Geliştirilen modeller ilgili temadaki süreçleri içeriyor mu?					
Etkinlikler	Geliştirilen modeller ilgili temadaki süreçleri etkinlik olarak içeriyor mu?					
İş Akışı	Geliştirilen modeller ilgili temadaki iş akışlarını içeriyor mu?					
Karar Noktaları	Geliştirilen modeller ilgili temadaki karar noktalarını içeriyor mu?					
Döngüler	Geliştirilen modeller ilgili temadaki geri bildirim yapılarını içeriyor mu?					
Modellerin Kullanıcı İhtiyaçlarına Uygunluğu:						
İşlevsel İhtiyaçlar	Geliştirilen modeller kullanıcıların bütün işlevsel ihtiyaçlarını karşılıyor mu?					
İşlevsel Olmayan İhtiyaçlar	Geliştirilen modeller kullanıcıların bütün işlevsel olmayan ihtiyaçlarını karşılıyor mu?					
Modellerin UML Kurallarına Uygunluğu:						
Söz Dizimsel Uygunluk	Geliştirilen modeller UML söz dizimsel kurallarına uygun mu?					
Anlamsal Uygunluk	Geliştirilen modeller anlamsal olarak UML kurallarına uygun mu?					
5: Çok Yüksek; 4: Yüksek; 3: Orta; 2: Düşük; 1: Çok Düşük						

(5) Çok Yüksek Düzeyde Karşılanıyor: Ölçüt, ilgili bileşen tarafından eksiksiz ve mükemmel bir şekilde karşılanıyor. Örneğin: "Geri Bildirim Döngüsü" tamamen etkin ve sürekli iyileştirme sağlıyor.

(4) Yüksek Düzeyde Karşılanıyor: Ölçüt, büyük ölçüde karşılanıyor ancak bazı eksiklikler mevcut. Örneğin: "Performans ve Kaynak Kullanımı" genel olarak iyi yönetiliyor ancak zaman zaman optimizasyon fırsatları bulunuyor.

(3) Orta Düzeyde Karşılanıyor: Ölçüt, kısmen karşılanıyor. Eksiklikler veya iyileştirme gereksinimleri açık bir şekilde görülebilir. Örneğin: "Karar Noktalarının Doğruluğu" bazı durumlarda hatalı karar mekanizmaları içeriyor.

(2) Düşük Düzeyde Karşılanıyor: Ölçüt, belirgin eksiklikler gösteriyor ve sürecin iyileştirilmesi gerekiyor. Örneğin: "Risk Yönetimi Uygulama' sının yalnızca temel seviyede ele alındığı ve ayrıntılı bir strateji olmadığı fark ediliyor.

(1) Çok Düşük Düzeyde Karşılanıyor: Ölçüt, ilgili bileşen tarafından neredeyse hiç karşılanmıyor veya uygulanmıyor. Örneğin: "Test Otomasyonu Seviyesi" hiç mevcut değil ya da yalnızca manuel süreçler kullanılıyor.

6.2.1.3. Senaryonun çalıştırılması

Bu aşamada, araştırma doğrultusunda tasarlanan senaryonun çalıştırılması ve test edilmesi süreçleri gerçekleştirilmiştir. Senaryo, pilot eğitim simülatör yazılımının geliştirilmesi ve uygulanması bağlamında, belirlenen aktörlerin etkileşimlerini ve süreçleri gözlemlemek için yapılandırılmıştır. Yazılım ekibi tarafından senaryonun uygulanması sırasında her bir tema (örneğin, DevOps süreçleri, ihtiyaç yönetimi, takım yönetimi vb.) adım adım ele alınmış, bu süreçlerde modellenen iş akışlarının uygunluğu ve etkinliği değerlendirilmiştir.

Senaryonun çalıştırılması sırasında, belirlenen süreçlere yönelik kullanım örnekleri (use case) takip edilerek, modellerin gereksinimlere uygunluğu, kullanıcı ihtiyaçlarını karşılama düzeyi ve UML kurallarına uyumu incelenmiştir. Özellikle, iş akışlarının ve karar noktalarının gerçek dünyadaki uygulamalarla tutarlılığına dikkat edilmiştir. Bu süreçte,

araştırma ekibi ve yazılım ekibi arasındaki iş birliği, geri bildirim döngüleri ile sürekli iyileştirme için veri toplama ve analiz yapma fırsatı sunmuştur.

Sonuç olarak, senaryonun çalıştırılması sırasında elde edilen bulgular, geliştirilen modellerin geçerliliği ve doğruluğuna ilişkin önemli içgörüler sağlamış ve modelleme süreçlerinin hem teorik hem de pratik uygulamalarda etkinliğini değerlendirmeye olanak tanımıştır. Çalıştırma aşamasından elde edilen bu sonuçlar, araştırmanın genel amaçlarına yönelik önemli katkılar sunmuş ve uyarlama modellerinin iyileştirilmesine temel oluşturmuştur.

Tablo 6.2. Senaryo Değerlendirme Matrisi

Bileşenler	Aktörler	Kullanım Örnekleri	Süreçler	Etkinlikler	İş Akışı	Karar Noktaları	Döngüler	İşlevsel İhtiyaçlar	İşlevsel Olmayan İhtiyaçlar	UML Kuralları
DevOps Süreçleri	4	5	4	5	4	4	5	5	4	5
İhtiyaç Yönetimi	5	4	4	3	4	3	3	4	4	5
Takım Yönetimi	4	4	3	4	3	4	4	5	4	5
Risk Yönetimi	3	4	3	4	4	5	4	4	3	4
Kalite Yönetimi	4	5	4	5	4	3	4	4	4	5
Yapay Zeka Yönetimi	5	5	4	5	4	4	5	5	4	5

6.2.1.4. Bulgular ve değerlendirme

Bu bölümde, araştırma kapsamında geliştirilen uyarlama modellerinin geçerleme ve doğrulama süreçlerinden elde edilen bulgular detaylı olarak ele alınmakta ve senaryo analizi çerçevesinde değerlendirmeler sunulmaktadır. Araştırmanın temel amacı doğrultusunda, belirlenen ölçütlere uygun olarak geliştirilen modellerin değerlendirilmesi ve modellenen süreçlerin etkinliği kapsamlı bir analizle ortaya konulmuştur.

6.2.1.4.1. Senaryo çalıştırma ve değerlendirme bulguları

Araştırmanın senaryo analizi, savunma sanayiinde pilot eğitimi için geliştirilen bir simülatör yazılımı bağlamında gerçekleştirilmiştir. Bu bağlamda, DevOps proje yönetimi kapsamında tanımlanan altı ana tema (DevOps süreçleri, ihtiyaç yönetimi, takım yönetimi, risk yönetimi, kalite yönetimi, yapay zeka yönetimi) sistematik olarak ele alınmış ve

modellenen süreçlerin gerçek dünyadaki uygulanabilirliği değerlendirilmiştir. Simülatör yazılımı bağlamında yapılan analizler, bu temaların yazılım geliştirme süreçlerine entegrasyonunda önemli bulgular sunmuştur.

- DevOps Süreçleri bileşeni kapsamında, kullanım örneklerinin (use case) ve geri bildirim döngülerinin etkili bir şekilde modellenmiş olduğu gözlemlenmiştir. Ancak, bazı iş akışlarında ve karar noktalarında eksiklikler tespit edilmiştir. Süreçlerin otomasyon odaklı yapısı, teslim sürelerini kısaltmak açısından güçlü bir çerçeve sunarken, karar mekanizmalarının daha net tanımlanması gerektiği ortaya çıkmıştır.
- İhtiyaç Yönetimi bileşeni, işlevsel ve işlevsel olmayan gereksinimlerin büyük ölçüde karşılandığını göstermiştir. Özellikle gereksinim tanımlama ve geçerleme süreçleri etkin bir şekilde ele alınmış; ancak, gözden geçirme ve doğrulama süreçlerinde daha fazla analitik yaklaşıma ihtiyaç duyulmuştur. Bu, gereksinimlerin değişen proje gerekliliklerine adaptasyonu açısından kritik bir bulgudur.
- Takım Yönetimi bileşeni kapsamında yapılan değerlendirmeler, takım içi iletişim ve iş birliği süreçlerinin yüksek düzeyde etkin olduğunu ortaya koymuştur. Takım üyelerinin rolleri açıkça tanımlanmış ve görev dağılımı dengeli bir şekilde gerçekleştirilmiştir. Bununla birlikte, bazı karar mekanizmalarının daha kolektif bir yaklaşımla ele alınması gerektiği vurgulanmıştır.
- Risk Yönetimi süreçleri, planlama ve önlem alma süreçlerinin güçlü bir yapıya sahip olduğunu göstermiştir. Ancak, risklerin önceliklendirilmesi ve etkilerinin analiz edilmesi aşamalarında eksiklikler tespit edilmiştir. Özellikle, simülatör yazılımı gibi karmaşık sistemlerde risklerin daha ayrıntılı bir şekilde analiz edilmesi gerektiği sonucuna ulaşılmıştır.
- Kalite Yönetimi bileşeni, süreç etkinliklerinin ve geri bildirim döngülerinin oldukça yüksek bir düzeyde karşılandığını göstermiştir. Ancak, kalite gereksinimlerinin tanımlanması sırasında daha detaylı bir dokümantasyon ve analiz sürecine ihtiyaç duyulduğu belirlenmiştir. Bu, kalite hedeflerinin sürdürülebilir bir şekilde sağlanması açısından önem taşımaktadır.
- Yapay Zeka Yönetimi bileşeni, model geliştirme, geçerleme ve değerlendirme süreçlerinde yüksek performans sergilemiştir. Özellikle veri işleme ve özellik mühendisliği aşamalarında başarılı sonuçlar elde edilmiştir. Ancak, performans izleme süreçlerinin sürekli iyileştirme perspektifiyle ele alınması gerektiği tespit edilmiştir.

