DÜZCE ÜNİVERSİTESİ | LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YAPAY ZEKA DESTEKLİ BÜYÜK DİL MODELİ İLE KODSUZ WEB GELİŞTİRME PLATFORMU TASARIMI VE UYGULAMASI

OSMAN KOÇ

YÜKSEK LİSANS TEZİ

ELEKTRİK-ELEKTRONİK VE BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN
PROF. DR. İBRAHİM YÜCEDAĞ

EŞ DANIŞMAN
DR. ÖĞR. ÜYESİ ÜMİT ŞENTÜRK

DÜZCE, 2025

T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YAPAY ZEKA DESTEKLİ BÜYÜK DİL MODELİ İLE KODSUZ WEB GELİŞTİRME PLATFORMU TASARIMI VE UYGULAMASI

Osman KOÇ tarafından hazırlanan tez çalışması aşağıdaki jüri tarafından Düzce Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. İbrahim YÜCEDAĞ

Düzce Üniversitesi

Eş Danışman

Dr. Öğr. Üyesi Ümit ŞENTÜRK

Bolu Abant İzzet Baysal Üniversitesi

Jüri Üyeleri

Prof. Dr. İbrahim YÜCEDAĞ
Düzce Üniversitesi _____

Prof. Dr. İbrahim Alper DOĞRU
Gazi Üniversitesi _____

Doç. Dr. Abdullah Talha KABAKUŞ
Düzce Üniversitesi _____

Tez Savunma Tarihi: 13/01/2025

BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

13 Ocak 2025

Osman KOÇ

TEŞEKKÜR

Bu tezin hazırlanmasında emeği geçen birçok değerli insanın katkılarını ifade etmeden geçemem. Öncelikle, bilgi ve tecrübeleriyle bu çalışmanın her aşamasında rehberlik eden danışman hocam Prof. Dr. İbrahim YÜCEDAĞ'a sonsuz teşekkür ederim. Kendisi, akademik çalışmalarıma yön verirken verdiği değerli öneriler ve sabırlı yaklaşımıyla bana ilham kaynağı olmuştur.

Eş danışmanım Dr. Öğr. Üyesi Ümit ŞENTÜRK'e, tez çalışmamın detaylarına gösterdiği titizlik, sağladığı kıymetli katkılar ve her zaman sunduğu teşvik edici destek için gönülden teşekkür ederim. Çalışmalarımı daha da ileriye taşımamda kendisinin katkıları yadsınamaz bir yere sahiptir.

Bu süreç boyunca yanımda olan sevgili eşime ise en derin şükranlarımı sunarım. Onun sabrı, desteği ve her zaman hissettirdiği anlayış olmasaydı, bu çalışmayı tamamlamak benim için çok daha zor olurdu. Zor zamanlarda yanımda olması ve bana güç vermesi, bu çalışmayı mümkün kılan temel unsurlardan biri olmuştur.

Bu vesileyle, emeği geçen herkese en içten teşekkürlerimi sunarım.

13 Ocak 2025 **Osman KOÇ**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

ÇİZELGE LİSTESİ

Sayfa No

KISALTMALAR

ADASYN	Adaptive Synthetic Sampling
AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
BDM	Büyük Dil Modeleri
BERT	Bidirectional Encoder Representations from Transformers
DDİ	Doğal Dil İşleme
DL	Deep Learning
GenAI	Generative Artificial Intelligence
GPT	Generative Pre-trained Transformer
HF	Hugging Face
KOBİ	Küçük ve Orta Büyüklükteki İşletmeler
LC	Low Code
Llama	Large Language Model Meta AI
LLM	Large Language Model
NC	No Code
NLP	Natural Language Processing
RAG	Retrieval Augmented Generation
RL	Reinforcement Learning
SMOTE	Synthetic Minority Oversampling Technique
ULMFiT	Universal Language Model Fine-Tuning
UX	User Experience
ÜYZ	Üretken Yapay Zeka
YSA	Yapay Sinir Ağları
YZ	Yapay Zeka
ZOA	Zero Operation Assistance

SİMGELER

m	Dizideki toplam kelime sayısı
P	Olasılık
w	Kelime (word)

ÖZET

YAPAY ZEKA DESTEKLİ BÜYÜK DİL MODELİ İLE KODSUZ WEB GELİŞTİRME PLATFORMU TASARIMI VE UYGULAMASI

Osman KOÇ

Düzce Üniversitesi
Lisansüstü Eğitim Enstitüsü, Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi
Danışman: Prof. Dr. İbrahim YÜCEDAĞ
Eş Danışman: Dr. Öğr. Üyesi Ümit ŞENTÜRK

Ocak 2025, 71 sayfa

Bu tez çalışması, Yapay Zeka (YZ, Artificial Intelligence – AI) destekli dil modellerinin kodsuz yazılım geliştirme süreçlerindeki potansiyelini ele almakta ve bu alandaki inovatif bir yaklaşımı temsil eden ZOA platformunun geliştirilmesi ve değerlendirilmesini kapsamaktadır. ZOA, kullanıcılara doğal dil girdileriyle modern web bileşenleri oluşturma imkânı sunan bir kodsuz platform olarak tasarlanmıştır. Platform, özellikle HTML, CSS ve JavaScript tabanlı çıktılar üretmek için Doğal Dil İşleme (DDİ, Natural Language Processing – NLP) tekniklerinden faydalanmaktadır. Araştırmanın ilk aşamasında, literatürde mevcut kodsuz platformların avantajları ve sınırlılıkları incelenmiş, bu platformların yazılım geliştirme süreçlerine etkileri detaylandırılmıştır. Yapılan literatür taramasında, YZ destekli kodsuz platformların maliyet ve zaman açısından önemli avantajlar sağladığı ancak kullanıcı deneyimi ve ölçeklenebilirlik gibi alanlarda iyileştirme potansiyeli bulunduğu tespit edilmiştir. ZOA'nın geliştirilme sürecinde, Python tabanlı bir API ve React teknolojisi ile oluşturulmuş bir kullanıcı arayüzü entegre edilmiştir. Model, Tailwind CSS kullanılarak derlenen geniş bir HTML ve CSS veri seti ile eğitilmiş ve kullanıcı girdilerini anlamlandırarak dinamik çıktılar üretebilecek şekilde optimize edilmiştir. Eğitim sürecinde, veri setinin çeşitliliği ve modelin performansına etkisi detaylı bir şekilde analiz edilmiştir. Sonuç olarak, ZOA'nın performansı çeşitli test senaryolarıyla değerlendirilmiş ve platformun kullanıcı dostu yapısı ile geliştirme süreçlerini demokratikleştirme potansiyeline sahip olduğu ortaya konulmuştur. Çalışmanın bulguları, kodsuz platformların gelecekteki geliştirme süreçlerinde YZ ve DDİ entegrasyonunun önemini vurgulamaktadır. Ayrıca, bu çalışma ZOA'nın başarımını artırmak için kullanılabilecek farklı veri setleri ve optimizasyon yöntemleri üzerine öneriler sunmaktadır.

Anahtar Sözcükler: Kodsuz, Yapay Zeka, Doğal Dil İşleme, Web Geliştirme, Dil Modeli

ABSTRACT

DESIGN AND IMPLEMENTATION OF A NO-CODE WEB DEVELOPMENT PLATFORM WITH AN AI-SUPPORTED LARGE LANGUAGE MODEL

Osman KOÇ
Düzce University
Graduate School, Department of Electrical, Electronics, and Computer Engineering
Master's Thesis
Supervisor: Prof. Dr. İbrahim YÜCEDAĞ
Co-supervisor: Asst. Prof. Ümit ŞENTÜRK

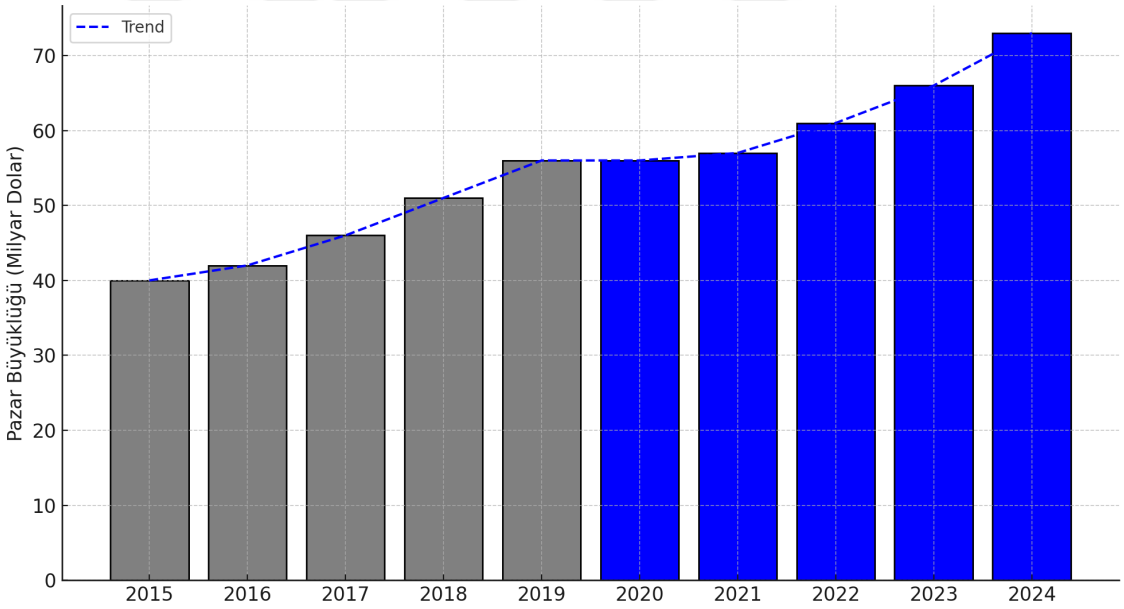
January 2025, 71 pages

This thesis examines the potential of AI-supported language models in no-code software development processes and focuses on the development and evaluation of the ZOA platform, which represents an innovative approach in this field. ZOA is designed as a no-code platform that allows users to create modern web components using natural language inputs. The platform particularly leverages Natural Language Processing (NLP) techniques to generate HTML, CSS, and JavaScript-based outputs. In the initial phase of the research, the advantages and limitations of existing no-code platforms were analyzed, and their impacts on software development processes were detailed. The literature review identified that AI-supported no-code platforms provide significant cost and time advantages while revealing potential areas for improvement, such as user experience and scalability. During the development of ZOA, a Python-based API and a React-powered user interface were integrated. The model was trained with an extensive HTML and CSS dataset compiled using Tailwind CSS and optimized to interpret user inputs and produce dynamic outputs. The diversity of the dataset and its impact on the model's performance were thoroughly analyzed during the training process. In conclusion, the performance of ZOA was evaluated through various test scenarios, demonstrating its user-friendly structure and potential to democratize development processes. The findings highlight the importance of integrating AI and NLP in future no-code platform development processes. Furthermore, this study provides recommendations for improving ZOA's performance through alternative datasets and optimization techniques.

Keywords: No-Code, Artificial Intelligence, Natural Language Processing, Web Development, Language Model

1. GİRİŞ

Yazılım geliştirme dünyası, sürekli evrilen ve dönüşen bir ekosistem olarak, günümüz iş dünyasında her sektörde kaçınılmaz bir gereklilik haline gelmiştir. Özellikle dijital dönüşüm süreçleri, işletmelerin yazılım ihtiyaçlarını hızla artırmaktadır. Bu ihtiyaçla doğru orantılı olarak, yazılım sektörü 2019 yılında 565 milyar dolarlık büyüklüğe ulaşmış ve katlanarak devam etmektedir; büyüme grafiği *Şekil 1.1*'de gösterilmektedir [1]. Son yıllarda yaşanan bulut bilişim altyapılarının yaygınlaşması ve pazar yerleri ile sınır ötesi müşterilere ulaşma imkânı gibi durumlar da buradaki büyümeyi tetiklemiştir.



Şekil 1.1. 2015 – 2024 yılları arasında dünyadaki yazılım sektörünün pazar büyüklüğü (milyar dolar).

Sektörün hızla büyümesi ve birçok alana katkı sağlaması, yazılım maliyetlerinin önemli ölçüde artmasına yol açtı. Yazılımların kaynak gereksinimleri, lisans ücretleri, geliştirme, destek ve bakım maliyetleri gibi birçok gider kalemi, özellikle küçük ve orta ölçekli işletmeler (KOBİ'ler) için büyük bir mali yük oluşturdu. Bu bağlamda, kodsuz (No Code – NC) ve az kodlu (Low Code – LC) platformlar, özellikle küçük ve orta ölçekli kobiler (KOBİ) için büyük bir avantaj olarak ortaya çıkmıştır. Kullanıcıların minimal teknik

bilgiyle uygulama geliştirmelerine olanak tanıyarak, yazılım maliyetlerini önemli ölçüde düşürmektedir. Gartner'ın 2022 ve 2023 raporlarına göre, dünya genelinde kodsuz ve az kodlu platform kullanım oranı 2022 yılında %19,6 artmış ve bu platformların kullanımının 2023 yılına kadar %20'yi aştığı görülmektedir. Ayrıca, Gartner'ın 2024 yılı raporuna göre, düşük kodlu uygulama geliştirme teknolojileri pazarının 26,9 milyar dolara ulaşacağı öngörülmektedir [2].

Yapay Zeka (YZ) ve Doğal Dil İşleme (DDİ) gibi ileri teknolojilerin bu platformlarda kullanılması, yazılım geliştirme süreçlerini daha da hızlandırabilir ve maliyetleri önemli ölçüde düşürebilmektedir. YZ tabanlı kodsuz ve az kodlu platformlar, kullanıcıların doğal dilde verdiği komutları anlayarak, bu komutlara uygun şekilde kod üretebilme yeteneğine sahiptir. Bu tez, YZ destekli dil modellerinin yazılım geliştirme süreçlerine olan katkılarını ele almakta ve bu kapsamda geliştirilen kodsuz platformun geliştirme süreçlerini ve etkilerini ele almaktadır. Geliştirilen kodsuz platforma, "Sıfır İşlem Yardımı" anlamına gelen "Zero Operation Assistance" (ZOA) adı verilmiştir.