6.2.1.4.2. Matris değerlendirme sonuçları

Tablo 6.2'de sunulan Senaryo Değerlendirme Matrisi, her bir bileşenin belirlenen ölçütler doğrultusunda kapsamlı bir şekilde analiz edilmesini sağlamıştır. Matrisin değerlendirme sonuçları, bileşenlerin güçlü yönlerini ve geliştirilmesi gereken alanlarını ortaya koymuştur. Öne çıkan bulgular şu şekildedir:

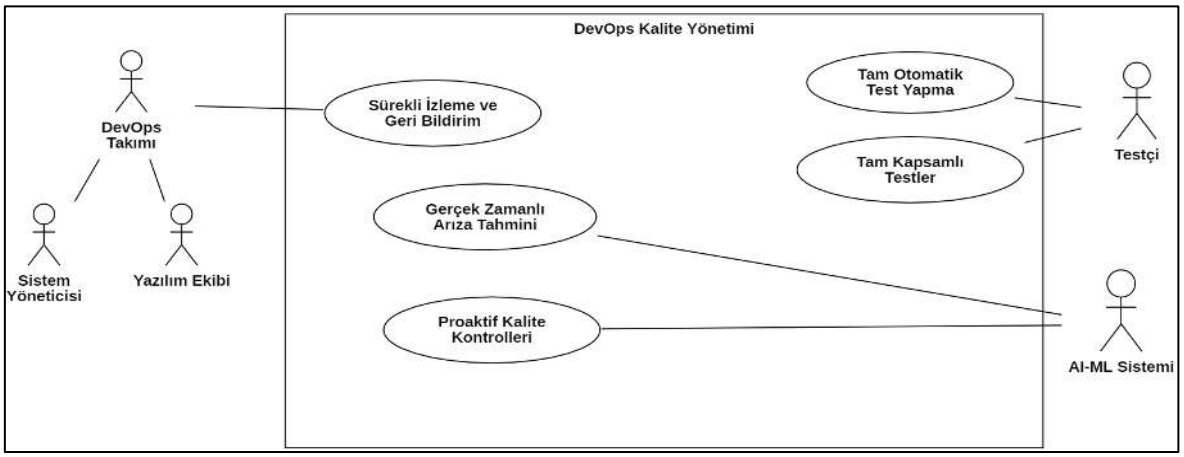
- Aktörlerin Kapsamı: Genel olarak yüksek düzeyde karşılanmış; ancak bazı süreçlerde dış paydaşların dahil edilmesi gerektiği görülmüştür. Bu eksiklik, süreçlerin dış bağımlılıkları daha iyi ele alması gerektiğini göstermektedir.
- Kullanım Örnekleri ve Süreçler: Modellerin, belirlenen temalarla ilgili kullanım örneklerini ve süreçleri büyük ölçüde içerdiği gözlemlenmiştir. Ancak, bazı temalarda iş akışlarının daha ayrıntılı tanımlanmasına ihtiyaç duyulmuştur.
- UML Kurallarına Uygunluk: Geliştirilen modellerin söz dizimsel ve anlamsal uygunluk açısından tatmin edici bir performans sergilediği görülmüştür. Bu durum, modellerin teknik olarak geçerli bir temele sahip olduğunu doğrulamaktadır.

6.2.1.4.3. Genel değerlendirme

Araştırmadan elde edilen bulgular, geliştirilen uyarlama modellerinin hem teorik hem de pratik uygulamalarda önemli bir potansiyele sahip olduğunu göstermektedir. Simülatör yazılımı bağlamında yapılan senaryo analizi, modellerin yazılım geliştirme süreçleriyle uyumunu ve uygulanabilirliğini ortaya koymuştur. Ancak, bazı süreçlerin daha iyi optimize edilmesi ve detaylandırılması gerektiği anlaşılmıştır.

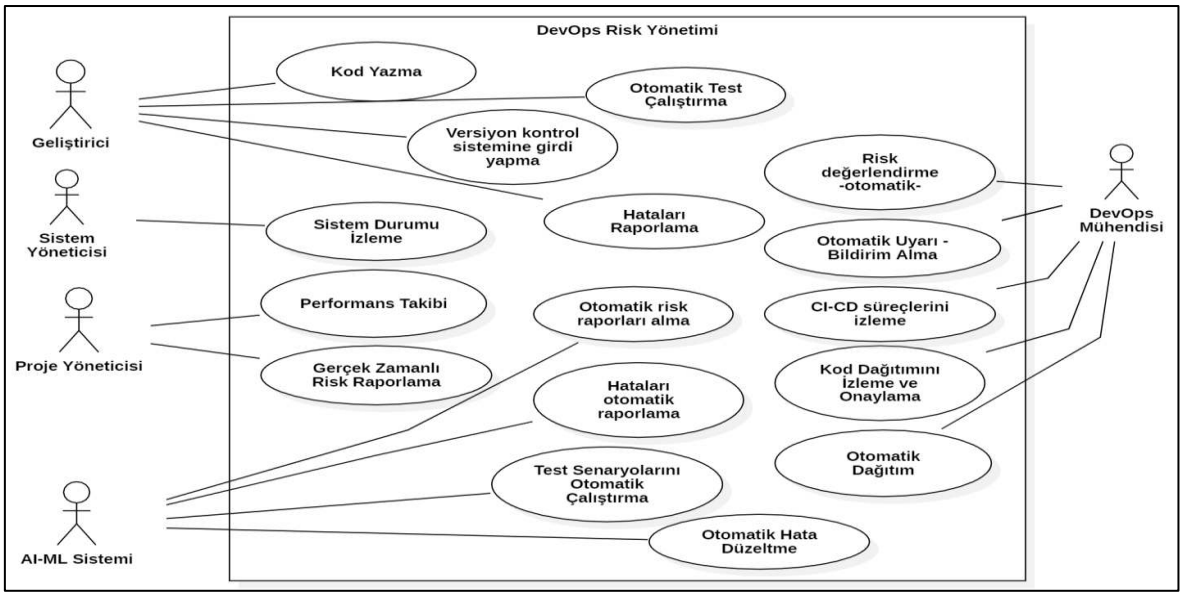
Sonuç olarak, bu araştırmanın çıktılarının hem yazılım mühendisliği hem de proje yönetimi disiplinlerinde teorik bilgi birikimine ve uygulama süreçlerine önemli katkılar sunabileceği değerlendirilmektedir. Özellikle, DevOps süreçlerinin entegrasyonu ve yapay zeka bileşenlerinin yönetimine ilişkin bulgular, savunma sanayii ve benzeri yüksek teknoloji alanlarında referans bir çerçeve sunmaktadır. Gelecekte yapılacak çalışmaların, bu araştırmanın metodolojik yaklaşımından ve bulgularından faydalanarak daha geniş kapsamlı uygulamalar geliştirebileceği öngörülmektedir.

6.3. Uyarlama Modellerinin Güncellenmesi



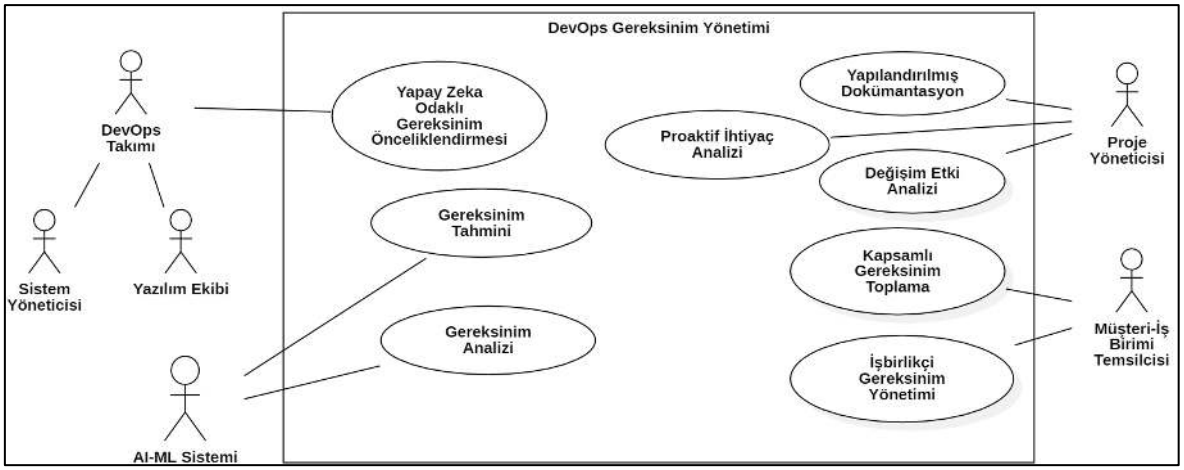
Şekil 6.7. DevOps Kalite Yönetimi Güncellenen Model

Kalite yönetiminin kuramsal çerçevedeki durumu ve görüşmecilerden gelen öneriler kapsamında, otomasyonun etkin bir şekilde kullanıldığı süreçleri içermektedir. Tam test otomasyonu sayesinde manuel müdahaleler en aza indirilmektedir. Gerçek zamanlı geri bildirim ve proaktif kalite kontrolleri, sorunların erken tespit edilmesini ve çözülmesini sağlamaktadır. Yapay Zeka/Makine Öğrenimi Motoru, test süreçlerini optimize ederek tahmine dayalı hata tespiti gerçekleştirebilir. Sürekli izleme ve geri bildirim, tüm sürecin daha hızlı ve daha kaliteli bir şekilde ilerlemesine olanak tanımaktadır. Böylece, kalite yönetimi proaktif ve sürekli iyileştirilen bir hale gelmektedir.



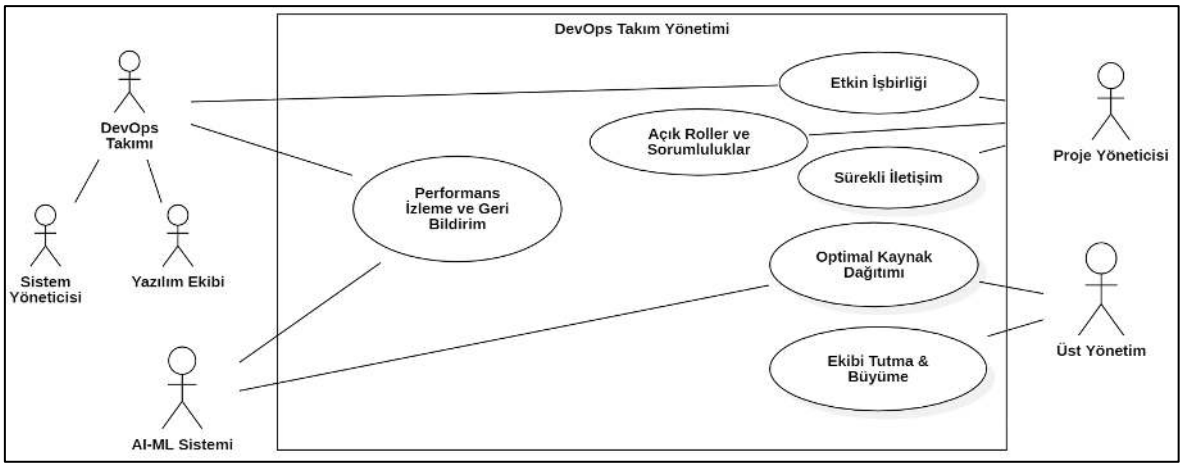
Şekil 6.8. DevOps Risk Yönetimi Güncellenen Model

Şekil 6.8'de yer alan modele göre risk yönetiminin kuramsal çerçevedeki durumu ve görüşmecilerden gelen öneriler kapsamında proaktif bir yaklaşımla ele alınması gerekmektedir. Otomatik risk belirleme ve değişiklik etkisi analizi gibi teknolojik araçlarla tespit edilmesi sağlanabilmektedir. Yapay Zeka/Makine Öğrenimi Motoru, potansiyel sorunları önceden tahmin ederek risklerin önceliklendirilmesine katkıda bulunmaktadır. Ayrıca, reaktif risk önceliklendirme ve iş birliği temelli risk yönetimi süreçleriyle proje yöneticisi ve DevOps ekibi arasında sürekli iletişim ve koordinasyon sağlanmaktadır. Bu ideal durum, risklerin daha verimli yönetilmesine ve proje performansının iyileştirilmesine olanak tanımaktadır.



Şekil 6.9. DevOps Gereksinim Yönetimi Güncellenen Model

Şekil 6.9'daki modele göre gereksinim yönetiminin kuramsal çerçevedeki durumu ve görüşmecilerden gelen öneriler kapsamında sistematik ve teknoloji destekli bir yaklaşımla ele alınmalıdır. Kapsamlı gereksinim toplama süreçleri sayesinde eksiksiz bilgi toplanabileceği değerlendirilmektedir. Yapılandırılmış dokümantasyon ile gereksinimlerin standart bir formatta tutulması sağlanabilmektedir. Ayrıca, yapay zeka destekli gereksinim önceliklendirme ve değişiklik etkisi analizi süreçleri, gereksinimlerin etkili bir şekilde yönetilmesine yardımcı olmaktadır. İş birliği temelli gereksinim yönetimi, tüm paydaşların dahil olduğu sürekli iletişim ve koordinasyon sürecini ifade etmektedir. Bu yapı, gereksinim yönetimini daha etkili ve verimli hale getirmektedir.



Şekil 6.10. DevOps Takım Yönetimi Güncellenen Model

Şekil 6.10'daki modele göre, takım yönetiminin kuramsal çerçevedeki durumu ve görüşmecilerden gelen öneriler kapsamında iş birliğine dayalı ve teknolojik olarak desteklenmiş bir yapıya sahip olması gerekmektedir. İş birliği yapan ekipler sayesinde ekip üyeleri arasında güçlü bir bağ kurulmaktadır. Açık görev ve sorumluluklar ile herkesin ne yapması gerektiği net bir şekilde belirlenmektedir. Sürekli iletişim ile bilgi akışı düzenli ve kesintisiz bir hale getirilmektedir. Yapay Zeka/Makine Öğrenimi Motoru, kaynakların en verimli şekilde tahsis edilmesini ve ekip performansının sürekli izlenmesini sağlamaktadır. Ekip sürekliliği ve büyümesi süreçleri, ekip motivasyonunu artırarak daha verimli ve başarılı bir ortam yaratmaktadır.

7. TARTIŞMA

DevOps'un uygulanmasında karşılaşılan teknik zorluklar, özellikle altyapı otomasyonu ve CI/CD süreçlerinin entegrasyonu bağlamında belirginleşmektedir (Tamanampudi, 2019). CI/CD süreçlerinin doğru yönetilmemesi, kod hatalarının geç tespit edilmesine ve yazılım hatalarının hızlı bir şekilde çözülememesine neden olmaktadır (Jayakody ve Wijayanayake, 2022). Dedüktif analiz, bu teknik zorlukların araştırma sorularıyla nasıl ilişkilendirildiğini ve mevcut teorik çerçeveye uygunluğunu belirlemekte önemli bir rol oynamıştır. Özellikle log analizi gibi büyük veri yönetimi işlemlerinin manuel olarak yürütülmesi, süreçlerde ciddi zaman kaybına yol açmaktadır (Vemuri ve Tatikonda, 2024). Bu bulgu, endüktif analiz yoluyla görüşmelerden elde edilen yeni temalar ve örneklerle desteklenmiştir. Çözüm olarak, DevOps süreçlerinin başarısının artırılması için otomasyon araçlarının daha yaygın ve etkili bir şekilde kullanılması önerilmektedir (Fawzy ve diğerleri, 2023).

Organizasyonel zorluklar ise kültürel direnç ve iş birliği eksikliği etrafında yoğunlaşmaktadır (Patel ve Tyagi, 2022). DevOps, organizasyon yapısında köklü değişimler gerektiren bir yaklaşım olduğundan, bu değişimlere karşı gösterilen direnç uygulamanın başarısını olumsuz etkileyebilmektedir (Khan ve diğerleri, 2022). Dedüktif analiz, silo yapılarının ekipler arası iletişim kopukluklarına yol açarak organizasyonel etkinliği azalttığını göstermiştir. Endüktif analiz, bu soruna yönelik, mülakat verilerinden elde edilen "kültürel değişime duyulan ihtiyaç" ve "liderlik stratejilerinin etkisi" gibi temaları ortaya çıkarmaktadır. Bu bağlamda, liderlik stratejilerinin kültürel değişimi teşvik edecek şekilde yapılandırılması, DevOps'un organizasyon içinde benimsenmesini kolaylaştırmaktadır (Maroukian ve Gulliver, 2020).