YZ tabanlı kodsuz platformlar, yazılım geliştirme maliyetlerini önemli ölçüde düşürmekte ve geliştirme sürecini hızlandırmaktadır. Daha önce yapılan literatür araştırmasında [3] bu platformların geleneksel yöntemlere göre kullanıcı dostu yapısı ve daha hızlı çözüm üretmesiyle öne çıktığı görülmüştür. Bunun yanı sıra, YZ destekli sistemlerin doğal dil işleme kabiliyetleri, kullanıcıların karmaşık teknik bilgileri olmadan etkili web sayfaları oluşturmasına imkân tanımaktadır. ZOA platformu bu bağlamda tasarlanmış ve geliştirilmiştir. ZOA, kullanıcıların doğal dil komutlarıyla modern ve işlevsel web arayüzleri oluşturmasına olanak tanımaktadır. Çalışmada kullanılan YZ dil modeli, geniş ve çeşitlendirilmiş bir HTML – CSS veri seti ile eğitilmiştir. Tailwind CSS tabanlı bu veri seti, platformun çıktılarında estetik ve kullanıcı dostu bir tasarım sağlamak için tercih edilmiştir. Ayrıca, Python ve React teknolojilerinin bir arada kullanıldığı bu platform hem arka uç hem de ön yüz geliştirme süreçlerini entegre eden bir yapı sunmaktadır. Çalışmada ZOA'nın performansını ve kullanıcı deneyimini değerlendirmek amacıyla çeşitli test senaryoları uygulanmıştır. Bu testler, kullanıcıların taleplerini nasıl karşılayabildiğini ve çıktılarının doğruluğunu ölçmeye yöneliktir. Elde edilen sonuçlar, bu tür platformların gelecekteki yazılım geliştirme süreçlerinde sahip olacağı potansiyeli açıkça ortaya koymaktadır. Ayrıca, platformun ilerleyen dönemlerde daha geniş veri setleriyle eğitilerek performansının artırılabileceği ve farklı endüstri ihtiyaçlarına cevap verebilecek şekilde özelleştirilebileceği öngörülmektedir.

1.1. TEZİN AMACI

Bu çalışmanın temel amacı, YZ destekli dil modellerinin yazılım geliştirme süreçlerine olan katkılarını kapsamlı bir şekilde incelemektir. Özellikle kodsuz platformların teknik bilgi gereksinimini en aza indirerek geniş bir kullanıcı kitlesine ulaşma potansiyeli, bu çalışmanın çıkış noktalarından birini oluşturmaktadır. Bu bağlamda, ZOA platformunun geliştirilmesi hem kullanıcı deneyiminin iyileştirilmesi hem de yazılım geliştirme maliyetlerinin azaltılması açısından yenilikçi bir çözüm sunmayı hedeflemektedir.

Tez, mevcut kodsuz platformların sınırlarını genişletmek için YZ tabanlı bir dil modeli kullanmanın avantajlarını araştırmaktadır. ZOA platformu, kullanıcıların doğal dil girdileriyle web geliştirme bileşenleri oluşturmasını sağlarken, aynı zamanda kullanıcı deneyimini demokratikleştiren bir altyapı sunmaktadır. Bu hedef doğrultusunda, platformun hem teknik hem de kullanıcı dostu yönleri detaylı bir şekilde ele alınmıştır. Bu amaç doğrultusunda, dil modeli entegrasyonu, platformun fonksiyonelliği, kullanıcı deneyimi ve performans analizi olmak üzere dört hedef belirlenmiştir. Dil modeli entegrasyonu ile DDİ tekniklerinin web geliştirme süreçlerinde nasıl etkin bir şekilde kullanılabileceğini incelemek ve geniş kapsamlı bir dil modeli eğitimini gerçekleştirme hedeflendi. Platformun fonksiyonelliği için de ZOA platformunun kullanıcı gereksinimlerini karşılayacak şekilde tasarlanması ve doğal dil komutlarının doğru şekilde analiz edilerek dinamik web çıktıları üretmesi hedeflendi. Kullanıcı deneyiminde ise teknik bilgiye sahip olmayan kullanıcıların bile platformu kolayca kullanmasını sağlayacak bir arayüz geliştirmesi amaçlandı. Son olarak, performans analizi için platformun performansını ve çıktı kalitesini ölçmek için kapsamlı test senaryoları geliştirmek ve analiz sonuçlarına dayalı iyileştirmelerin yapılması hedeflendi.

Bu çalışmanın geniş perspektifi, YZ destekli çözümlerin yazılım geliştirme süreçlerinde nasıl dönüştürücü bir etkiye sahip olabileceğini göstermektedir. Dil modellerinin kullanımı, kullanıcıların fikirlerini kolayca prototiplere dönüştürebileceği bir araç sağlamaktadır. Özellikle KOBİ'lerin yazılım geliştirme kapasitesini artırmada bu tür bir platformun stratejik öneme sahip olduğu düşünülmektedir. Ayrıca, ZOA platformu hem bireysel hem de kurumsal kullanıcılara yönelik çözümleriyle farklılaştırılmış bir deneyim sunmayı hedeflemektedir. Kullanıcıların karmaşık teknik detaylarla uğraşmadan üretkenliklerini artırmalarını sağlayan bir altyapı oluşturan ZOA, bu yönüyle kullanıcı dostu ve güçlü bir platform örneği olarak ele alınmıştır. Bu bağlamda, ZOA'nın tasarımı

ve uygulama süreçleri, gelecekteki yazılım geliştirme araçları için referans teşkil edebilecek bir model olarak değerlendirilmiştir.

Literatürdeki çalışmalar, kodsuz platformların yazılım geliştirme süreçlerinde sağladığı avantajların yanı sıra kullanıcı deneyimi ve ölçeklenebilirlik konularında sınırlamalar taşıdığını göstermektedir. Bu tez, bu sınırlamaların ötesine geçmek ve daha esnek, güçlü bir çözüm sunmak için ZOA platformunu bir örnek vaka olarak ele almıştır. Ayrıca, bu çalışmada elde edilen bulgular, sadece yazılım geliştirme alanında değil, YZ tabanlı kullanıcı ara yüzleri ve otomasyon süreçlerinde de yeni açılımlar sunmayı amaçlamaktadır. Platformun hem bireysel hem de kurumsal kullanıcılar için geliştirme süreçlerini kolaylaştırması ve hızlandırması hedeflenmektedir. Böylelikle, yazılım geliştirme ekosistemine katkı sağlayacak yeni nesil teknolojilerin temelini oluşturması beklenmektedir. Bu tez, aynı zamanda, dil modellerinin teknik bilgi gerektirmeyen kullanıcı grupları tarafından nasıl kullanılabileceğini göstermeyi amaçlamakta ve bu bağlamda teknolojiye erişimi daha demokratik bir hale getirmeyi hedeflemektedir. ZOA platformunun geliştirilmesi, yazılım geliştirme süreçlerini herkes için erişilebilir kılmayı sağlayacak yenilikçi bir çözüm sunmaktadır. Geliştirilen platform, gelecekte daha karmaşık yazılım süreçlerini kolaylaştıracak sistemlerin temel taşı olarak değerlendirilebilir.

1.2. LİTERATÜRE KATKI

Bu çalışma hem YZ hem de kodsuz yazılım geliştirme platformları üzerine yapılan araştırmalara anlamlı katkılar sunmayı hedeflemektedir. Literatürde, YZ destekli dil modellerinin yazılım süreçlerinde nasıl kullanılabileceği konusunda çeşitli çalışmalar yapılmış olsa da bu teknolojilerin kodsuz platformlara entegrasyonu hâlâ sınırlı bir alandır. Bu tez, YZ tabanlı dil modellerinin kodsuz platformlarla birleştirilmesiyle mevcut bilgiye yenilikçi bir perspektif kazandırmayı amaçlamaktadır. Bu tez kapsamında geliştirilen ZOA platformu, YZ'nin kodlama süreçlerine entegrasyonu, kullanıcı deneyiminin demokratikleşmesi, performans değerlendirme, optimizasyon, veri seti çeşitliliği, model eğitimi ve yazılım geliştirme ekosistemine yapılan yeni yaklaşımlar ile literatüre katkı sağlamaktadır. Literatürde mevcut çalışmalar genellikle yapay zekanın yazılım geliştirme süreçlerine etkilerini ele alırken, bu çalışmaların çoğu dil modellerinin kodsuz platformlara entegrasyonunu ihmal etmektedir. ZOA platformu, DDİ tekniklerini kullanarak, kullanıcıların doğal dil girdilerinden doğrudan işlevsel web bileşenleri

üretmesini mümkün kılarak bu boşluğu doldurmaktadır. Öte yandan çoğu yazılım geliştirme aracı teknik bilgiye dayalıdır ve bu durum, yazılım geliştirme süreçlerini birçok kullanıcı için erişilemez kılmaktadır. Bu çalışma, ZOA platformu aracılığıyla, teknik bilgisi olmayan kullanıcıların bile karmaşık web uygulamaları oluşturabileceği bir yapı sunmaktadır. Bu, literatürde kullanıcı deneyimini demokratikleştirme konusundaki çalışmalara önemli bir katkı sağlamaktadır. Tez kapsamında gerçekleştirilen test senaryoları ve performans analizleri, YZ destekli kodsuz platformların çıktılarının doğruluğunu ve etkinliğini değerlendirmek için detaylı bir çerçeve sunmaktadır. Bu tür bir değerlendirme yöntemi, gelecekteki çalışmalar için temel oluşturabilir. ZOA platformunun geliştirilmesi sırasında kullanılan geniş ve çeşitli veri setleri, dil modellerinin farklı senaryolarda nasıl performans gösterdiğini analiz etmeye olanak tanımıştır. Literatürde sıklıkla ihmal edilen veri seti çeşitliliği ve model eğitimi süreçleri, bu tezde detaylı bir şekilde ele alınmıştır. ZOA, sadece bir platform değil, aynı zamanda yazılım geliştirme süreçlerini daha erişilebilir ve esnek hale getirebilecek bir ekosistemin temelini oluşturmaktadır. Bu bağlamda, tezde sunulan model ve yaklaşım, gelecekteki araştırmalara rehberlik edecek yenilikçi bir çerçeve sağlamaktadır. Ayrıca, kodsuz yazılım geliştirme süreçlerine yapay zekanın nasıl entegre edilebileceğini detaylı bir şekilde ele alarak hem akademik hem de pratik anlamda yeni bir katkı sunmaktadır. Geliştirilen platform, sadece bir teorik model olarak kalmamakta, aynı zamanda gerçek dünya uygulamaları için kullanılabilir bir araç olarak değerlendirilmektedir. Bu yönüyle, yazılım geliştirme ve YZ entegrasyonu alanında ilerleyen çalışmalar için önemli bir referans kaynağı olacağı öngörülmektedir.

1.3. LİTERATÜR ÖZETİ

YZ destekli kodsuz yazılım geliştirme platformları, son yıllarda yazılım mühendisliği ve kullanıcı deneyimi alanlarında giderek daha fazla ilgi görmektedir. Literatürdeki çalışmalar [3], bu platformların yazılım geliştirme sürecinde sağladığı avantajlara dikkat çekerken, aynı zamanda bazı sınırlamalarını da vurgulamaktadır. Kodsuz platformlar, teknik bilgiye sahip olmayan kullanıcıların yazılım geliştirme süreçlerine dâhil olmasına olanak tanırken, YZ teknolojileri bu süreçleri daha verimli ve kullanıcı dostu hale getirmektedir. YZ, insan zekasını temel alan ancak biyolojik sınırlamalarla kısıtlanmayan, akıllı makineler ve bilgisayar programları geliştirme amacı güden bir bilim ve mühendislik dalı olarak tanımlanmaktadır [4], [5], [6], [7].

DDİ tabanlı dil modelleri, kullanıcıların teknik bilgiye ihtiyaç duymadan yazılım geliştirme süreçlerini kolaylaştırmaktadır. Literatürde yer alan araştırmalar [8], [9], dil modellerinin, kullanıcıların doğal dil girdilerini anlayarak bu girdileri işlevsel yazılım bileşenlerine dönüştürmede etkili olduğunu göstermektedir. Örneğin, kullanıcıların "bir ürün sayfası oluştur" gibi basit bir komutla karmaşık bir web sayfası tasarımını başlatabilmesi, DDİ tabanlı sistemlerin gücünü ortaya koymaktadır. DDİ yöntemini kullanan Büyük Dil Modelleri (BDM, Large Language Models – LLMs) yapay zeka geliştirilmesinde kullanılmaktadır.

Literatürdeki bir diğer önemli odak noktası, YZ destekli sistemlerin ölçeklenebilirlik ve kullanıcı deneyimi üzerindeki etkileridir [10]. YZ destekli kodsuz platformların kullanımı, kullanıcı deneyimini iyileştirme potansiyeline sahiptir. Bununla birlikte, bu platformların daha geniş ölçeklerde uygulanabilirliği konusunda bazı sınırlamalar mevcuttur. Özellikle büyük veri setlerinin işlenmesi ve daha karmaşık projelerin yönetimi gibi konular, literatürde tartışılan başlıca zorluklar arasındadır [11]. Veri çeşitliliği ve model eğitimi, YZ tabanlı platformların başarısını etkileyen kritik faktörlerdir. Geniş ve çeşitli veri setleri, dil modellerinin performansını artırırken, eksik veya yetersiz veri setleri bu modellerin sınırlı kalmasına neden olabilir. Literatürde [12], YZ modellerinin başarısını artırmak için veri çeşitliliğinin önemi sıklıkla vurgulanmaktadır. Bu kapsamda, geliştirilen ZOA platformunda kullanılan veri setlerinin çeşitliliği, literatürde önerilen yöntemlerle uyum göstermektedir.

Kodlama süreçlerinin demokratikleştirilmesi, sıkça tartışılan bir diğer konudur [13]. Geleneksel yazılım geliştirme süreçleri genellikle yalnızca teknik bilgiye sahip kişiler tarafından erişilebilirken, YZ destekli kodsuz platformlar, daha geniş bir kullanıcı kitlesine erişim sağlamaktadır. Bu, yazılım geliştirme süreçlerinin daha erişilebilir hale gelmesi açısından önemli bir adımdır. Kullanıcıların teknik engelleri aşarak yaratıcı fikirlerini hayata geçirebilmeleri, bu platformların en güçlü yönlerinden biri olarak öne çıkmaktadır. YZ destekli kodsuz platformların avantajlarının yanı sıra sınırlamaları da ele alınmaktadır. Özellikle güvenlik ve veri gizliliği konuları, bu platformların yaygınlaşmasıyla birlikte daha fazla önem kazanmaktadır. Kullanıcıların, verilerinin güvenli bir şekilde işlenmesini sağlamak için YZ modellerinin daha şeffaf hale getirilmesi gerektiği literatürde vurgulanmaktadır [14].