Sonuç olarak, bu çalışmada kullanılan dedüktif ve endüktif analiz yaklaşımları, DevOps'un teknik ve organizasyonel zorluklarını derinlemesine incelemek ve çözüm yolları geliştirmek için kapsamlı bir çerçeve sağlamıştır. Dedüktif analiz, mevcut teorik çerçeveye uygun bulguları sistematize ederken; endüktif analiz, yeni temalar ve içgörülerle bu bulguları zenginleştirmiştir. Bu yöntemlerin bir arada kullanılması, hem teorik hem de pratik bağlamda DevOps adaptasyonuna yönelik stratejik önerilerin geliştirilmesine olanak tanımıştır.

Katılımcıların ifadeleri doğrultusunda, bulgular dört ana tema altında incelenmiştir: İletişim ve İşbirliği Eksiklikleri, Teknolojik Entegrasyon Zorlukları, Süreç Yönetimi ve Otomasyon, ve Kültürel Değişim.

7.1. İletişim ve İşbirliği Eksiklikleri

Mülakatlarda, geliştirici ve operasyon ekipleri arasındaki iletişim eksikliklerinin projelerin gecikmesine ve yüksek hata oranlarına yol açtığı sıkça vurgulanmıştır. Katılımcılar, geri bildirim döngülerinin yavaş ilerlediğini ve bu durumun değişikliklerin etkili bir şekilde değerlendirilmesine engel olduğunu belirtmiştir. Özellikle ekipler arası silo etkisinin, proaktif sorun çözme çabalarını sekteye uğrattığı ifade edilmiştir. Bu doğrultuda, bazı katılımcılar, düzenli retrospektif toplantılar ve işbirliğine dayalı iletişim kanallarının güçlendirilmesi gerektiğini ifade etmişlerdir. Ayrıca, Slack gibi çevrimiçi iletişim araçlarının kullanımı ile anlık geri bildirimlerin artırılması ve ekip üyeleri arasındaki bilgi paylaşımının teşvik edilmesi önerilmiştir. Bu çıkarımda 1. ve 3. soru baz alınmıştır.

7.2. Teknolojik Entegrasyon Zorlukları

Mülakatlar sırasında, mevcut kurumsal yapıların ve eski teknolojik altyapıların DevOps süreçlerine entegre edilmesinde ciddi zorluklarla karşılaşıldığı ifade edilmiştir. Katılımcılar, eski versiyonlama sistemlerinin ve monolitik mimarilerin, DevOps'un getirdiği esneklik ve hızla uyumsuzluk gösterdiğini belirtmişlerdir. Bu bağlamda, mikroservis mimarisi ve konteyner teknolojilerinin (Docker, Kubernetes) kademeli olarak devreye alınması, eski sistemlerin modernizasyonunda kilit bir strateji olarak önerilmiştir. Ayrıca, Infrastructure as Code (IaC) yaklaşımlarının benimsenmesi ve altyapının otomatikleştirilmesi gerektiği vurgulanmıştır Bu çıkarımda 2. ve 4. soru baz alınmıştır.

7.3. Süreç Yönetimi ve Otomasyon

DevOps süreçlerinde otomasyon eksiklikleri ve manuel iş yüklerinin fazlalığı, projelerin teslim sürelerini olumsuz yönde etkilemektedir. Mülakat katılımcıları, özellikle sürekli entegrasyon ve sürekli teslimat (CI/CD) süreçlerinde yaşanan aksaklıklara dikkat

çekmişlerdir. Katılımcılar, bu eksikliklerin giderilmesi için otomasyonun artırılması gerektiğini ifade etmişlerdir. Sürekli entegrasyon ve teslimat pipeline'larının tamamen otomatik hale getirilmesi ve test süreçlerinin kapsamlı bir şekilde entegre edilmesi gerektiği vurgulanmıştır. Bununla birlikte, proaktif izleme ve log yönetimi araçlarının (Prometheus, Grafana, ELK Stack vb.) etkin kullanımının, sorunların erken teşhis edilmesi ve hızlı müdahale edilmesi açısından kritik önem taşıdığı belirtilmiştir. Bu çıkarımda 1. ve 5. soru baz alınmıştır.

7.4. Kültürel Değişim

DevOps'un organizasyonel bir değişim olduğu ve sadece teknik bir dönüşümden ibaret olmadığı, mülakatların bir diğer önemli bulgusudur. Katılımcılar, DevOps'a geçiş sürecinde kültürel dirençle karşılaştıklarını ve bu direncin, eski alışkanlıkların terk edilmesindeki zorluklardan kaynaklandığını dile getirmişlerdir. Bu noktada, katılımcılar, DevOps kültürünün organizasyonel yapıda benimsenmesi için eğitim programlarının ve farkındalık çalışmalarının kritik bir rol oynadığını ifade etmişlerdir. Ayrıca, çapraz fonksiyonel ekiplerin oluşturulmasının, geliştirici ve operasyon ekipleri arasındaki sinerjiyi artırdığı ve sorumluluk bilincini güçlendirdiği vurgulanmıştır. Bu çıkarımda 2. 6. ve 7. soru baz alınmıştır.

8. SONUÇ VE ÖNERİLER

Bu çalışma, DevOps uygulamalarında karşılaşılan zorlukları teknik, organizasyonel ve kültürel boyutlarda ele alarak, adaptasyon sürecini kolaylaştırmaya yönelik model önerileri geliştirmiştir. Savunma sanayiinde uygulanan senaryo analizleri, geliştirilen uyarlama modellerinin uygulanabilirliğini göstermiştir. Bulgular, DevOps'un teknik ve organizasyonel zorluklarını ele alarak yazılım geliştirme süreçlerinde verimliliği artıracak, özelleştirilmiş ve stratejik bir yol haritası sunmaktadır.

8.1. Teknik Düzeyde İyileştirme ve Otomasyon Stratejileri

DevOps uygulamalarında teknik zorluklar, özellikle sürekli entegrasyon ve sürekli teslimat (CI/CD) süreçlerinin eksiksiz bir şekilde uygulanamaması, manuel süreçlerin fazlalığı ve eski sistemlerin yeni teknolojilerle entegrasyonundan kaynaklanmaktadır. Katlılımcılar, CI/CD entegrasyonunun ve altyapı otomasyonunun iyileştirilmesi gerektiğini vurgulamıştır. Bu bağlamda, AI/ML tabanlı otomasyon araçlarının kullanımı önerilmektedir. Bu araçlar, hataların daha hızlı tespit edilmesine, sistem performansının artırılmasına ve proaktif sorun yönetimine olanak sağlayabilir. Ayrıca, katılımcılar, otomatik test süreçlerinin CI/CD "pipeline"larına entegrasyonunun yazılım kalitesini artıracağını ve sürekli entegrasyon süreçlerini destekleyeceğini belirtmiştir. Mikro hizmet mimarilerinin getirdiği karmaşıklık ve eski sistemlerden yeni sistemlere geçiş süreci, DevOps uygulamalarında karşılaşılan teknik engellerden diğeridir. Bu nedenle, "Infrastructure as Code" (IaC) yaklaşımlarının benimsenmesi ve konteyner teknolojilerinin etkin kullanımı, sistemlerin daha esnek ve ölçeklenebilir olmasını sağlayacaktır.

Teknik düzeyde, CI/CD entegrasyonunun ve altyapı otomasyonunun geliştirilmesi için AI/ML tabanlı araçlar önerilmektedir. Bu araçların, hataların daha hızlı tespit edilmesine ve sistem performansının artırılmasına yardımcı olabileceği değerlendirilmektedir. Otomasyonun doğru uygulanamaması, sürekli entegrasyon süreçlerini sekteye uğratabilmektedir. Özellikle sürekli test süreçleri zaman zaman göz ardı edilmekte, bu da yazılım kalitesini olumsuz etkilemektedir. Mikro hizmet mimarilerinin artan karmaşıklığı ve

eski sistemlerden yeni sistemlere geçiş sürecinin, organizasyonların DevOps uygulamalarını zorlaştıran faktörler olduğu değerlendirilmektedir.

8.2. Organizasyonel ve Kültürel Dönüşüm Stratejileri

DevOps'un organizasyonel düzeyde başarılı bir şekilde uygulanabilmesi, yalnızca teknik bir dönüşümden ibaret olmayıp, aynı zamanda organizasyonel kültürün de değişimini gerektirmektedir. Katılımcılar, ekipler arasında yaşanan iletişim kopukluklarının ve geleneksel çalışma biçimlerine olan direncin DevOps adaptasyonunu zorlaştırdığını ifade etmiştir. Organizasyonlar, DevOps kültürünü benimseyebilmek için liderlik stratejilerini yeniden gözden geçirmelidir. Bu kapsamda, ekipler arası bağımsızlığı destekleyen ve iletişimi artıran stratejilerin, organizasyonel dirençlerin aşılmasında kritik rol oynayacağı belirtilmektedir.

Mülakatlarda, organizasyonların DevOps'a geçiş sürecinde ekip üyelerinin DevOps kültürü ve araçları konusunda yeterli eğitimi almadıkları ifade edilmiştir. Eğitim eksikliği, sürecin etkin bir şekilde yönetilmesini ve ekiplerin iş birliği yapmasını engellemektedir. Bu bağlamda, düzenli eğitim programları, mentorluk ve farkındalık çalışmalarının, ekiplerin DevOps prensiplerini daha iyi anlamalarına ve bu kültürü benimsemelerine katkı sağlayacağı düşünülmektedir. Ekipler arasında iletişim kopukluğu ve geleneksel çalışma biçimlerine olan direnç, DevOps'u benimseme sürecinde en büyük zorluklardan biri olarak değerlendirilmektedir. Organizasyonel düzeyde, DevOps'un benimsenmesi sırasında karşılaşılan kültürel dirençleri aşmak için liderlik stratejileri kritik rol oynamaktadır. Ekipler arası bağımsızlığı destekleyen ve iletişimi artıran stratejiler, organizasyonel dirençleri aşmada etkili olabilmektedir.

8.3. Rol ve Sorumlulukların Belirlenmesi

DevOps uygulamaları, geliştirme ve operasyon ekipleri arasında iş birliğine dayalı bir çalışma gerektirmektedir. Ancak, mülakat bulguları, ekipler arası sahiplenme eksikliği ve rol belirsizliklerinin DevOps süreçlerinde sıkça karşılaşılan sorunlar olduğunu ortaya koymaktadır. Dağıtım ve sürüm süreçlerinde net bir sahiplik olmaması, gecikmelere ve

projelerin başarısızlıkla sonuçlanmasına neden olmaktadır. Bu doğrultuda, DevOps rollerinin ve sorumluluklarının açık bir şekilde tanımlanması, organizasyonel etkinliğin artırılması adına önemli bir adım olarak öne çıkmaktadır.

8.4. Güçlükler ve Çözüm Önerileri

- Teknik ve organizasyonel zorluklar, süreçlerin iyileştirilmesi için bütüncül bir yaklaşımla ele alınmalıdır.
- Ekipler arası iletişim eksikliklerini gidermek için düzenli toplantılar ve etkili iletişim araçları kullanılmalıdır.
- Kültürel dönüşüm için DevOps felsefesini destekleyen eğitimler ve farkındalık programları düzenlenmelidir.

8.4.1. İhtiyaç yönetimi

- Gereksinimlerin doğru bir şekilde belirlenmesi ve izlenebilirliğin sağlanması için kullanım durumu (use case) ve kullanıcı hikayesi (user story) gibi yöntemler kullanılmalıdır.
- İhtiyaçların önceliklendirilmesinde paydaş analizi ve maliyet-zaman değerlendirmesi yapılmalıdır.
- Gereksinim doğrulama ve geçerleme süreçleri formal yöntemlerle desteklenmelidir.

8.4.2. Takım yönetimi

- Takım içindeki roller ve sorumluluklar net bir şekilde tanımlanmalıdır.
- Takım gelişimi için sürekli eğitim programları ve mentorluk süreçleri oluşturulmalıdır.
- Uzaktan ve hibrit ekiplerin etkin yönetimi için uygun dijital araçlar kullanılmalıdır.

8.4.3. Risk yönetimi

- Risklerin sistematik bir şekilde tanımlanması ve kayıt altına alınması sağlanmalıdır.
- Kritik risklerin önceliklendirilmesi ve etkilerini azaltmaya yönelik stratejik planlar yapılmalıdır.
- Sürekli izleme ve değerlendirme ile risklerin etkisi kontrol altında tutulmalıdır.

8.4.4. Kalite yönetimi

- Kalite standartlarına uygun süreçlerin benimsenmesi ve bu standartların düzenli olarak değerlendirilmesi gerekmektedir.
- Yazılımın işlevsel ve işlevsel olmayan gereksinimlerini karşılamak için kalite metrikleri kullanılmalıdır.
- Paydaş memnuniyetini artırmak için kalite değerlendirme süreçleri etkinleştirilmelidir.

8.4.5. Yapay zekâ kullanımı

- Yapay zeka araçları, hata tespiti ve otomasyon süreçlerini desteklemek için etkin bir şekilde entegre edilmelidir.
- Yapay zeka destekli analizlerle risk yönetimi ve karar destek süreçleri güçlendirilmelidir.
- Yapay zeka uygulamalarının etik ve güvenlik standartlarına uygunluğu sağlanmalıdır.

Bu araştırma, DevOps uygulamalarında karşılaşılan teknik ve organizasyonel zorlukları kapsamlı bir şekilde ele almış ve bu zorlukların aşılmasına yönelik stratejik çözüm önerileri geliştirmiştir. Geliştirilen uyarlama modelleri, savunma sanayi bağlamında bir simülatör yazılımı üzerinde değerlendirilmiş ve süreçlerin uygulanabilirliği ortaya konulmuştur. Bulgular, DevOps süreçlerinin gerçek dünyadaki uygulamalarını optimize etmeye yönelik somut katkılar sunmakta ve yazılım geliştirme süreçlerinde verimliliği artıracak bir yol haritası sağlamaktadır.

Araştırma kapsamında gerçekleştirilen yapılandırılmış mülakatlar, DevOps süreçlerinde karşılaşılan zorlukları farklı perspektiflerden anlamaya yönelik zengin bir veri seti sunmuştur. Bununla birlikte, katılımcıların yaş dağılımına göre verilen yanıtlar arasında anlamlı bir bağlantı olmadığı gözlemlenmiştir. Katılımcıların yaş grupları farklılık gösterse de, DevOps'un uygulanmasında karşılaşılan teknik ve organizasyonel zorluklar ile bu zorluklara yönelik öneriler açısından benzer temalar öne çıkmıştır. Bu durum, DevOps ile ilgili zorlukların yaşa bağlı olmaksızın daha çok organizasyonel ve süreç odaklı olduğunu göstermektedir.

Gerçek hayat uygulamaları açısından bu çalışmanın bulguları, savunma sanayi gibi yüksek güvenlik ve hassasiyet gerektiren sektörlerde doğrudan uygulanabilecek önemli çıkarımlar ortaya koymuştur. Çalışma sonuçları, pratik anlamda somut uygulamalara ışık tutmaktadır. Araştırma, yalnızca sektörel gereksinimlere yönelik katkılar sağlamakla kalmayıp, aynı zamanda yazılım geliştirme süreçlerinde verimliliği artıracak stratejik bir çerçeve sunmaktadır.

KAYNAKLAR

Acevedo-Dueñas, D., Muñoz, M., & Galván-Cruz, S. (2023, October). Use of Devops in Very Small Entities: Systematic Mapping. *In 2023 12th International Conference On Software Process Improvement (CIMPS) (pp. 78-83).* IEEE.

Akbar, M. A., Mahmood, S., Shafiq, M., Alsanad, A., Alsanad, A. A. A., & Gumaei, A. (2023*). Identification and prioritization of DevOps success factors using fuzzy-AHP approach. Soft computing,* 1-25.

Akbar, M. A., Naveed, W., Mahmood, S., Alsanad, A. A., Alsanad, A., Gumaei, A., & Mateen, A. (2020). *Prioritization based taxonomy of DevOps challenges using fuzzy AHP analysis. IEEE Access, 8,* 202487-202507.

Akbar, M. A., Smolander, K., Mahmood, S., & Alsanad, A. (2022). *Toward successful DevSecOps in software development organizations: A decision-making framework. Information and Software Technology, 147,* 106894.

Alharbi, E. T., & Qureshi, M. R. J. (2014). *Implementation of risk management with SCRUM to achieve CMMI requirements. International Journal of Computer Network and Information Security, 6(11),* 20.

Almeida, F., Simões, J., & Lopes, S. (2022). *Exploring the benefits of combining DevOps and agile. Future Internet, 14(2),* 63.

Alnafessah, A., Gias, A. U., Wang, R., Zhu, L., Casale, G., & Filieri, A. (2021). *Quality-aware devops research: Where do we stand?. IEEE access, 9,* 44476-44489.

Amaro, R., Pereira, R., & da Silva, M. M. (2022). Capabilities and practices in DevOps: a multivocal literature review. *IEEE Transactions on Software Engineering, 49(2),* 883-901.

Amaro, R., Pereira, R., & da Silva, M. M. (2024). *Mapping DevOps capabilities to the software life cycle: A systematic literature review. Information and Software Technology,* 107583.