Tüm bu sonuçlar gösteriyor ki, YZ ve kodsuz platform teknolojilerinin yazılım geliştirme süreçlerinde dönüştürücü bir etkisi bulunmaktadır. Bununla birlikte, literatürde eksik

kalan yönlere dikkat çekmek gereklidir. Özellikle YZ destekli, basit ve kullanıcı dostu bir kodsuz platform eksikliği ve hem küçük hem büyük ölçekli işletmeler için çalışabilir halde olması gibi konular daha fazla araştırmayı gerektirmektedir. Bu tez, bu tür eksikliklerin ele alınması ve ZOA platformu gibi yenilikçi çözümlerle giderilmesi açısından önemli bir referans sunmaktadır. Bununla birlikte, platformların etkinliği, veri setlerinin kalitesi, kullanıcı deneyimi ve güvenlik önlemleri gibi faktörlere bağlıdır. ZOA platformu, bu alanlarda yapılan yenilikçi çalışmaları temel alarak hem akademik hem de endüstriyel bağlamda değerli bir katkı sağlamayı amaçlamaktadır. İlerleyen bölümlerde, ZOA platformunun tasarım ve geliştirme süreçleri detaylandırılacak ve bu süreçlerin literatürde ele alınan yaklaşımlarla ne derece örtüştüğü analiz edilecektir. Bu bağlamda, platformun sunduğu yeniliklerin ve çözüm önerilerinin, mevcut çalışmaların ötesine nasıl geçtiği irdelenecek ve ileriye dönük araştırmalar için yön gösterici bir kaynak oluşturacaktır.

Kodlama ve YZ teknolojilerinin entegrasyonu ile yazılım geliştirme alanında daha fazla erişilebilirlik sağlamak için yeni fırsatların sunulması bu çalışmanın da fikir aşamasında etkili olmuştur. Örneğin, ZOA platformunun geliştirilmesi sırasında DDİ teknikleri, kullanıcıların karmaşık teknik bilgilere ihtiyaç duymadan yazılım çıktıları oluşturmasını sağlamak amacıyla kullanılmıştır. Bu platform, kullanıcıların "Bir iletişim formu oluştur" veya "Ürün listeleme ekranı oluştur" gibi doğal dil komutlarıyla dinamik ve işlevsel web sayfaları geliştirmesine olanak tanımaktadır. Ayrıca, ZOA, büyük veri setlerinden anlamlı bilgiler çıkararak kullanıcı gereksinimlerine hızlı bir şekilde yanıt verebilecek şekilde optimize edilmiştir. Bu uygulama hem bireysel hem de kurumsal kullanıcılar için yazılım geliştirme süreçlerini daha erişilebilir hale getiren önemli bir örneği temsil etmektedir. Özellikle DDİ tekniklerinin kodsuz platformlara entegrasyonu, yazılım süreçlerini önemli ölçüde kolaylaştırmaktadır. Literatürde, bu entegrasyonun pratik uygulamalarına ilişkin kapsamlı çalışmalar bulunmamakla birlikte, mevcut araştırmaların bu teknolojilerin potansiyelini ortaya koyduğu görülmektedir [13], [15].

Kodsuz platformların kullanım kolaylığı, geliştiricilerin iş yükünü hafifletirken, kullanıcıların yazılım geliştirme süreçlerinde daha aktif bir şekilde yer almasını sağlamaktadır. Bu durum, yazılım geliştirme süreçlerinin demokratikleşmesini hızlandırmış ve yazılım alanında yeni kullanıcı gruplarının yer almasına imkân tanımıştır. Literatürde bu süreçlerin kullanıcı deneyimine olan katkıları detaylı bir şekilde incelenmiştir [16]. Öte yandan, kodsuz platformların geliştirilmesinde YZ tekniklerinin

kullanımı, yazılım geliştirme süreçlerini otomatikleştirme potansiyeli taşımaktadır. YZ algoritmaları, kullanıcıların taleplerine hızlı ve doğru bir şekilde yanıt vererek yazılım çıktılarının kalite ve verimliliğini artırmaktadır. Bununla birlikte, bu teknolojilerin güvenlik ve veri gizliliği konularında geliştirilmesi gereken alanları bulunmaktadır [14].

Bir diğer önemli konu ise, YZ destekli dil modellerinin veri işleme kapasitesidir. ZOA platformunda, bu kapasite çeşitli optimizasyon teknikleri ve geniş veri setleriyle geliştirilmiştir. Örneğin, DDİ tabanlı algoritmaların büyük hacimli veri setlerini hızlı ve doğru bir şekilde analiz edebilmesi, kullanıcıların girdilerini daha anlamlı sonuçlara dönüştürmede kritik bir rol oynamaktadır. Ayrıca, ZOA'nın veri çeşitliliği dikkate alınarak eğitilmesi, platformun farklı senaryolara adaptasyon yeteneğini artırmıştır. Bunun sonucunda hem kullanıcı deneyimi iyileştirilmiş hem de platformun çıktılarının doğruluğu ve güvenilirliği önemli ölçüde artırılmıştır. Literatürde yer alan araştırmalar [17], büyük veri setlerinin YZ modellerinin performansını olumlu yönde etkilediğini ve daha doğru sonuçlar elde edilmesini sağladığını göstermektedir. Bu bağlamda, ZOA platformunda kullanılan geniş veri setleri, literatürdeki önerilerle uyumlu bir şekilde modellenmiştir. Uygulanan yöntemler, YZ destekli kodsuz platformların performansını artırmak için önerilen yöntemlerle paralellik göstermektedir. Örneğin, veri çeşitliliği ve kullanıcı girdilerinin işlenmesi, platformun kullanıcı ihtiyaçlarına yönelik uyarlanabilirliğini artıran unsurlar olarak ele alınmıştır. Bu durum, platformun genel kullanıcı deneyimi üzerindeki olumlu etkisini artırmıştır. Bunun yanı sıra, ZOA platformunun sunduğu yenilikçi özellikler, yazılım geliştirme süreçlerine ilişkin mevcut yaklaşımlardan ayrılmaktadır. En büyük sınırlamalardan biri olan kullanıcıların teknik bilgi eksikliği, ZOA'nın DDİ tabanlı altyapısı sayesinde aşılmıştır. Bu sayede, kullanıcılar karmaşık komutlarla uğraşmadan yazılım geliştirme sürecine doğrudan katkıda bulunabilmektedir.

YZ destekli kodsuz platformların ölçeklenebilirliği, geniş çaplı projelerde sağladığı esneklik ile de dikkat çekmektedir. Literatürde YZ desteğinin farklı sektörlerdeki uygulama potansiyeli ele alınmakta ve özellikle eğitim, sağlık ve yazılım gibi alanlarda etkin bir şekilde kullanılabileceği vurgulanmaktadır [18], [19], [20], [21], [22], [23], [24], [25]. ZOA platformu, literatürde ele alınan tüm bu unsurları bir araya getirerek hem teorik hem de pratik anlamda önemli bir yenilik sunmaktadır. Gelecek çalışmalar için bir model teşkil eden bu platform, yazılım geliştirme süreçlerini daha erişilebilir ve etkili hale getirme potansiyeline sahiptir.

2. GENEL BİLGİLER

Bu bölümde, tez kapsamında ele alınan temel kavramlar, kullanılan teknolojiler, kütüphaneler ve diğer ilgili terimler detaylı bir şekilde açıklanmaktadır. Her bir başlık, konunun anlaşılmasını kolaylaştırmak ve okuyucunun çalışma kapsamında geliştirilen ZOA adlı, YZ destekli kodsuz platformun arkasındaki altyapıyı daha iyi anlamasını sağlamak amacıyla ele alınmıştır.

Yapay Zeka (YZ), makinelerin insan benzeri bir şekilde düşünme, öğrenme ve problem çözme yeteneklerini geliştirme teknolojisidir. Günümüzde yapay zekanın kullanımı, birçok farklı uygulama ile örneklendirilebilir. Örneğin, OpenAI tarafından geliştirilen ChatGPT [26], [27], kullanıcıların metin tabanlı sorularına yanıt verebilme kapasitesine sahip gelişmiş bir DDİ modelidir. Bunun yanı sıra, Microsoft CoPilot ve GitHub CoPilot [28], [29] gibi araçlar, yazılım geliştirme süreçlerinde kod önerileri sağlayarak geliştiricilere zaman kazandırmaktadır. Bu uygulamalar, yapay zekanın yazılım mühendisliği ve kullanıcı etkileşimi üzerindeki dönüştürücü etkisini göstermektedir. Ayrıca, Claude [30], [31] gibi modeller, metin analizi ve işleme konusunda karmaşık görevleri başarıyla yerine getirerek, YZ kullanımını geniş bir yelpazede pratik hale getirmektedir. Bu kavramın kökenleri, 20. yüzyılın ortalarına kadar uzanmakta olup, ilk olarak 1956 yılında Dartmouth Konferansı'nda "Yapay Zeka" terimi ortaya atılmıştır [7]. O dönemde, bilgisayarların sadece matematiksel işlemleri değil, aynı zamanda karmaşık problemleri çözme yeteneğine sahip olabileceği öne sürülmüştür.

Yapay zekanın ilk yıllarında geliştirilen sistemler, satranç oynama gibi belirli kurallara dayalı görevlerde sınırlı başarılar göstermiştir. Ancak, bilgi işlem gücündeki artış ve veri miktarındaki büyüme ile, YZ daha karmaşık uygulamalar için kullanılabilir hale gelmiştir. 1980'lerde uzman sistemlerin yükselişi, tıp ve mühendislik gibi alanlarda yeni kapılar açmıştır. Günümüzde YZ, otomatik kod üretimi, hataların tespiti ve çözümü gibi yazılım geliştirme süreçlerinde geniş bir uygulama yelpazesine sahiptir. Bu tez kapsamında geliştirilen YZ destekli kodsuz ZOA platformu, bu teknolojiyi özellikle dil modelleme ve DDİ süreçlerinde etkin bir şekilde kullanmaktadır. Örneğin, ZOA, kullanıcıların doğal dil girdilerinden hareketle karmaşık web bileşenleri oluşturmasına olanak tanır. Bu süreçte, derin öğrenme (Deep Learning – DL) algoritmaları ve geniş veri setlerinden yararlanarak kullanıcı deneyimini geliştirmektedir.

Yapay zekanın günümüzdeki etkisi, sağlık, finans, eğitim ve ulaşım gibi sektörlere de yayılmış durumdadır. Otonom araçlardan tıbbi teşhise, YZ çözümleri insanların hayatını kolaylaştıran araçlar haline gelmiştir [32], [33]. Özellikle DDİ gibi alanlarda kaydedilen ilerlemeler, yapay zekanın günlük yaşamdaki rolünü artırmıştır. DDİ, ZOA platformunda kullanıcıların ihtiyaçlarını hızlı ve doğru bir şekilde karşılamak için kullanılmaktadır. Diğer yandan, YZ etik ve güvenlik gibi alanlarda da tartışmalara yol açmaktadır [34], [35]. Verilerin güvenliği ve yapay zekanın sorumlu kullanımı, bu teknolojinin sürdürülebilir bir şekilde büyümesi için kritik öneme sahiptir. ZOA platformu, bu tartışmalara cevap olarak modern şifreleme teknikleri ve veri koruma standartlarını entegre ederek, kullanıcı verilerinin güvenli bir şekilde işlenmesini sağlamaktadır. Geldiğimiz noktada YZ, geçmişten günümüze sürekli evrilerek hem akademik dünyada hem de endüstriyel uygulamalarda devrim yaratmıştır. ZOA gibi platformlar, bu teknolojinin yazılım geliştirme süreçlerinde nasıl dönüştürücü bir etki yaratabileceğini göstermektedir.

Doğal Dil İşleme (DDİ) [36], bilgisayarların insan dillerini anlamasını, yorumlamasını ve bu bilgilerden anlamlı çıktılar üretmesini sağlayan YZ alt dallarından biridir. Bu teknoloji, insan dilindeki karmaşıklıkları çözümleyerek kelimelerin anlamlarını, dil bilgisi kurallarını ve bağlam içindeki ilişkilerini değerlendirme yeteneğine sahiptir [8]. DDİ, günümüzde makine çevirisinden sesli asistanlara, duygu analizi araçlarından metin özetleme sistemlerine kadar geniş bir yelpazede uygulanmaktadır.

Tez kapsamında geliştirilen YZ destekli kodsuz platformda, DDİ teknolojisi kullanıcı girdilerini analiz etmek ve bu girdilere uygun işlevsel web bileşenleri üretmek için önemli bir rol üstlenmektedir. Örneğin, bir kullanıcının "Bir iletişim formu oluştur" veya "Ürün listesi ekranı oluştur" gibi doğal dilde verdiği talimatlar, DDİ algoritmaları tarafından işlenerek yazılım çıktısına dönüştürülmektedir. Bu süreçte, DDİ yalnızca kullanıcının söylediği kelimeleri değil, aynı zamanda bağlamı, amacı ve olası tasarım ihtiyaçlarını da değerlendirerek uygun HTML ve CSS kodlarını üretir. Bu sayede, teknik bilgiye sahip olmayan kullanıcıların bile doğal dil girdileriyle profesyonel web sayfaları oluşturabilmesi sağlanmaktadır. DDİ'nin bu gibi uygulamaları, yalnızca kullanıcı deneyimini iyileştirmekle kalmaz, aynı zamanda yazılım geliştirme süreçlerini hızlandırarak maliyetleri düşürür ve erişilebilirliği artırır. Bu özellikler, DDİ teknolojisinin kodsuz platformlarda neden merkezi bir konumda olduğunu açıkça ortaya koymaktadır.