Ambler, S., & Lines, M. (2020, July). *Introduction to disciplined agile delivery*. Project Management Institute.

Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B. & Zimmermann T. (2019). *Software engineering for machine learning: A case study. Microsoft Research,* https://www.microsoft.com.

Antunes, P., & Tate, M. (2024). *"What's Going On" with BizDevOps: A qualitative review of BizDevOps practice. Computers in Industry, 157,* 104081.

Avritzer, A. (2020, March). *Challenges and approaches for the assessment of micro-service architecture deployment alternatives in DevOps: A tutorial presented at ICSA 2020. In 2020 IEEE International Conference on Software Architecture Companion (ICSA-C) (pp. 1-2).* IEEE.

Azad, N., & Hyrynsalmi, S. (2023*). DevOps critical success factors—A systematic literature review. Information and Software Technology, 157,* 107150.

Azuma, M. (2011, September). *The impact of ICT evolution and application explosion on software quality: a solution by ISO/IEC 250nn square series of standards. In Proceedings of the 8th international workshop on Software quality* (pp. 1-2).

Banica, L., Radulescu, M., Rosca, D., & Hagiu, A. (2017). *Is DevOps another project management methodology?. Informatica Economica,* 21(3).

Battina, D. S. (2019). *An intelligent devops platform research and design based on machine learning. training,* 6(3).

Battina, D. S. (2020). Devops, *A New Approach To Cloud Development & Testing. International Journal of Emerging Technologies and Innovative Research (www. jetir. org),* ISSN, 2349-5162.

Battina, D. S. (2021). The Challenges and Mitigation Strategies of Using DevOps during Software Development. *International Journal of Creative Research Thoughts (IJCRT),* ISSN, 2320-2882.

Bijwe, A., & Shankar, P. (2022, October). Challenges of adopting DevOps culture on the internet of things applications-A solution model. *In 2022 2nd International conference on technological advancements in computational sciences (ICTACS)* (pp. 638-645). IEEE.

Boehm, B. W., & DeMarco, T. (1997). *Software risk management. IEEE software, 14(3),* 17.

Bollieddula, G. (2022). *Challenges and Solutions in the Implementation of DevOps Tools & Security (DevSecOps): A Systematic Review*.

Bolscher, R., & Daneva, M. (2019, January). Designing software architecture to support continuous delivery and DevOps: a systematic literature review. *In 14th International Conference on Software Technologies, ICSOFT 2019 (pp. 27-39)*. SCITEPRESS.

Boss, R. W. (1983). Team building and the problem of regression: The personal management interview as an intervention. *The Journal of Applied Behavioral Science, 19(1),* 67-83.

Bourque P. and Richard E. (2014). SWEBOK Version 3.0. IEEE, ISBN-10: 0-7695-5166-1.

Carturan, S., & Goya, D. (2019, May). Major Challenges of Systems-of-Systems with Cloud and DevOps–a financial experience report. *In 2019 IEEE/ACM 7th International Workshop on Software Engineering for Systems-of-Systems (SESoS) and 13th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (WDES) (pp. 10-17).* IEEE.

Cervone, H. F. (2007). Standard methodology in digital library project management. *OCLC Systems & Services: International digital library perspectives, 23(1),* 30-34.

Chaouch, S., Mejri, A., & Ghannouchi, S. A. (2019). A framework for risk management in Scrum development process. *Procedia Computer Science, 164,* 187-192.

Chiari, M., Xiang, B., Nedeltcheva, G. N., Di Nitto, E., Blasi, L., Benedetto, D., & Niculut, L. (2023, June). DOML: a new modelling approach to infrastructure-as-code. *In International Conference on Advanced Information Systems Engineering (pp. 297-313).* Cham: Springer Nature Switzerland.

Cuadra, J., Hurtado, E., Sarachaga, I., Estévez, E., Casquero, O., & Armentia, A. (2024). *Enabling DevOps for Fog Applications in the Smart Manufacturing domain: A Model-Driven based Platform Engineering approach. Future Generation Computer Systems, 157*, 360-375.

Díaz, J., Pérez, J., Alves, I., Kon, F., Leite, L., Meirelles, P., & Rocha, C. (2024). *Harmonizing DevOps taxonomies A grounded theory study. Journal of Systems and Software, 208, 111908*.

El Aouni, F., Moumane, K., Idri, A., Najib, M., & Jan, S. U. (2024). A systematic literature review on Agile, Cloud, and DevOps integration: Challenges, benefits. *Information and Software Technology,* 107569.

Eramo, R., Said, B., Oriol, M., Bruneliere, H., & Morales, S. (2024). An architecture for model-based and intelligent automation in DevOps. *Journal of Systems and Software, 217,* 112180.

Eramo, R., Tucci, M., Di Pompeo, D., Cortellessa, V., Di Marco, A., & Taibi, D. (2024). *Architectural support for software performance in continuous software engineering: A systematic mapping study. Journal of Systems and Software, 207,* 111833.

Esaki, K., Azuma, M., & Komiyama, T. (2013). Introduction of quality requirement and evaluation based on ISO/IEC square series of standard. *In Trustworthy Computing and Services: International Conference, ISCTCS 2012, Beijing, China, May 28–June 2, 2012, Revised Selected Papers (pp. 94-101).* Springer Berlin Heidelberg.

Faustino, J., Adriano, D., Amaro, R., Pereira, R., & da Silva, M. M. (2022). *DevOps benefits: A systematic literature review. Software: Practice and Experience, 52(9),* 1905-1926.

Fawzy, A. H., Wassif, K., & Moussa, H. (2023). *Framework for automatic detection of anomalies in DevOps. Journal of King Saud University-Computer and Information Sciences, 35(3),* 8-19.

Fernandes, M., Ferino, S., Fernandes, A., Kulesza, U., Aranha, E., & Treude, C. (2022, May). Devops education: An interview study of challenges and recommendations. *In Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Software Engineering Education and Training* (pp. 90-101).

Fernandes, M., Ferino, S., Kulesza, U., & Aranha, E. (2020, October). Challenges and recommendations in DevOps education: A systematic literature review. *In Proceedings of the XXXIV Brazilian Symposium on Software Engineering* (pp. 648-657).

Firdaouss, L., Ayoub, B., Manal, B., & Ikrame, Y. (2022). Automated VPN configuration using DevOps. *Procedia Computer Science, 198,* 632-637.

G. Giray, A software engineering perspective on engineering machine learning systems: State of the art and challenges, *The Journal of Systems & Software, 180,* 1-35, (2021).

Gashaw, M. T., & Beshah, T. (2022). *DevOps Implementation Practices and Challenges in the Ethiopian Software Development Industries* June 2022.

Giamattei, L., Guerriero, A., Pietrantuono, R., Russo, S., Malavolta, I., Islam, T., ... & Panojo, F. S. (2023). Monitoring tools for DevOps and microservices: A systematic grey literature review. *Journal of Systems and Software,* 111906.

Gokarna, M., & Singh, R. (2021, February). DevOps: a historical review and future works. *In 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)* (pp. 366-371). IEEE.

Grande, R., Vizcaíno, A., & García, F. O. (2024). Is it worth adopting DevOps practices in Global Software Engineering? Possible challenges and benefits. *Computer Standards & Interfaces, 87,* 103767.

Guest G., MacQueen K.K. ve Namey E.E. (2012). Applied Thematic Analysis, SAGE, USA.

Hammad, M., & Inayat, I. (2018, December). Integrating risk management in scrum framework. *In 2018 International Conference on Frontiers of Information Technology (FIT) (pp. 158-163).* IEEE.

Hamza, U., Sharifah, M. S. M., & Lee, N. (2023). *DevOps Adoption Guidelines, Challenges, and Benefits: A Systematic Literature Review. ICRRD Journal, (1),* 2773-5958.

Hasselbring, W., Henning, S., Latte, B., Möbius, A., Richter, T., Schalk, S., & Wojcieszak, M. (2019, March). Industrial devops. *In 2019 IEEE International Conference on Software Architecture Companion (ICSA-C) (pp. 123-126).* IEEE.

Hemon-Hildgen, A., Rowe, F., & Monnier-Senicourt, L. (2020). *Orchestrating automation and sharing in DevOps teams: a revelatory case of job satisfaction factors, risk and work conditions. European journal of information systems, 29(5),* 474-499.

Hossain, E., Babar, M. A., Paik, H. Y., & Verner, J. (2009, December). Risk identification and mitigation processes for using scrum in global software development: A conceptual framework. *In 2009 16th Asia-Pacific Software Engineering Conference (pp. 457-464).* IEEE.

Hrusto, A., Engström, E., & Runeson, P. (2023). Towards optimization of anomaly detection in DevOps. *Information and Software Technology, 160,* 107241.