Büyük Dil Modelleri (BDM, Large Language Models – LLMs) [37], kelime dizilerinin olasılık dağılımlarını tahmin ederek doğal dil işlemede kullanılır. Herhangi bir uzunluktaki (m) kelimeler dizisini dil modeline verildiğinde, tüm dizinin olasılığına olasılık atmaktadır; matematiksel karşılığı denklem (2.1) ile gösterilmiştir [38].

$$P(w_1, \dots, w_m) \tag{2.1}$$

BDM, gelişen derin öğrenme teknikleri, büyük veri setleri, güçlü hesaplama kaynakları, dil anlama ve üretme yeteneklerini artırmak için kullanılmaktadır [39]. Bu modellerin temel özelliği, büyük bir metin veri setini eğitim verisi olarak kullanmalarıdır. Bu veri setleri, internetten veya kitaplardan toplanan milyonlarca metin belgesini içerebilir ve bu da modelin dil hakkında derin bir anlama becerisi geliştirmesine yardımcı olur.

2010 yılından itibaren GPT (Generative Pre-trained Transformer), BERT (Bidirectional Encoder Representations from Transformers) gibi büyük dil modelleri DDİ alanında çığır açtı [40]. Dili anlama ve üretme yetenekleri büyük ölçüde geliştirdi. 2020'lere geldiğimizde ise GPT-3 [41] ve GPT-4 [42], [43] gibi dil modelleri büyük ölçekli dil üretme yeteneğiyle popüler hale geldi ve birçok alanda kullanılmaya başlandı. En büyük kullanım alanlarından biri ise yazılım oldu. Yazılım alanında kod üretme, hata çözümleme gibi konularda büyük dil modellerinden destek alınmaktadır.

GPT-4 [42], OpenAI tarafından geliştirilen büyük bir dil modelidir ve 2023 yılında piyasaya sürülmüştür. Bu model, önceki versiyonları olan GPT-3'ten daha gelişmiş özellikler sunmaktadır. GPT-4, daha geniş ve daha güncel bir veri kümesi üzerinde eğitildiğinden dolayı daha karmaşık dil görevlerini yerine getirme yeteneğine sahiptir. Özellikle, DDİ alanında önemli bir ilerleme kaydetmiş; metin oluşturma, makine çevrisi ve metin özetleme gibi görevlerde daha yüksek bir doğruluk oranı sağlamıştır [44]. Modelin eğitimi sırasında insan benzeri yanıtlar üretme yeteneği geliştirilmiş ve bu sayede kullanıcılarla etkileşimli bir deneyim sunulmuştur. GPT-4'ün çıkışı, YZ ve dil işleme alanında önemli bir dönüm noktası olarak kabul edilmektedir. Kullanıcı sayısı, modelin piyasaya sürülmesinden sadece birkaç gün sonra bir milyona ulaşmış ve iki ay içinde aylık 100 milyon aktif kullanıcıya ulaşarak tarihin en hızlı büyüyen tüketici uygulaması olmuştur [44]. Bu hızlı benimseme, kullanıcıların modelin sunduğu etkileşimli ve bilgilendirici yanıtları takdir etmesinden kaynaklanmaktadır [45]. GPT-4, aynı zamanda daha önceki sürümlerde gözlemlenen bazı sınırlamaları aşmayı hedeflemiş

ve daha iyi bir bağlam anlayışı geliştirmiştir [44].

Claude, Anthropic tarafından geliştirilen bir başka büyük dil modelidir. Bu model, özellikle güvenlik ve etik konularına odaklanarak tasarlanmıştır. Claude, kullanıcıların güvenli bir şekilde etkileşimde bulunabilmesi için tasarlanmış bir dizi önlem içermektedir. Model, insan benzeri yanıtlar üretme yeteneği ile dikkat çekmektedir, ancak aynı zamanda kullanıcıların güvenliğini sağlamak için çeşitli filtreleme mekanizmaları da içermektedir. Claude, DDİ alanında önemli bir alternatif olarak öne çıkmakta ve kullanıcıların ihtiyaçlarına göre özelleştirilebilen bir yapı sunmaktadır. Claude'un geliştirilmesi, YZ alanında etik ve güvenlik konularının öneminin arttığı bir dönemde gerçekleşmiştir. Model, kullanıcıların güvenliğini sağlamak amacıyla tasarlanmış olup bu sayede kullanıcıların modelle olan etkileşimlerinde daha az risk taşımaktadır. Claude, aynı zamanda kullanıcıların geri bildirimlerine dayalı olarak sürekli olarak güncellenmekte ve iyileştirilmektedir; bu da onu dinamik bir dil modeli haline getirmektedir [7], [30], [31].

Qwen, Alibaba tarafından geliştirilen bir büyük dil modelidir. Bu model, özellikle Asya pazarında büyük bir etki yaratmayı hedeflemektedir. Qwen, DDİ yetenekleri ile dikkat çekmekte ve çok dilli destek sunarak farklı dillerde kullanıcılarla etkileşimde bulunabilmektedir. Model, özellikle ticaret ve müşteri hizmetleri alanında kullanılmak üzere tasarlanmış olup, kullanıcıların ihtiyaçlarına yönelik özelleştirilmiş çözümler sunmaktadır. Qwen'in geliştirilmesi, Alibaba'nın YZ alanındaki büyüme stratejisinin bir parçasıdır. Model, kullanıcıların sorularına hızlı ve doğru yanıtlar verebilme yeteneği ile öne çıkmakta ve bu sayede müşteri memnuniyetini artırmayı hedeflemektedir. Qwen, aynı zamanda büyük veri analizi ve makine öğrenimi tekniklerini kullanarak sürekli olarak kendini geliştirmekte ve kullanıcı deneyimini iyileştirmeye yönelik adımlar atmaktadır [46].

Llama (Large Language Model Meta AI), Meta (eski adıyla Facebook) tarafından geliştirilen bir dil modelidir. Bu model, özellikle araştırma ve akademik alanlarda kullanılmak üzere tasarlanmıştır. Llama, büyük veri kümesi üzerinde eğitilmiş olup, dil anlama ve metin oluşturma konularında yüksek bir performans sergilemektedir. Model, akademik araştırmalar için önemli bir araç olarak öne çıkmakta ve araştırmacılara çeşitli dil görevlerini yerine getirme konusunda yardımcı olmaktadır. Llama'nın geliştirilmesi, Meta'nın YZ alanındaki araştırma ve geliştirme çabalarının bir parçasıdır. Model, kullanıcıların dil işleme ihtiyaçlarına yönelik özelleştirilmiş çözümler sunmakta ve bu

sayede akademik çalışmalara katkıda bulunmaktadır. Llama, aynı zamanda araştırmacıların dil modellerini daha iyi anlamalarına yardımcı olmak amacıyla çeşitli araçlar ve kaynaklar sunmaktadır [47], [48].

Dil modellerinin eğitimi, doğal dil işlemenin temel taşlarından biri olup, büyük miktarda dil verisini analiz ederek insan dilini anlama ve üretme yeteneklerini geliştirir. Bu süreç, dil modellerinin hem genel hem de görev özelinde etkili olmasını sağlamak için çeşitli yöntemleri içerir. Bu yöntemlerden biri olan İnce Ayar (Fine-Tuning), önceden eğitilmiş bir dil modelinin belirli bir görev için optimize edilmesi sürecidir ve genellikle sınırlı veriyle çalışırken üstün performans elde etmek için kullanılır. Bir diğer önemli yaklaşım ise Getirme Destekli Üretim (Retrieval-Augmented Generation – RAG) yöntemidir. RAG, dil modellerine dış veri kaynaklarından bilgi çekme yeteneği ekleyerek hem bilgiye dayalı metin üretimini hem de soru-cevap gibi karmaşık görevlerdeki doğruluğu artırır. Bu iki yöntem, dil modeli eğitiminin evriminde farklı ancak tamamlayıcı rollere sahiptir ve doğal dil işlemedeki başarının kritik unsurlarını oluşturur.

İnce ayar ile model eğitimi, özellikle önceden eğitilmiş modellerin özel görevlere uyarlanmasında, makine öğreniminde önemli bir strateji olarak öne çıkmıştır. Bu yöntem, büyük veri setleri üzerinde eğitilmiş bir modelin genelleştirilmiş bilgisini kullanarak, sınırlı veriyle belirli alanlarda üstün performans göstermesini sağlar. Howard ve Ruder'in (2018) açıkladığı gibi [49], ince ayar, modellerin yeni görevlere uyarlanması sırasında aşırı öğrenme (overfitting) ve katastrofik unutma gibi zorlukların üstesinden gelinmesine olanak tanır. İnce ayar, transfer öğrenimi prensibine dayanır; burada bir kaynak alandaki bilgi, hedef alana aktarılır. DDİ alanında ince ayar, önceden eğitilmiş bir dil modelinin çeşitli metin sınıflandırma görevlerine uyarlanmasını içeren Evrensel Dil Modeli İnce Ayarı (Universal Language Model Fine-Tuning – ULMFiT) yaklaşımı ile örneklenir [49]. Bu süreç, genel alan ön eğitimi, görev özelinde dil modeli uyarlaması ve sınıflandırıcı ince ayar olmak üzere üç ana aşamadan oluşur. ULMFiT, yeni verilere uyum sağlarken temel özellikleri koruma sorunlarını çözmek için ayrık ince ayar ve eğik üçgen öğrenme oranları gibi yenilikçi teknikler sunmuştur.

İnce ayarın etkinliği, önceden eğitilmiş modelin kalitesine ve kapsamına bağlıdır. Örneğin, Vikipedi gibi kaynaklardan eğitilen genel alan dil modelleri, çeşitli dilbilimsel özellikleri yakalayarak aşağı akış görevleri için sağlam bir temel sağlar. Özel görevlere uygulandığında, ince ayar, modelin yeni veri setinin inceliklerine uyum sağlamasına olanak tanır. Örneğin, ince ayarın duygu analizi ve konu sınıflandırma görevlerinde, çok

daha az etiketlenmiş veriyle en son teknoloji seviyesinde sonuçlar elde ettiği gösterilmiştir [49], [50]. Friederich'in (2017) ele aldığı gibi [50], ince ayara dair felsefi bir perspektif, bunun fiziksel sistemlerin ince ayarına benzetildiğini vurgular; bu sistemlerde, parametreler optimal performans sağlamak için ayarlanır. Bu benzetme, hiperparametre ayarı ve gradyan optimizasyonunun model performansını artırmak için kullanıldığı makine öğrenimine de uzanır. Bu tür bir ayar, ince ayar yöntemlerinin başarısı için kritik olan genelleştirme ve uzmanlaşma arasındaki dengeyi yansıtır. Ayrıca, Goodfellow ve arkadaşlarının (2016) derin öğrenme üzerine yaptığı temel çalışma [51], ince ayarın teorik temellerini vurgular; bunlar arasında gradyan tabanlı optimizasyon tekniklerinin ve düzenleme stratejilerinin önemi yer alır. Bu ilkeler, uyarlama sürecinin görev özelinde doğruluğu artırırken, modelin genel dilsel yeteneklerini korumasını sağlar.

İnce ayar, makine öğreniminde genelleştirilmiş öğrenme ile alan özelinde mükemmellik arasındaki boşluğu dolduran dönüştürücü bir yaklaşımı temsil eder. Ayrık öğrenme oranları gibi teknikler kullanılarak ve önceden eğitilmiş bilgiden yararlanılarak ince ayar, yalnızca model performansını optimize etmekle kalmaz, aynı zamanda ileri düzey YZ teknolojilerinin, çeşitli ve kaynak açısından sınırlı görevlere uygulanmasını da demokratikleştirir. Teorik bilgiler ve pratik metodolojilerin sinerjisi, makine öğreniminde ince ayar ile elde edilebileceklerin ufkunu genişletmeye devam etmektedir.

RAG, DDİ alanında, büyük dil modelleri ile alan özelindeki gereksinimler arasında köprü kuran devrim niteliğinde bir gelişmedir. RAG, dış veri kaynaklarından bilgi çekmeyi ve metin üretimini birleştirerek özellikle alan bilgisi gerektiren durumlarda yanıtların doğruluğunu ve alaka düzeyini artırır [52], [53]. RAG, iki temel bileşenden oluşur; bir "retriever" modeli ve bir "generator" modeli. Retriever, bilgi tabanları veya kurumsal belgeler gibi dış veri kaynaklarını tarayarak bağlama uygun bilgileri çekerken, generator bu bağlamı kullanarak bilgilendirici ve uyumlu yanıtlar üretir. Bu çift mekanizma, büyük dil modellerinin "halüsinasyonlar" (mantıklı görünen ancak yanlış bilgiler üretme) gibi sınırlamalarını, çıktıları gerçek dünya verilerine dayandırarak aşar [53], [54]. Lewis ve arkadaşları (2021) tarafından tanıtılan RAG, üretken yapay zeka alanında önemli bir kilometre taşı olmuştur. RAG'ın kurumsal sistemlerdeki uygulamaları, veri parçalanması, ölçeklenebilirlik ve uyumluluk gibi zorlukların üstesinden gelmekteki faydasını kanıtlamaktadır. Örneğin, RAG destekli bir kurumsal chatbot, yapılandırılmış veri tabanları ve yapılandırılmamış belgelerden bilgi alıp sentezleyerek kullanıcı sorgularına doğru ve verimli çözümler sunabilir [54], [55].

RAG'ın dikkat çekici avantajlarından biri, alan özelindeki kullanım senaryolarına uyum sağlama yeteneğidir. Örneğin, sağlık alanında RAG, hasta kayıtlarını alıp doğru tıbbi tavsiyeler üreterek karar alma süreçlerini iyileştirmek için kullanılmaktadır. Benzer şekilde, finansal hizmetlerde RAG, politika belgelerini kullanıcı sorgularına göre yorumlayarak müşteri destek süreçlerini güçlendirmektedir [53], [55]. Takviyeli Öğrenme (Reinforcement Learning – RL) gibi optimizasyon teknikleri, RAG süreçlerini daha da geliştirmektedir. RL tabanlı modeller, dış bağlamın ne zaman alınması gerektiğine karar vererek, hesaplama maliyetlerini düşürürken doğruluğu koruyabilir veya artırabilir. Bu yaklaşım, kaynak kullanımını en aza indirerek çok adımlı konuşmalar ve gerçek zamanlı uygulamalarda daha yüksek verimlilik sağlar [54]. Avantajlarına rağmen, RAG uygulamasının bazı zorlukları vardır. Etkili bir uygulama, yüksek geri çağırma ve kesinlik sağlamak için güçlü retriever modellerini ve veri kaynaklarını verimli bir şekilde yapılandırıp indekslemek için gelişmiş ön işleme tekniklerini gerektirir. Ayrıca, hassas alanlarda güvenlik ve gizliliğin korunması kritik bir konu olmaya devam etmektedir [52], [54]. RAG, geleneksel üretken modelleri geliştiren güçlü bir çerçeve olarak öne çıkmaktadır ve metin üretim görevlerine dış bilgi entegrasyonu sağlar. Farklı sektörlerdeki uygulamaları, veri odaklı karar alma ve otomasyon süreçlerini devrim niteliğinde değiştirme potansiyelini ortaya koymaktadır. Gelecekteki araştırma yönleri arasında daha iyi alma mekanizmalarının geliştirilmesi, hesaplama yükünün azaltılması ve RAG'ın çok dilli ve çok modelli bağlamlara uygulanabilirliğinin genişletilmesi yer almaktadır.

Bir modelin başarısı, veri setinin büyüklüğüne, çeşitliliğine ve doğruluğuna bağlıdır. Örneğin, bir DDİ modelinin dil anlama kapasitesini artırmak için büyük ve çeşitlendirilmiş metin veri setleri kullanılmaktadır. Veri setleri genellikle eğitim sırasında üç gruba ayrılmaktadır: Eğitim (training), Doğrulama (validation) ve Test (testing). Eğitim aşaması, modelin öğrenme sürecinde kullanılırken, doğrulama aşaması modelin aşırı öğrenmesini (overfitting) önlemek için kullanılmaktadır. Test aşaması ise, modelin genel performansını değerlendirmektedir. Model eğitimi sırasında, veriler matematiksel olarak işlenmekte ve modelin öğrenmesini sağlayacak şekilde optimize edilmektedir. Eğitim sürecinin temel unsurları kaybı hesaplama (loss), optimizasyon (optimization), adımlama (epoch), parçalama (batch) ve aşırı öğrenme (overfitting) şeklinde sıralanabilmektedir.

Model eğitimi gerçekleştirilirken, verilerin tamamı aynı anda eğitime dahil

edilmemektedir. Veri seti, belli sayıda parçalar/yığınlar (batch) halinde eğitimde yer almaktadır. Her bir parçada ne kadar veri alınacağı yığın sayısı (batch size) parametresi ile ayarlanmaktadır. Bu paketlerin ilkinde eğitim tamamlanıp modelin başarımı test edilmekte, başarım değerine göre geriyeyayılım (backpropagation) ile ağırlıklar güncellenmektedir. Bu adımdan sonra yeni eğitim kümesi ile model tekrar eğitilerek ağırlık tekrar güncellenmektedir. Bu işlem, her bir eğitim adımında tekrarlanarak model için en uygun ağırlık değerleri hesaplanmaya çalışılmaktadır. Bu eğitim adımlarının her birine eğitim devri (epoch) adı verilmektedir. Genelde başarım oranı, epoch değeriyle doğru orantılı olarak, düşükten yükseğe doğru bir eğim göstermektedir. Eğitimde, bir sonraki adımın öncekine oranla ne kadar hızlı veya yavaş öğrenmesi gerektiğini öğrenme oranı (learning rate) parametresi ile belirlenmektedir. Modelin eğitim verilerine aşırı derecede uyum sağlaması durumu ise aşırı öğrenme (overfitting) olarak adlandırılmaktadır. Bu durum gerçekleştiğinde model, eğitim verilerine mükemmel bir şekilde uyum sağlamakla kalmaz, aynı zamanda yeni ve görünmemiş veriyle karşılaştığında hatalı veya yanıltıcı tahminlerde bulunabilmektedir. Bu durumun önüne geçebilmek için veri setindeki adeti yüksek tutmak ve bu veriler arasında çeşitlilik sağlamak gibi çözümler uygulanmaktadır.

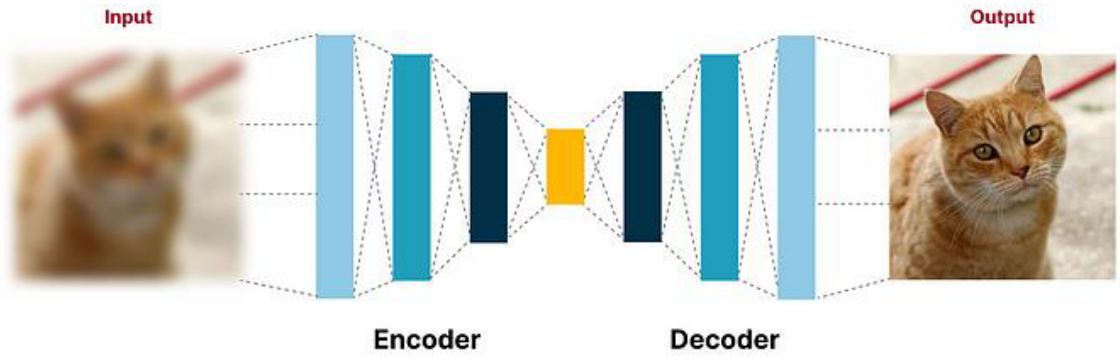
Veri setleri, yapay zeka model eğitiminin temel yapı taşıdır ve doğru bir şekilde hazırlanmış veri setleri, modellerin başarısını büyük ölçüde etkiler [92], [93]. İlk olarak makine öğreniminin temel taşlarını oluşturmak için geliştirilen veri setleri, modellenen problemin doğasına göre farklı biçimlerde ve türlerde olabilir. Örneğin, görsel veri setleri genellikle görüntü tabanlı modellerin eğitimi için kullanılırken, metin ve sekans veri setleri, DDİ veya zaman serisi analizleri için uygundur [94]. Etkili bir YZ modeli geliştirmek için kullanılan veri setleri, modelin eğitimi, doğrulaması ve test edilmesi aşamalarında kilit rol oynar. Yüksek kaliteli veri setleri, modellerin genel performansını artırırken düşük kaliteli veri setleri, yanlış genelleme ve hatalı sonuçlara yol açabilmektedir. Veri setlerinin doğruluğu, eksiksizliği ve dengesi, modellerin başarısını doğrudan etkiler. Eğitim verilerinde dengesizlik, özellikle azınlık sınıflarının doğru bir şekilde temsil edilemediği durumlarda modellerin performansını olumsuz etkileyebilir. Bu gibi durumlarda, SMOTE (Synthetic Minority Oversampling Technique) ve ADASYN (Adaptive Synthetic Sampling) gibi veri dengeleme teknikleri kullanılabilir [93].

Veri setleri genellikle veri temizleme, normalizasyon ve standartlaştırma, öznitelik seçimi

ve indirgeme gibi aşamalardan geçmektedir. Veri temizleme aşamasında eksik ve yanlış verilerin düzeltilmesi veya kaldırılması işlemleri uygulanır. Normalizasyon ve standartlaştırma adımında, değerlerin ölçeklenerek uygun bir dağılıma getirilmesi sağlanır. Öznitelik seçimi ve indirgeme adımında ise, modelin daha verimli çalışmasını sağlamak için gereksiz veya fazla detaylı verilerin elenmesi sağlanır [93]. Veri setinin boyutu, modelin genelleme yeteneğini etkilemektedir. Daha büyük veri setleri, modellerin çeşitli durumlar için daha iyi performans göstermesini sağlarken, büyük veri setlerinin işlenmesi daha fazla hesaplama kaynağı gerektirmektedir. Eğitim verilerinin çeşitliliği, modelin yeni durumlarda adaptasyon yeteneğini artırır ve gerçek dünya problemlerine uygulanabilirliği geliştirir [95]. Veri seti hazırlanırken verilen kaynaklar doğrultusunda, veri hazırlama süreçlerine gereken özen gösterilmeli ve bu süreçlerin modeli genel performansı nasıl etkilediği dikkatlice analiz edilerek iyi bir model eğitim süreci gerçekleştirilebilir.

2.1. ÜRETKEN YAPAY ZEKA

Üretken Yapay Zeka (ÜYZ, Generative AI – GenAI), içerik üreten / oluşturan YZ dil modellerini ifade etmektedir. Bu modelleri kullanarak metin, görsel, müzik ve video gibi çeşitli alanlarda içerik üretmek mümkündür. Özellikle DDİ ve görüntü işleme alanlarında büyük başarılar elde etmektedir. ÜYZ, çok büyük veriler kullanarak eğitilen ve sonrasında bu verileri analiz ederek yeni bir veri oluşturabilen dil modellerini kapsamaktadır. Temelinde derin öğrenme tekniklerini barındıran ÜYZ, bu tekniklerle birlikte Yapay Sinir Ağları (YSA, Artificial Neural Network – ANN) kullanılarak çok karmaşık ilişkiler öğrenilir ve veriler arasındaki desenler keşfedilir. YSA'nın bu yeteneği, bilgisayarların bir fotoğrafta ne olduğunu veya bir hikâyede neler olduğunu anlamasını sağlar. Bu süreçte derin öğrenme teknikleri devreye girmektedir. Derin öğrenme, bilgisayarlarında veriler arasındaki desenleri bulmasına yardımcı olmaktadır. Böylece, YZ modelleri öğrenme yeteneği kazanarak yeni veriler üretebilmektedir [56]. Bu modeller genellikle kodlayıcı (encoder) ve kod çözücü (decoder) olmak üzere iki işlemden geçmektedir. Kodlayıcı, veriyi inceleyip anlayarak özel bir şekilde temsil etme görevi üstelenir. Kod çözücü ise, bu temsil bilgisini alarak buradan yeni bir veri oluşturabilmektedir. Bunun gösterimi *Şekil 2.1*'de yapılmaktadır [57].



Şekil 2.1. Üretken Yapay Zeka dil modellerinin kodlayıcı ve kod çözücü örneği.

Şekil 2.1'de görüldüğü üzere, kodlayıcıya bir kedi fotoğrafı ile bu görselin bir kediye ait olduğu bilgisi girdi (input) olarak verilmektedir. Kodlayıcı bu girdiden kedinin nasıl göründüğünü öğrenir. Daha sonrasında bu öğrendikleri ile yeni bir kedi görselini (output) kod çözücü üzerinden oluşturabilmektedir. Bu yöntemle dil modeli eğitimi gerçekleştirilmektedir.

YZ konusundaki en büyük adımlardan biri, OpenAI [26] tarafından geliştirilen ChatGPT'nin 2020'li yıllarda tanıtılmasıyla atıldı [27]. ChatGPT, BDM temelinde, metin tabanlı etkileşimlerde insan benzeri yanıtlar üretebilen bir sohbet botu olarak tanındı. Verilen cevaplardaki "akıllılık" ve insan konuşmasına benzerlik şaşırtıcı hale geldiğinden kısa sürede ilgi çekti. Bu gelişme, yalnızca DDİ alanında değil, geniş bir yelpazede uygulama alanlarında devrim yarattı. Sağlık [21], [22], [23], yazılım [24], [25], eğitim [18], [19], [20], sistem güvenlik [58], [59], sosyal medya [60] gibi birçok alanda metinsel ve görsel içerik üretimi yeteneğiyle hızlı bir şekilde benimsenen ChatGPT, YZ'nin günlük hayatımıza ne kadar entegre olabileceğini göstermektedir. OpenAI'ın ChatGPT ile başlattığı bu yenilikçi dalga, kısa sürede teknoloji devlerini harekete geçirdi. Microsoft, OpenAI'ın GPT modellerini Azure [61] bulut altyapısına entegre ederek, YZ destekli iş çözümleri için önemli bir adım attı. Ayrıca, Microsoft'un Office ürünlerine YZ entegrasyonu, kullanıcıların üretkenlik süreçlerini tamamen dönüştürmeye yönelik bir atılım olarak karşımıza çıktı. Google ise, kendi büyük dil modellerini geliştirme yolunda ilerleyerek Bard [62] adlı bir sohbet botu tanıttı ve PaLM [63] modeli ile birçok sektöre çözümler sundu. Daha sonra bunu daha da geliştirerek adını Gemini [64] olarak değiştirdi.

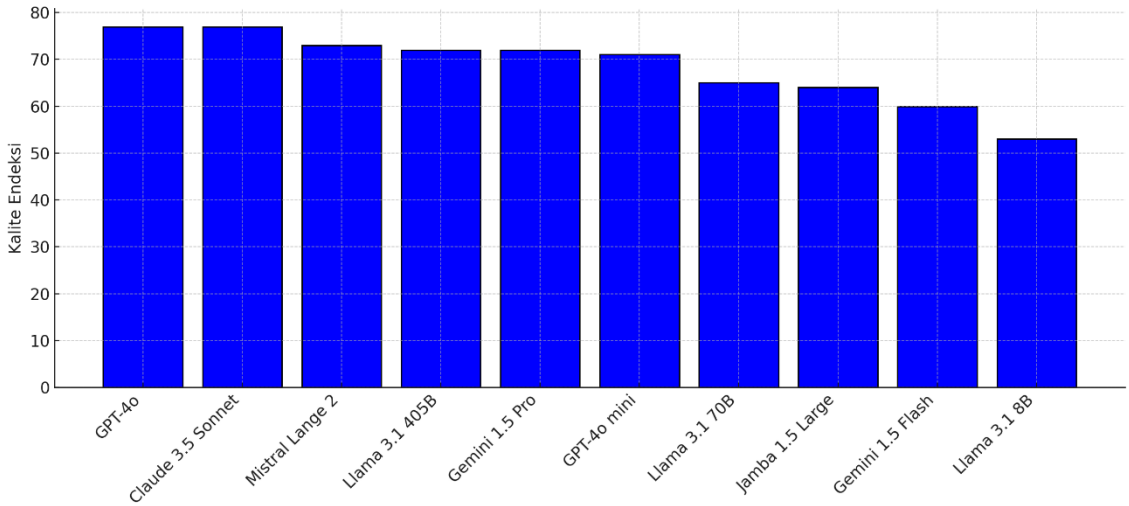
Meta'nın geliştirdiği LLaMA modelleri [48], açık kaynak kodlu yapısıyla araştırmacılar ve geliştiriciler için yeni fırsatlar sundu. Meta, bu modelleri daha düşük kaynak gereksinimleriyle yüksek performans sunacak şekilde optimize ederek erişimi

kolaylaştırdı. Alibaba'nın Qwen modeli, çok dilli yeteneği ile öne çıkmakta ve çeşitli amaçlar için tasarlanmış birçok versiyonu bulunmaktadır. Özellikle Qwen2.5-Coder [46] versiyonu ile kodlamadaki gücünü gösterdi. Bu konuda diğer bir isim olarak Anthropic'in Claude modeli öne çıkmaktadır. Olumlu ve yararlı diyaloglara odaklanan Claude [30], güvenlik ve veri gizliliği konularındaki hassasiyetinden dolayı dikkat çekmektedir.