Ibrahim, M. M. A., Syed-Mohamad, S. M., & Husin, M. H. (2019, February). Managing quality assurance challenges of DevOps through analytics. *In Proceedings of the 2019 8th International Conference on Software and Computer Applications* (pp. 194-198).

Ishikawa F., Yoshioka N. (2019). How do engineers perceive difficulties in engineering of machine-learning systems? *In Proceedings of IEEE/ACM 6th International Workshop on Software Engineering Research and Industrial Practice. May 28, 2019, Montreal, QC,* Canada.

ISO/IEC/IEEE 29148:2018 (2018). *Systems and software engineering-life cycle processes-Requirements Engineering*, Standard.

ISO/IEC/IEEE 42010:2011 (2011). Systems and software engineering-architecture description, standard, *https://www.iso.org/ obp/ ui/ # iso: std:iso-iec-ieee:42010:ed-1:v1:en.*

Jayakody, J. A. V. M. K., & Wijayanayake, W. M. J. I. (2021, September). Challenges for adopting DevOps in information technology projects. *In 2021 International Research Conference on Smart Computing and Systems Engineering (SCSE) (*Vol. 4, pp. 203-210). IEEE.

Jayakody, J. A. V. M. K., & Wijayanayake, W. M. J. I. (2022). DevOps Adoption in Information Systems Projects; A Systematic Literature Review. *International Journal of Software Engineering & Applications, 13(3),* 39-53.

Jayakody, J. A. V. M. K., & Wijayanayake, W. M. J. I. (2023, June). Process Improvement Framework for DevOps Adoption in Software Development. *In 2023 International Research Conference on Smart Computing and Systems Engineering (SCSE) (Vol. 6, pp. 1-7).* IEEE.

Jha, A. V., Teri, R., Verma, S., Tarafder, S., Bhowmik, W., Kumar Mishra, S., ... & Philibert, N. (2023). *From theory to practice: Understanding DevOps culture and mindset. Cogent Engineering, 10(1),* 2251758.

Karunarathne, M. A. W., Wijayanayake, W. M. J. I., & Prasadika, A. P. K. J. (2024, February). DevOps Adoption in Software Development Organizations: A Systematic Literature Review. *In 2024 4th International Conference on Advanced Research in Computing (ICARC) (pp. 282-287).* IEEE.

Khan, A. A., & Shameem, M. (2020). Multicriteria decision-making taxonomy for DevOps challenging factors using analytical hierarchy process. *Journal of software: evolution and process, 32(10),* e2263.

Khan, M. S., Khan, A. W., Khan, F., Khan, M. A., & Whangbo, T. K. (2022). Critical challenges to adopt DevOps culture in software organizations: *A systematic review. Ieee Access, 10,* 14339-14349.

Khattak, K. N., Qayyum, F., Naqvi, S. S. A., Mehmood, A., & Kim, J. (2023). *A Systematic Framework for Addressing Critical Challenges in Adopting DevOps Culture in Software Development: A PLS-SEM Perspective.* IEEE Access.

Komiyama, T., Fukuzumi, S. I., Azuma, M., Washizaki, H., & Tsuda, N. (2020). Usability of software–intensive systems from developers' point of view: Current status and future perspectives of international standardization of usability evaluation. In Human-Computer Interaction. *Design and User Experience: Thematic Area, HCI 2020, Held as Part of the 22nd International Conference, HCII 2020, Copenhagen, Denmark, July 19–24, 2020,* Proceedings, Part I 22 (pp. 450-463). Springer International Publishing.

Koren, I., Rinker, F., Meixner, K., Kröger, M., & Zeng, M. (2023, September). Implementing DevOps Practices in CPPS Using Microservices and GitOps. *In 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)* (pp. 1-4). IEEE.

Koren, I., Rinker, F., Meixner, K., Matevska, J., & Walter, J. (2023, May). Challenges and opportunities of devops in cyber-physical production systems engineering. *In 2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS)* (pp. 1-6). IEEE.

Krey, M. (2022). Devops adoption: challenges & barriers.

Kumeno F. (2019). Software engineering challenges for machine learning applications: A literature review". *Intell. Decis. Technol. 13 (4),* 463-476.

Laukkarinen, T., Kuusinen, K., & Mikkonen, T. (2017, May). DevOps in regulated software development: case medical devices. *In 2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER) (pp. 15-18).* IEEE.

Laukkarinen, T., Kuusinen, K., & Mikkonen, T. (2018). Regulated software meets DevOps. *Information and Software Technology, 97,* 176-178.

Lee, R.Y. (2019). *Object-oriented software engineering with UML: A hands-on approach, Nova Science Publishers,* USA.

Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2019). A survey of DevOps concepts and challenges. *ACM Computing Surveys (CSUR), 52(6),* 1-35.

Lunesu, M. I., Tonelli, R., Marchesi, L., & Marchesi, M. (2021). Assessing the risk of software development in agile methodologies using simulation. *IEEE Access, 9,* 134240-134258.

Luz, W. P., Pinto, G., & Bonifácio, R. (2019). Adopting DevOps in the real world: A theory, a model, and a case study. *Journal of Systems and Software, 157,* 110384.

Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., ... & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and software technology, 114,* 217-230.

Macarthy, R. W., & Bass, J. M. (2020, August). An empirical taxonomy of DevOps in practice. *In 2020 46th euromicro conference on software engineering and advanced applications (seaa) (pp. 221-228).* IEEE.

Marnewick, C., & Langerman, J. (2020). DevOps and organizational performance: the fallacy of chasing maturity. *IEEE Software, 38(5),* 48-55.

Maroukian, K., & Gulliver, S. R. (2020). Leading DevOps practice and principle adoption. *arXiv preprint arXiv:2008.*10515.

Mishra, A., & Otaiwi, Z. (2020). DevOps and software quality: A systematic mapping. *Computer Science Review, 38,* 100308.

Mousaei, M. A. H. D. I., & Gandomani, T. J. (2020). DevOps Approach and Lean Thinking in Agile Software Development: Opportunities, *Advantages, and Challenges. J. Soft Eng. Int. Syst,* 5, 1-10.

Mousaei, M., & Gandomani, T. J. (2018). A new project risk management model based on Scrum framework and Prince2 methodology. *International Journal of Advanced Computer Science and Applications*, 9(4).

Muñoz, M., Negrete, M., & Mejía, J. (2019). Proposal to avoid issues in the devops implementation: A systematic literature review. *New Knowledge in Information Systems and Technologies: Volume 1,* 666-677.

Muyet, Z., & Sahibuddin, S. (2022). Analysing the factors influencing DevOps adoption in the existing DevOps adoption models: A systematic review. *Open International Journal of Informatics, 10(2),* 97-107.

Nayanajith, A., & Wickramarachchi, R. (2024, February). Challenges Affecting the Successful Adoption of DevOps Practices: A Systematic Literature Review. *In 2024 4th International Conference on Advanced Research in Computing (ICARC)* (pp. 311-315). IEEE.

Okubo, T., & Kaiya, H. (2022). Efficient secure DevOps using process mining and Attack Defense Trees. *Procedia Computer Science, 207,* 446-455.

Österberg, G. (2020). *A Systematic Literature Review on DevOps and its Definitions, Adoptions,* Benefits and Challenges.

Patel, A. R., & Tyagi, S. (2022, April). Lightweight Review: Challenges and Benefits of Adopting DevOps. *In 2022 1st International Conference on Informatics (ICI) (pp. 235-237).* IEEE.

Patton, M. Q. (2015). *Qualitative research and evaluation methods: Integrating theory and practice* (4th ed.). Sage Publications.

Pérez-Sánchez, J., Rafi, S., de Gea, J. M. C., Ros, J. N., & Alemán, J. L. F. (2024). *A theory on human factors in DevOps adoption. Computer Standards & Interfaces,* 103907.

Plant, O. H., Aldea, A., & van Hillegersberg, J. (2024). *Improving DevOps team performance through context-capability coalignment: Towards a profile for public sector organizations. Information and Software Technology,* 107585.

Poston, R. S., & Richardson, S. M. (2012). Designing an academic project management program: A collaboration between a university and a PMI chapter. *Journal of Information Systems Education, 22(1),* 55-72.

Rafi, S., Yu, W., Akbar, M. A., Alsanad, A., & Gumaei, A. (2020). Multicriteria based decision making of DevOps data quality assessment challenges using fuzzy TOPSIS. *IEEE Access, 8, 4*6958-46980.

Rafi, S., Yu, W., Akbar, M. A., Mahmood, S., Alsanad, A., & Gumaei, A. (2021). Readiness model for DevOps implementation in software organizations. *Journal of Software: Evolution and Process, 33(4),* e2323.

Raj, P., & Sinha, P. (2015). Project management in era of agile and devops methodlogies. *In International Conference on Applied Sciences* (Vol. 9, No. 1, pp. 1024-1033).