Tüm bu gelişmeler, yazılım mühendisliğini yeniden şekillendirmekte ve gelecekte daha da önemli yeniliklerin kapısını aralamaktadır. Bununla birlikte, YZ'nin üretken ve yaratıcılığı sayesinde, kullanıcıların yenilikçi ürün geliştirmesi daha kolay hale geldi. YZ, BDM, DDİ ve kodsuz platformlar gibi teknoloji ve yöntemlerin gelişimi, yalnızca yazılım geliştirme süreçlerini dönüştürmekle kalmayarak, aynı zamanda teknoloji kullanımını ve yazılım geliştirmeyi daha geniş kitleler için erişebilir kılmaktadır.

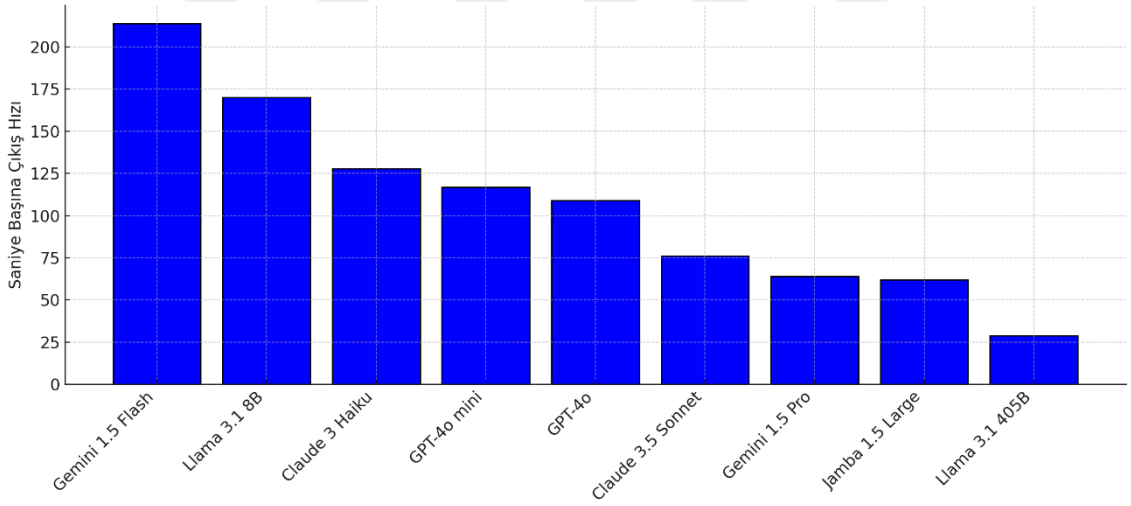
Büyük Dil Modelleri, milyonlarca veriyle eğitilerek yeni çıktılar üretebilme kapasitesine sahiptir. Bu veriler arasında açık kaynak yazılım kodları, bu alanda yazılmış akademik makaleler, belgeler ve kitaplardaki teknik bilgiler de bulunmaktadır. Bu verilerle eğitilen modeller, doğal dil ile verilen girdilerden kod üretebilme kabiliyeti kazandı. Yazılım kod hatası çözümlemeden, sıfırdan kod yazdırma, test senaryoları oluşturma veya geliştirilmiş bir yazılım projesi için teknik dokümanlar hazırlama gibi çok çeşitli kullanım alanları bulunmaktadır. BDM ile desteklenen YZ uygulamalarını bu tür alanlarda kullanarak geliştirici verimliliği ve üretkenliğini artırmak mümkün.

YZ ile kod üretimi konusunda kurumsal yazılım geliştirmede GitHub Copilot ve Microsoft Copilot yapay zekaları öne çıkarken, bireysel kullanımlarda CodeQwen1.5, Llama3 ve Claude 3 Opus gibi dil modellerini kullanan YZ ürünlerini görmek mümkün. Bu dil modelleri kod üretimi konusunda iyi sonuçlar üretebilmektedir [65]. Bu konuda popüler durumda olan YZ dil modelleri için kalite, hız ve maliyet bakımının karşılaştırılma grafikleri Şekil *2.2*, Şekil *2.3* ve Şekil *2.4*'te gösterilmektedir.



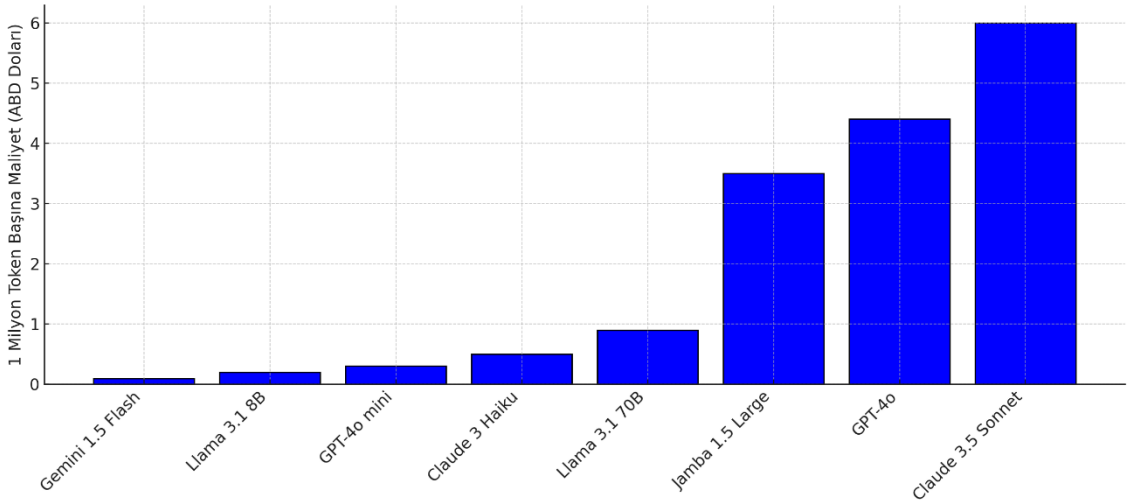
Şekil 2.2. Yapay Zeka dil modellerinin kalite indeksi grafiği.

Şekil *2.2* [66], YZ modellerinin kalite endeksini göstermektedir. Buradaki yüksek değerler daha iyi performansı temsil etmektedir. Bu indekse göre GPT-4o ve Claude 3.5 Sonnet 77 puan ile en yüksek kalite değerini alırken, Mistral Lange 2, Llama 3.1 405B ve Gemini 1.5 Pro yakın değerlerle bu sırayı takip etmektedir.



Şekil 2.3. Saniye başına çıkış hızı için YZ modellerinin ölçülen değerleri.

Şekil *2.3* [66] model başına saniyede üretilen çıktı belirteçlerinin sayısını göstermektedir. Bu sayı ne kadar yüksekse model o kadar hızlı çalışıyor demektir. Burada Gemini 1.5 Flash 214 belirteç ile en yüksek değere sahiptir ve en yakın rakibi olan Llama 3.1 8B modelinin 44 belirteç önündedir.



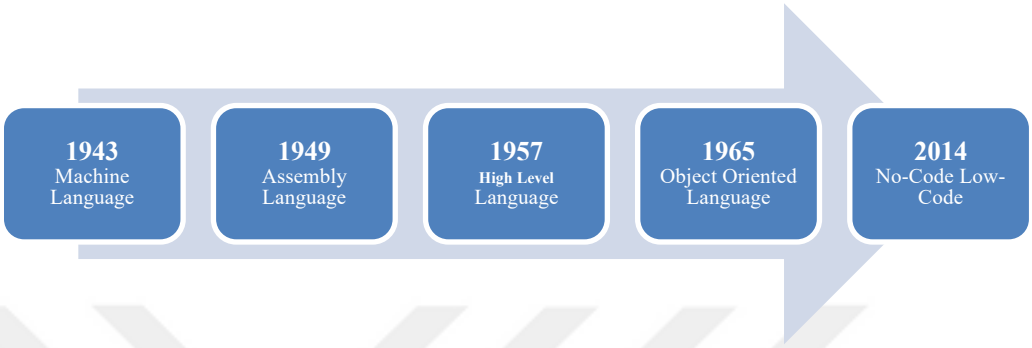
Şekil 2.4. YZ Modellerinin 1 milyon token başına harcanan ABD doları maliyet grafiği.

Şekil *2.4* [66] her bir milyon token için modellerin maliyetini ABD doları cinsinden göstermektedir. YZ dil modellerinde "token", bir metni işlemek için daha küçük parçalara bölünmüş birimlerdir [67], [68]. Bu parçalar bir kelime, karakter veya alt kelime olabilir. Modeller, metni anlamak ve işlemek için bir token'lar üzerinde çalışır. Örneğin, "Merhaba, dünya!" ifadesi token'lara ayrıldığında ["Merhaba", ",", "dünya", "!"] şeklinde olabilir. Bu grafikteki düşük değerler daha az maliyete işaret etmektedir. Gemini 1.5 Flash modeli 0.1 USD ile maliyet verimliliği açısından en iyi performansı göstermektedir. Bu üç grafik, yapay zeka modellerinin maliyet, hız ve kalite gibi farklı boyutlarda nasıl karşılaştırıldığını görsel olarak açıklamaktadır. Grafikler Ağustos 2024 itibariyle en son analiz verilerine göre hazırlanmıştır. Bu değerlere göre en iyi yapay zeka budur demek çok doğru değil ancak en iyisi yapay zekayı ne için kullanacağınıza ve kısıtlarınızın neler olacağına bağlı olarak değişebilir.

2.2. KODSUZ PLATFORMLAR

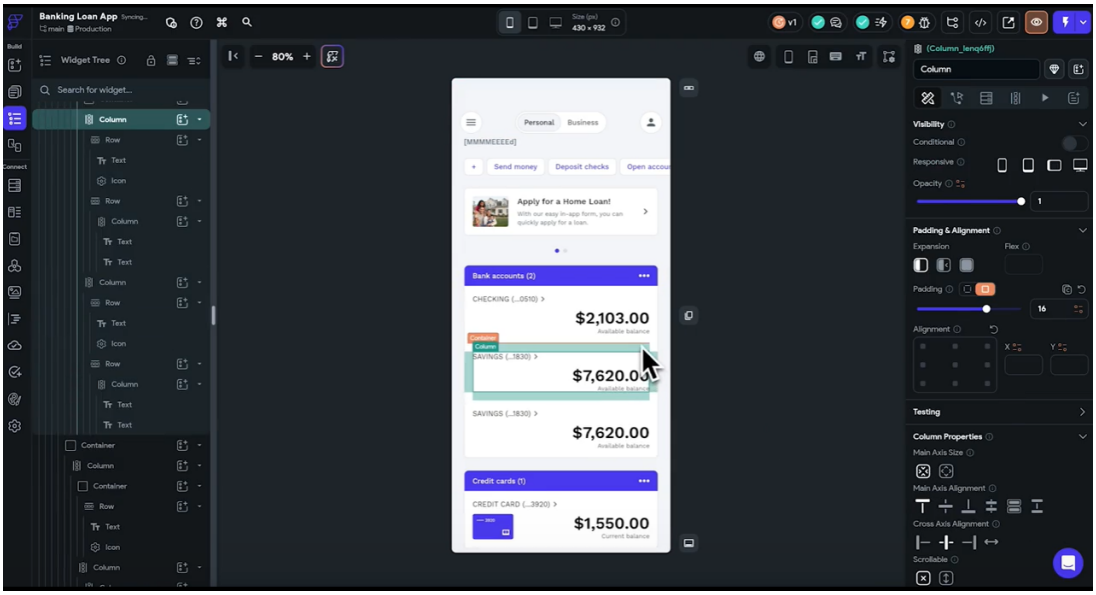
Kodsuz (No Code – NC) platformlar, geleneksel yazılım geliştirme süreçlerindeki zaman ve maliyetleri düşürmek ve teknik bilgisi olmayan kullanıcıların da yazılım geliştirmesine olanak tanımak amacıyla ortaya çıkmıştır. Bu platformlar, kullanıcıların sürükle – bırak arayüzler ve önceden hazırlanmış şablonlar ile hızlı bir şekilde uygulama geliştirmesine olanak tanımaktadır. Kodsuz platformlarla ilgili GitHub CEO'su Chris Wanstrath, "*Kodlamanın geleceği hiç kodlama yapmamaktır.*" ifadesini kullanmıştır [69]. Bu alanda hem akademik hem de deneysel olarak birçok çalışma yürütülmektedir. Moskal (2021) [57], kodsuz teknolojisinin dijitalleşme sürecindeki önemini ve bu teknolojinin

işletmelerde nasıl kullanılabileceğini tartışmaktadır. Van Lunteren (2023) [56], özelleştirilmiş nesne algılama modellerini eğitmek ve dağıtmak için kodsuz platformların kullanımını ele almaktadır. Liu ve arkadaşları (2024) [70] ise, evrensel bir kodsuz web tasarım aracının tasarımını ve uygulamasını detaylandırarak, bu tür araçların dijital izleme sistemlerinde verimliliği nasıl artırabileceğini göstermektedir.



Şekil 2.5. Algoritmaların makine kodundan kodsuz / az kodlu yapıya doğru evrimi.

Makine dilinden kodsuz / az kodlu yapılara doğru gerçekleşen evrim Şekil *2.5*'da gösterilmektedir [69]. Bu kapsamda, günümüzde kodsuz geliştirme fırsatları sağlayan birçok platform ve yazılım ürünü bulunmaktadır; Mendix, Xano, Zoho Creator, Webflow, Google AppSheet, ServiceNow App Engine, Zeroqode, Glide, Thunkable, Adalo, Blockly, Logotec App Studio ve FlutterFlow bunlardan birkaçı olarak sayılabilir. Şekil *2.6*'de FlutterFlow platformunun ekran görüntüsü bulunmaktadır. FlutterFlow, kodsuz olarak mobil uygulama geliştirmeye olanak tanıyan kodsuz bir platformdur.



Şekil 2.6. Kodsuz mobil uygulama geliştirilen FlutterFlow arayüzü.

Kodsuz platformlara yapay zeka desteğinin gelmesi ise bu süreci bir adım daha ileriye taşıma potansiyelindedir. YZ, kodsuz platformların yeteneklerini artırarak, kullanıcı deneyimini daha dinamik ve verimli hale getirmektedir. BDM ise bu değişimin merkezinde yer almaktadır; DDİ teknikleri kullanılarak, kullanıcıların basit metinsel ifadelerle karmaşık işlevler oluşturabilmesini sağlamaktadır. Bu gelişmeler, özellikle teknik bilgisi olmayan kişilerin yazılım geliştirme süreçlerine daha aktif katılımını sağlamakta ve aynı zamanda işletmeler için de önemli avantajlar sağlamaktadır. Geleneksel yazılım geliştirme süreçlerinde karşılaşılan maliyet ve zaman baskılarını azaltarak, KOBİ ve girişimcilerin dijital çözümler geliştirmesini kolaylaştırmaktadır. Özellikle birkaç kişilik iş gücüne sahip girişimci şirketlerinin daha az iş gücü ile üretim yapmasında kolaylık sağlamaktadır. Büyük ölçekli şirketlerse ise bazı görevlerin otomatize edilmesini ve kolaylaştırmasını sağlamaktadır.

Kodsuz ve az kodlu geliştirme olanağı tanıyan platformlar yazılım geliştirme süreçlerini büyük ölçüde kolaylaştırmış olsa da gelişmiş olan YZ ve BDM'nin gerisinde kaldı. Bu nedenle de YZ tabanlı kodsuz platformlar ihtiyaç haline geldi. Amazon İcra ve Yönetim Kurulu Başkanı (CEO) Andy Jassy, Ağustos 2024'te paylaştığı bir yazısında [71] YZ tabanlı kod geliştirmenin öneminden bahsetmektedir. Andy, yazısında yazılım geliştirme asistanı olarak geliştirdikleri Amazon Q'yu kullanarak bir uygulamayı birkaç saatte Java 17'ye dönüştürdüklerini fakat geleneksel yazılım geliştirme yöntemi ile yaptıkları taktirde 50 geliştiricinin günlük eforu ile yapılacağını dile getirdi. Buna bağlı olarak, YZ destekli yazılım geliştirme asistanı ile yılda 4500 geliştiricinin günlük çalışma maliyeti kadar tasarruf sağladıklarını vurguladı. Amazon Q, şu an için kodsuz bir platform sunmamaktadır, fakat bu yapıya benzer olarak Amazon SageMaker ürünü mevcut. Diğer YZ tabanlı kodsuz platformlara örnek olarak Buildfire AI, Akkio, DataRobot, Obviously AI, Google Teachable Machine, Lobe AI, Nanonets, Levity AI, Causaly AI, PredictNow AI, Invideo, AI Squared, E42, Flagright ve CallFluent AI gibi birçok platform verilebilir [72].

BuildFire AI [73], kod bilgisi olmadan, herkesin mobil uygulama geliştirmesine olanak tanıyan YZ destekli bir kodsuz platformdur. Platform, YZ desteği sayesinde diğer kodsuz tabanlı platformlara oranla çok daha hızlı şekilde uygulama geliştirme olanağı sunmaktadır. İşletme ile ilgili bazı sorulara verilen cevaplarla otomatik olarak ihtiyaç duyulan uygulama YZ tarafından oluşturulmaktadır. Hatta uygulamanın şirket markasıyla uyum içinde olması açısından şirketin web sitesindeki logoyu ve renkleri kullanmaktadır.

Böylece sadece tasarım için bile haftalarca uğraşmak yerine saniyeler içinde YZ ve kodsuz geliştirme olanaklarıyla bunu halletmek mümkündür. Geliştirilen mobil uygulamayı iOS ve Android işletim sistemlerinde yayınlama imkanı sunmaktadır. Akkio [74], YZ destekli bir sohbet botu oluşturmayı sağlar. Bu sohbet botunu kullanarak birçok iş kolayca yapılabildiği gibi, kodsuz geliştirme olanağını kullanarak sohbet botunu kendinize göre özelleştirmek mümkün. Bununla birlikte kod yazmadan bazı rapor ve grafikleri saniyeler içinde oluşturabilmektedir. Örneğin, "Aylık satışlarla ilgili bir grafik oluştur" gibi basit komutlarla detaylı grafikler oluşturabilmektedir. Böylece rapor oluşturmak için harcanan yüksek eforlardan zaman kazancı sağlanmaktadır. DataRobot [75], iş sonuçlarını iyileştirmek için üretken yapay zeka ve tahmine dayalı iş akışlarını kullanmaktadır. Platform, bankacılık, fintech, sağlık hizmetleri, sigorta, üretim ve havacılık dahil olmak üzere çok çeşitli sektörlerde kullanılmaktadır. Örneğin, hastaneler ve sağlık uygulamaları ile hangi hastaların kabul edilme olasılığının daha yüksek olduğunu tahmin etmek için kullanılabilir. Böylece doktorlar proaktif olarak sürece katılım sağlayabilir. Bir sigorta şirketi, belirli müşterilere çapraz satış yaparken hangi ürünlerin başarılı olma olasılığının en yüksek olduğunu tahmin etmek için kullanabilir. Depolar, kaynak planlamasını optimize etmek için trafiği tahmin edebilir. DataRobot, kod gerektirmeden karmaşık birçok iyi kolaylaştırmaktadır.

Her alanda birçok ürünü bulunan Google, kodsuz geliştirme alanında da geri durmayarak Teachable Machine [76] adlı çözümünü sundu. Bu platform, YZ dil modeli eğitimi ve üretimi için makine öğrenimi kütüphanesi olan Tensorflow'u kullanmaktadır. Teachable Machine kullanılarak kameradan ve dosya yükleme ile alınan görüntülerden kolayca veri seti (dataset) oluşturulabilir. Daha sonra bu veri seti ile herhangi bir kodlama yapmadan dil modeli oluşturup kullanma olanağı sunmaktadır. Örneğin, kameradan farklı açılarla taratılan bir yüzden veri seti ile modeli oluşturacaktır. Oluşturduğu modelin kodlarını az kodlu (Low Code – LC) yöntemi ile düzenleme imkanı bulunmaktadır. Makine öğrenimi alanlarında çalışmalar yapan kişiler için kolaylaştırıcı bir platform olarak öne çıkmaktadır. Microsoft tarafından geliştirilen Lobe AI [77], Google Teachable Machine platformuna oldukça benzer ve rakip olarak değerlendirilecek niteliktedir. Ağustos 2024 itibariyle, yalnızca görüntü sınıflandırması kullanarak çalışıyor; modeli eğitmek için görüntüleri etiketleme yapılmakta fakat nesne algılama ve veri sınıflandırma modellerinin ileriki zamanda olacağı bilgisi mevcut. Ayrıca, Lobe AI, proje için en iyi makine öğrenimi mimarisini otomatik olarak seçen proje şablonlarına sahiptir.

Amazon SageMaker [78], kurumsal şirketler için tasarlanmış, bulut tabanlı, YZ destekli bir kodsuz platformdur. Bu platform, veri bilimciler, mühendisler ve geliştiricilere kolayca veri işleme, model oluşturma, model eğitme ve dağıtma imkanı sağlamaktadır [79]. Kodsuz geliştirme olanağı ile teknik bilgisi olmayan kullanıcılar bile makine öğrenimi alanlarında çalışmalar yapabilirken, az kodlu özelliği ile de teknik bilgisi olan kullanıcılar platform tarafından oluşturulan kodları düzenleyerek iyileştirme veya özelleştirmeler yapabilmektedir. Platform, hizmet ve kullanım açısından Google'ın Teachable Machine ve Microsoft'un Lobe AI ürünlerine benzerdir. SageMaker'ın en büyük avantajlarından biri, platform üzerinde geliştirdiğiniz modelinizi dağıtma olanağıdır. Böylece bunu bir web üzerinden paylaşma ve uygulamalarda kullanma özgürlüğü sunmaktadır. Levity AI [80], yüklenen veri setini kullanarak YZ ve makine öğrenimi modelleri oluşturmayı sağlayan kodsuz bir iş akışı otomasyon platformudur [81]. Kullanıcılar, Levity AI ile belirli tetikleyicilere dayalı olarak ortak görevleri tamamlamak için YZ destekli bir model oluşturabilir. CRM veri tabanları, e-posta sağlayıcıları ve yönetim sistemleri gibi halihazırda kullanılan sistemlere bağlayıp, YZ blokları oluşturarak iş akışı oluşturulabilir. Böylece, sürekli yapılan görevler için kolayca bir otomasyon uygulaması yapmak mümkündür. AI Squared [82], web tabanlı uygulamaları desteklemek için hem öngörücü hem de üretken yapay zeka modelleri kullanır. Platform, halihazırda kullanılan web uygulaması üzerinde uygulanabilir. Var olan web uygulamasını AI Squared üzerinde YZ destekli az kodlu yöntemi kullanarak, hızlıca düzenleme ve geliştirme olanağı tanımaktadır. Bu yönüyle diğer birçok yapay zeka platformundan farklıdır. CallFluent AI [83], kod yazmadan bir dakika içinde YZ destekli sesli çağrı temsilcisi oluşturabilen bir platformdur. Oluşturulan bu temsilciler, müşteri sorularını yanıtlama veya randevu ayarlama gibi görevleri yöneterek iletişimi daha sorunsuz ve daha verimli hale getirebilmektedir. Platform, esnek bir yapıya sahip olduğundan, işletmelerin kendi özel ihtiyaçlarına göre ek geliştirmeler yapabilmelerine olanak tanımaktadır. Günlük görevleri otomatikleştirmek için zamandan tasarruf sağlanacak birçok konuda kullanılabilir. Tüm bu olanaklar, işletmelerde maliyetleri düşürmek için oldukça önemlidir. YZ tabanlı kodsuz platformların yazılım geliştirme süreçlerine etkilerini tam anlamıyla kavrayabilmek için bu platformların hem geleneksel kodsuz platformlarla hem de kendi aralarında kıyaslanması gerekmektedir. Bu bölümde, YZ tabanlı platformların geleneksel platformlara kıyasla sundukları avantajlar, farklı YZ tabanlı uygulamalar arasındaki farklar ve bu platformların çeşitli kullanım alanlarına göre nasıl değerlendirildiği ele alınacaktır.

Çizelge 2.1. YZ destekli ve geleneksel kodsuz platformların karşılaştırılması.

Özellik	**YZ Destekli Kodsuz Platform**	**Geleneksel Kodsuz Platform**
Kullanım kolaylığı	Doğal dil komutlarıyla daha kolay kullanım	Sürükle-bırak arayüzü, kullanımı kolay ama öğrenme süreci var
Özelleştirme yeteneği	Yüksek – YZ, karmaşık ve özel ihtiyaçları anlayabilir.	Hızlı – Manuel bileşen seçimi ve yapılandırma gerektirir.
Geliştirme hızı	Çok Hızlı – YZ, istekleri hızlı bir şekilde koda dönüştürür.	Ortak – Bazı manuel müdahaleler gerekebilir.
Otomasyon seviyesi	Yüksek – Rutin görevler otomatik olarak gerçekleştirilir.	Orta – Bazı manuel müdahaleler gerekebilir.
Kod kalitesi	Orta – YZ, en iyi uygulamaları ve güncel standartları uygular. Fakat hatalı kod yazma ihtimali bulunur.	Değişken – Platform ve kullanıcı becerilerine bağlıdır.
Öğrenme eğrisi	Düşük – Doğal dil kullanımı sayesinde hızlı adaptasyon.	Orta – Platform özelliklerini öğrenme gereksinimi.
Maliyet	Başlangıçta yüksek, uzun vadede düşük.	Genellikle daha düşük başlangıç maliyeti.
Ölçeklenebilirlik	Yüksek – YZ, ihtiyaçlara kolayca entegrasyon sağlar.	Orta – Manuel ölçeklendirme gerektirebilir.
Entegrasyon kabiliyetleri	Geniş – YZ, çeşitli sistemlerle kolay entegrasyon sağlar.	Sınırlı – Önceden tanımlanmış entegrasyonlarla sınırlı.
Hata ayıklama ve sorun giderme	Gelişmiş – YZ, hataları tespit edip düzeltebilir.	Manuel – Kullanıcının hataları bulup düzeltmesi gerekir.
Veri güvenliği	Orta – YZ kullanımı ek güvenlik önlemleri gerektirebilir.	Yüksek – Daha az karmaşık, daha kolay güvenlik sağlanabilir.
Güncellemeler ve bakım	Otomatik – YZ, sürekli öğrenme ve iyileştirme sağlar.	Manuel – Periyodik güncellemeler ve bakım gerektirir.

YZ tabanlı kodsuz platformlar, geleneksel kodsuz platformlara kıyasla önemli avantajlar sunmaktadır. Geleneksel platformlar, kullanıcıların sınırlı teknik bilgiyle basit uygulamalar geliştirmelerine olanak tanırken, YZ tabanlı platformlar ise karmaşık işlevleri de mümkün kılmaktadır. Örneğin, Lobe AI ve Amazon SageMaker gibi YZ

destekli platformlar, makine öğrenimi modellerini entegre etmeyi ve otomatikleştirmeyi mümkün kılarken, geleneksel kodsuz platformları genellikle bu tür özellikleri sunmamaktadır. Ayrıca, YZ destekli platformların kullanıcı arayüzü daha sezgisel ve kullanıcı dostudur; bu da geliştirme sürecini hızlandırır ve maliyetleri düşürür. Çizelge 2.1'de YZ destekli kodsuz platformlar ile geleneksel kodsuz platformlar arasındaki temel farklar gösterilmektedir. YZ destekli platformlar, DDİ yetenekleri sayesinde daha kolay kullanım, daha hızlı geliştirme süreci ve daha yüksek özelleştirme olanakları sunmaktadır. Ayrıca otomatik hata ayıklama ve sürekli öğrenme özellikleri ile uzun vadede verimli olabilmektedirler.

Çizelge 2.2. YZ destekli kodsuz platformların karşılaştırılması.