S. Nalchigar, E. Yu & K. Keshavjee, *Modeling machine learning requirements from three perspectives: A case report from the healthcare domain, Requirements Engineering, 26,* 237-254, (2021).

Sinha, R., Shameem, M., & Kumar, C. (2020, February). SWOT: Strength, weaknesses, opportunities, and threats for scaling agile methods in global software development. *In Proceedings of the 13th Innovations in Software Engineering Conference (formerly known as India Software Engineering Conference)* (pp. 1-10).

Sravan, S. S., Ganesh, C. S., Kiran, K. V. D., Chandra, T. A., Aparna, K., & Vignesh, T. (2023, March). Significant Challenges to espouse DevOps Culture in Software Organisations By AWS: A methodical Review. *In 2023 9th International conference on advanced computing and communication systems (ICACCS) (Vol. 1, pp. 395-401).* IEEE.

Sriraman, G., & Shriram, R. (2024). Slide-block: End-to-end amplified security to improve DevOps resilience through pattern-based authentication. Heliyon, 10(4).

Tamanampudi, V. M. (2019). Automating CI/CD Pipelines with Machine Learning Algorithms: Optimizing Build and Deployment Processes in DevOps Ecosystems. Distributed Learning and Broad Applications in Scientific Research, 5, 810-849.

Tatineni, S. (2023). Applying DevOps Practices for Quality and Reliability Improvement in Cloud-Based Systems. *Technix international journal for engineering research (TIJER), 10(11),* 374-380.

Tavares, B. G., da Silva, C. E. S., & de Souza, A. D. (2019). *Risk management analysis in Scrum software projects. International Transactions in Operational Research, 26(5),* 1884-1905.

Toh, M. Z., Sahibuddin, S., & Bakar, R. A. (2021, August). A Review on DevOps Adoption in Continuous Delivery Process. *In 2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)* (pp. 98-103). IEEE.

Toh, M. Z., Sahibuddin, S., & Mahrin, M. N. R. (2019, February). Adoption issues in DevOps from the perspective of continuous delivery pipeline. *In Proceedings of the 2019 8th international conference on software and computer applications (*pp. 173-177).

Unhelkar B. (2018). Software engineering with UML, CRC Press, *Taylor & Francis Group, USA*.

Uysal M. P. (2021). Machine learning and data science project management from an agile perspective: Methods and challenges. In:Vannie Naidoo, Rahul Verma editors. *Contemporary challenges for agile project management. USA: IGI Global,* 73-89.

Uysal M. P. (2022b). An integrated and multi-perspective approach to the requirements of machine learning. *IFIP International Conference on Industrial Information Integration (ICIIIE 2022), December 10, 2022,* Bangkok, Thailand.

Uysal M. P., & Mergen, A. E. (2015). *Yazılım Yeniden Yapılamaya Yönelik Model Güdümlü ve Kaliteye Yönelimli Süreç Modeli.* In UYMS.

Uysal, M. P. (2022a). Machine learning-enabled healthcare information systems in view of Industrial Information Integration Engineering. *Journal of Industrial Information Integration, 30(1),* 1-18.

Vemuri, N., Thaneeru, N., & Tatikonda, V. M. (2024). AI-Optimized DevOps for Streamlined Cloud CI/CD. *International Journal of Innovative Science and Research Technology, 9(7),* 10-5281.

Villamizar H., Escovedo T., and Kalinowski M. (2021). Requirements engineering for machine learning: A systematic mapping study. *47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), September 1-3,* 2021, Palermo, Italy.

Wiedemann, A., Wiesche, M., & Krcmar, H. (2019, June). Integrating development and operations in cross-functional teams-toward a devops competency model. In Proceedings of the 2019 on Computers and People Research Conference (pp. 14-19).

Yang, D., Wang, D., Yang, D., Dong, Q., Wang, Y., Zhou, H., & Hong, D. (2020). DevOps in practice for education management information system at ECNU. *Procedia Computer Science, 176,* 1382-1391.

Zhao, X., Clear, T., & Lal, R. (2024). Identifying the primary dimensions of DevSecOps: A multi-vocal literature review. Journal of Systems and Software, 112063.Zarour, M., Alhammad, N., Alenezi, M., & Alsarayrah, K. (2019). *A research on DevOps maturity models. Int. J. Recent Technol. Eng, 8(3),* 4854-4862.

EKLER

EK 1: Yarı Yapılandırılmış Mülakat Soruları

NU	SORULAR	AMAÇ
1	DevOps Proje Yönetim Çerçevesi doğrultusunda karşılaştığınız **ana sorunlar ve ana problem sahaları** nelerdir? Bu kapsamda **çözüme yönelik** kullandığınız ve/veya önerdiğiniz yöntem, teknik ve araçlar nelerdir?	(a) İlgili kurumdaki ana problem sahalarını ve kullanıldıkları çözümleri belirlemek (b) Geliştirilecek DevOps uygulama ve entegrasyon modeline temel oluşturmak
2	DevOps proje yönetim çerçevesi süreçlerinin ilgili yazılım **projelerine** (kurumsal/standart yazılım, görev-kritik, güvenlik-kritik sistem yazılımları), (b) **organizasyona** ve (c) **proje takımına uyarlarken** karşılaştığınız güçlükler ve kullandığınız çözümler nelerdir?	Kurumda mevcut haliyle kullanılan DevOps proje yönetim çerçevesinin projeye, organizasyona ve proje takımına uyarlanmasıyla ilgili durumların belirlenmesi
3	**Proje Takımı Yönetimi** kapsamında DevOps proje yönetim çerçevesi uygularken takım yönetimiyle ilgili konularda nelere dikkat ediyorsunuz? Karşılaştığınız sorunlara çözüm amacıyla kullandığınız yöntem ve teknikler nelerdir (**takım kültürü, takımın oluşturulması, rollerin belirlenmesi, eğitim vb**.)	Proje Takım Yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.
4	DevOps proje yönetim çerçevesini uygularken **İhtiyaç Mühendisliği** ve **Yazılım Yönetimi** kapsamında karşılaştığınız problemler ve bunlara yönelik kullandığınız çözümler nelerdir? (İhtiyaç mühendisliği: Paydaş yönetimi, ihtiyaçların belirlenmesi, analizi ve önceliklendirilmesi, ihtiyaçların izlenmesi, ihtiyaçların doğrulanması ve geçerlemesi vb.konular)	İhtiyaç Mühendisliği ve Yazılım Yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.
5	DevOps proje yönetim çerçevesini uygularken **risk yönetimini** nasıl gerçekleştiriyorsunuz ve çeşitli riskleri (proje yönetimi riskleri, teknik riskler, kurumsal riskler ve dış kaynaklı riskler) nasıl yönetiyorsunuz, kullandığınız yöntem, teknik ve araçlar nelerdir? (Risklerin Belirlenmesi ve Tanımlanması, Risklerin Analizi ve Değerlendirilmesi, Risklerin Önceliklendirilmesi, Risklerin Planlaması ve Önlemler, Risklerin İzlenmesi ve Kontrol Edilmesi)	Risk yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.
6	DevOps proje yönetim çerçevesini uygularken **ürün ve proje süreçlerinin kalite yönetimini** nasıl gerçekleştiriyorsunuz, kullandığınız yöntem, teknik ve araçlar nelerdir?	Kalite yönetimi kapsamında yapılan çalışmaları, sorunları ve çözüm önerilerini belirlemektir.
7	DevOps proje yönetim süreçlerinin, araçlarının ve ilkelerinin mevcut kurumsal proje yönetimi yaklaşımlarına, teknik alt yapıya, proje takıma **entegrasyonunda** karşılaştığınız güçlükler, kullandığınız yöntem ve araçlar ile çözüm önerileriniz nelerdir?	DevOps proje yönetim süreçlerinin entegrasyonuyla ilgili sorunları ve çözüm önerilerini belirlemektir.
8	DevOps proje yönetim çerçevesinin yapay zekâ yazılımlarının (makine öğrenmesi, derin öğrenme vb.) geliştirilmesine yönelik görüşleriniz nelerdir?	DevOps proje yönetim çerçevesinin yapay zekâ yazımlarının geliştirilmesine yönelik görüşlerin belirlenmesi
9.	Günümüzde LLM vb. **yapay zeka** yazılımları **yazılım geliştirme süreçlerine entegre edilmekte, çeşitli işlevlerin gerçekleştirilmesi, yazılımın geliştirilmesi** vb. amaçlarla kullanılmaktadır. Bu konuyla ilgili görüşleriniz nelerdir? Gerek genel olarak ve gerekse kendi kurumunuzda kullanımlarına yönelik neler önerebilirsiniz?	Yapay zekâ yazılımlarının yazılım mühendisliği süreçlerinde kullanımıyla ilgili sorunları ve çözüm önerilerini belirlemektir.