Platform	**Ana Odak**	**Temel Özellik**	**Hedef Kitle**	**Kullanım Alanları**
Amazon Q	Genel YZ asistanı	DDİ, veri analizi, kod üretimi	İşletmeler, geliştiriciler	İş zekası, yazılım geliştirme
BuildFire AI	Mobil uygulama geliştirme	Otomatik kod üretimi, tasarım önerileri	KOBİ'ler, girişimciler	Mobil uygulama oluşturma
Akkio	Tahmine dayalı analitik	Otomatik makine öğrenimi, veri görselleştirme	Veri analistleri, işletmeler	Müşteri analizi, risk değerlendirme
DataRobot	Kurumsal YZ platformu	Otomatik makine öğrenimi, model dağıtımı	Büyük işletmeler	Tahmine dayalı modelleme, veri bilimi
Obviously AI	Tahmine dayalı analitik	Doğal dil sorguları, hızlı model oluşturma	KOBİ'ler, veri analistleri	Satış tahmini, müşteri segmentasyonu
Google Teachable Machine	Görüntü sınıflandırma	Kolay model eğitimi, web tabanlı arayüz	Eğitimciler, öğrenciler	Eğitim projeleri, basit YZ uygulamaları
Lobe AI	Görüntü sınıflandırma	Sürükle – bırak model eğitimi, kolay dışa aktarma	Geliştiriciler, tasarımcılar	Görsel tanıma uygulamaları
Causaly AI	Biyomedikal araştırma	Nedensel çıkarım, literatür analizi	Araştırmacılar, ilaç şirketleri	İlaç keşfi, hastalık araştırması

Geleneksel kodsuz platformlar, daha düşük başlangıç maliyetleri ve daha basit yapıları ile özellikle küçük ölçekli projeler için cazip bir seçenek olmaya devam etmektedir. Her

iki yaklaşımın da kendine göre avantajları ve kullanım alanları bulunmaktadır. İşletmeler kendi ihtiyaçlarına, teknik kabiliyetlerine ve bütçelerine göre en uygun platformu seçmelidir. Gelecekte, YZ teknolojilerinin gelişmesiyle birlikte, bu iki yaklaşımı birleştiren daha gelişmiş hibrit platformların ortaya çıkması beklenebilir. Bu tez de bu beklentiyi hedef alan bir çalışma içermektedir. Tez kapsamında geliştirilen YZ destekli kodsuz platform ile küçük ölçekli projelerin düşük maliyetle geliştirilmesi gibi olanaklar sunulmaktadır. BuildFire AI, Akkio ve DataRobot gibi platformlar farklı işlevler sunar. Bu platformlar çeşitli kullanım senaryoları için optimize edilmiştir. Örneğin, BuildFire AI özellikle mobil uygulama geliştirme için tasarlanırken, Akkio veri analitiği ve iş zekası çözümleri için optimize edilmiştir. DataRobot ise otomatik makine öğrenimi modellemesinde güçlüdür ve büyük veri kümelerini işleme kapasitesine sahiptir. Platformlara göre bazı özellik karşılaştırmaları Çizelge 2.2'de gösterilmektedir.

Burada gösterilen çeşitli YZ destekli kodsuz platformlar, teknolojinin farklı sektörlere ve iş süreçlerine nasıl entegre edilebileceğini göstermektedir. Her platform, kendine özgü güçlü yönleri ve uzmanlık alanlarıyla öne çıkmaktadır. Kullanıcılar, spesifik ihtiyaçlarına, teknik yeterliliklerine, bütçelerine ve uzun vadeli hedeflerine en uygun platformu seçebilmektedir. Örneğin, kurumsal bir şirket iş zekası ve kod üretimi için Amazon Q'yu tercih edebilirken, girişimci küçük çaplı bir şirket mobil uygulama geliştirmek için BuildFire AI veya FlutterFlow tercih edebilir. Öte yandan, mevcut uygulamalarına YZ yetenekleri eklemek isteyen bir işletme, AI Squared platformunu tercih edebilir. Seçim yaparken, platformun ölçeklenebilirliği, entegrasyon yetenekleri, kullanıcı dostu olması ev sunduğu destek hizmetleri gibi faktörler de göz önünde bulundurulmalıdır. YZ teknolojisinin hızla geliştiği bu dönemde, seçilen platformun sürekli güncellenen ve yeniliklere açık olan bir yapıda olması da önemli bir kriterdir. Sonuç olarak, en verimli YZ platformu, kullanıcının mevcut gereksinimlerini karşılamanın yanı sıra, gelecekteki büyüme ve değişim potansiyelini de destekleyebilen platform olacaktır.

Web geliştirme alanında CSS çerçevelerinin kullanımı son yıllarda giderek artmaktadır. Tailwind CSS, bu çerçeveler arasında özelleştirilebilir yapısı ile öne çıkmaktadır. Bu çerçeve, platformun çıktılarının tutarlı ve profesyonel bir görünüm sunmasını sağlamaktadır. Geleneksel CSS çerçeveleri olan Bootstrap ve Foundation ile karşılaştırıldığında, Tailwind CSS'nin önceden tanımlı daha az bileşen içermesi, hızlı prototipleme ve çok daha hafif CSS çıktıları [84], [85], performans optimizasyonu [86] ve mobil görünüme uyumlu tasarımlar için özelleştirilebilir sınıflar içermesi başlıca

avantajlar arasında yer almaktadır. Bu avantajlar, geliştiricilerin Tailwind CSS'yi özellikle Angular ve React gibi komponent bazlı çerçevelerde kullanımlarını yaygınlaştırmıştır [84].

React, Meta (eski adıyla Facebook) tarafından 2013 yılında açık kaynak olarak sunulan [87], kullanıcı arayüzü oluşturmayı hedefleyen bir JavaScript kütüphanesidir. Bileşen tabanlı yapısı ve performansı odaklı özellikleriyle kısa sürede geniş bir geliştirici topluluğu tarafından benimsenmiştir. React'in temel amacı, web uygulamalarında kullanıcı deneyimini artırmak ve geliştiricilere esneklik sağlamaktır. İlk olarak Facebook'un iç ihtiyaçlarını karşılamak amacıyla geliştirilen React, Instagram gibi büyük platformlarda da kullanılmaya başlanarak gücünü kanıtlamıştır. React'ın en dikkat çekici özelliklerinden biri, bileşen tabanlı yapısıdır [88]. Bu yaklaşım, kullanıcı arayüzlerinin küçük, bağımsız ve yeniden kullanılabilir parçalara bölünmesine olanak tanır. Bu bileşenler, uygulamaların modülerliğini artırarak karmaşık yapıları daha kolay yönetilebilir hale getirir. React ayrıca Virtual DOM adı verilen yenilikçi bir teknoloji kullanır. Geleneksel yöntemlerde yapılan her DOM manipülasyonu tarayıcıda bir yeniden boyama işlemi gerektirirken, Virtual DOM bu işlemi minimize ederek performansı artırır [88]. JSX adı verilen JavaScript uzantısı, HTML benzeri bir sözdizimi kullanarak bileşenlerin oluşturulmasını kolaylaştırır. Bu sayede geliştiriciler, HTML ve JavaScript'i tek bir dosyada birleştirerek daha sezgisel bir kodlama deneyimi yaşar. React, performans avantajları ve kullanıcı deneyimini iyileştiren özellikleriyle öne çıkar. Bileşenlerin yeniden kullanılabilir olması, yazılım geliştirme süreçlerini hızlandırır ve bakım maliyetlerini düşürür. React'in geniş bir ekosistemi ve topluluk desteği, geliştiricilerin karşılaştıkları sorunlara hızlı çözümler bulmasını sağlar [88]. Ayrıca, React Native gibi projelerle mobil uygulama geliştirme süreçlerinde de kullanılabilir hale gelerek kapsamını genişletmiştir. Bununla birlikte, React'in bazı dezavantajları da bulunmaktadır. Kütüphanenin öğrenme eğrisi, özellikle yeni başlayanlar için zor olabilir. JSX'in ve Virtual DOM'un kavramsal karmaşıklığı, deneyimsiz geliştiriciler için başlangıçta kafa karıştırıcı olabilir. Ayrıca, React ekosisteminin hızla değişen yapısı ve sık güncellemeler, uzun vadede uyumluluk sorunlarına yol açabilir. Bu durum, geliştiricilerin sürekli olarak yeni sürümleri takip etmelerini ve projelerini güncellemelerini gerektirir. React, rakipleri Angular ve Vue.js ile karşılaştırıldığında, genellikle daha hafif bir kütüphane olmasıyla dikkat çeker. Angular, daha kapsamlı bir çerçeve sunar, ancak bu durum projeyi daha ağır ve karmaşık hale getirebilir. Vue.js ise

öğrenme kolaylığı ve esneklik sunarak React ve Angular arasında bir denge kurar. Ancak, React'in geniş topluluk desteği ve ekosistemi, çoğu zaman onu tercih edilen bir seçenek haline getirir. React modern web geliştirme dünyasında önemli bir yere sahiptir; bileşen tabanlı yaklaşımı, performans odaklı yapısı ve geniş topluluk desteğiyle, web geliştirme süreçlerini daha verimli hale getirmiştir.

Python, modern yazılım geliştirme ve bilimsel araştırma dünyasında yaygın olarak kullanılan, yüksek seviyeli, genel amaçlı bir programlama dilidir [89]. Guido van Rossum tarafından 1989 yılında geliştirilmeye başlanmış ve ilk kez 1991 yılında yayımlanmıştır [89], [90], [91]. Python, kullanıcı dostu tasarımı, geniş kütüphane ekosistemi ve çok yönlülüğü ile kısa sürede geniş bir kullanıcı kitlesine ulaşmıştır. Bu popülerliği çeşitli teknik özelliklerinden kaynaklanmaktadır. Örneğin, Python'un sözdizimi İngilizce'ye oldukça benzerdir. Bu durum, yeni başlayanlar için dilin öğrenilmesini kolaylaştırır ve kodun daha okunabilir hale gelmesini sağlar. Bunun dışında, Windows, macOS ve Linux gibi farklı platformlarda çalışabilir olması, veri işlemeden dosya yönetimine ve ağ programlamaya kadar geniş ve güçlü bir kütüphaneye sahip olması, açık kaynaklı bir dil olması gibi özellikleri bulunmaktadır. Python hem yapısal hem de nesne yönelimli programlamayı destekler [91]. Bu, dilin farklı uygulama ihtiyaçlarına uyum sağlamasını kolaylaştırır ve esneklik olma avantajı sağlar. Ayrıca yorumlanabilir bir dil olmasından dolayı, derleme aşamasına gerek kalmadan hızlı geliştirme süreçlerine olanak tanır [90] ve bu da Python'un bir diğer avantajı olarak görülebilir. Bunların yanı sıra dezavantajları da bulunmaktadır; yorumlanan bir dil olması nedeniyle, Python, derlenen dillere göre (örneğin C++ ve Java) daha yavaş çalışabilir [91]. Bir diğer dezavantajı ise, mobil uygulama geliştirme için yaygın olarak kullanılan bir dil değildir ve bu alanda rakiplerine göre sınırlı bir rol oynamaktadır [91]. Python, çeşitli alanlarda geniş bir uygulama yelpazesine sahiptir. NumPy ve Pandas gibi kütüphanelere sahip olması, veri analistlerinin ve bilim insanlarının tercih etmelerinde önemli bir rol oynamaktadır [90], [91]. Django ve Flask gibi çerçevelere sahip olması, Python'un web geliştirme için güçlü bir araç olmasını sağlamaktadır [90]. Son olarak, TensorFlow ve Scikit-Learn gibi kütüphaneler, Python'u makine öğrenimi ve yapay zeka konusunda lider bir dil olarak öne çıkarmaktadır [90].

Çeşitli hazır veri setleri ve BDM'lere kolay erişim imkanı tanıyan Hugging Face (HF) [98] platformunda açık kaynak olarak paylaşılan birçok YZ dil modeli bulunmaktadır. GPT, Llama, Qwen dahil olmak üzere geniş yelpazede ve çeşitli boyutlarda birçok model

ve veri seti barındırmaktadır. HF üzerinde üyelik oluşturarak bunlara erişim imkanı elde ederken aynı zamanda geliştirilen model ve veri setlerini de paylaşma imkanı sunulmaktadır. Bu konuda açık kaynak yazılım projelerinin paylaşıldığı GitHub platformuna çok benzerdir. HF, model eğitiminde kullanılmak üzere belirli ücretler karşılığında kiralanan, CPU ve GPU destekli çeşitli sanal sunucular da sunmaktadır. HF, önceden eğitilmiş (pretrained) dil modellerini indirip eğitmeyi kolaylaştıran Transformers isimli bir Python kütüphanesi de sunmaktadır. Bu kütüphane, içerisinde web servisleri (API) ve araçlar barındırarak, hazır metotlar ile kullanım imkanı tanımaktadır. Metotlar yardımıyla önceden eğitilmiş model üzerine çalışılabilir; böylece zaman ve kaynak tasarrufu yapılabilmektedir. Kütüphaneyi Python ile geliştirilen çalışmalarda kullanmak mümkündür. Desteklediği güncel model ve çerçeveler (framework) için HF Transformers dokümantasyonuna [99] bakılabilir.

Google Colab, makine öğrenmesi ve derin öğrenme modellerinin geliştirilmesinde popüler bir araç olarak öne çıkan, bulut tabanlı bir Jupyter Notebook ortamıdır. Kullanıcıların herhangi bir kurulum gerektirmeksizin Python kodlarını bir web tarayıcısı üzerinden çalıştırmasına olanak tanır. Google Colab, özellikle yüksek hesaplama gücü gerektiren projeler için GPU ve TPU desteği sunarak, araştırmacıların pahalı donanımlar olmaksızın derin öğrenme ve makine öğrenmesi gibi yoğun hesaplama gerektiren işlemleri gerçekleştirmelerini sağlar [100], [101]. Colab kullanıcıları, projelerinde Google Drive entegrasyonu sayesinde büyük veri setlerine kolayca erişebilir ve bu veri setlerini bulut üzerinde işleyebilirler. Bu platform, kullanıcı dostu arayüzü, ücretsiz GPU desteği ve kolay paylaşım özellikleri ile eğitimden araştırmaya kadar geniş bir yelpazede uygulama bulur. Örneğin, deri kanseri gibi sağlık sorunlarının derin öğrenme ile tespiti veya mikroekonomi modellemeleri gibi akademik çalışmalar, Google Colab üzerinde başarıyla yürütülebilmektedir [100], [102]. Bu özellikleri ile Google Colab, düşük maliyetli bir çözüm sunarken aynı zamanda eğitimciler ve araştırmacılar için etkili bir öğrenme ve uygulama platformu olma özelliği taşımaktadır.

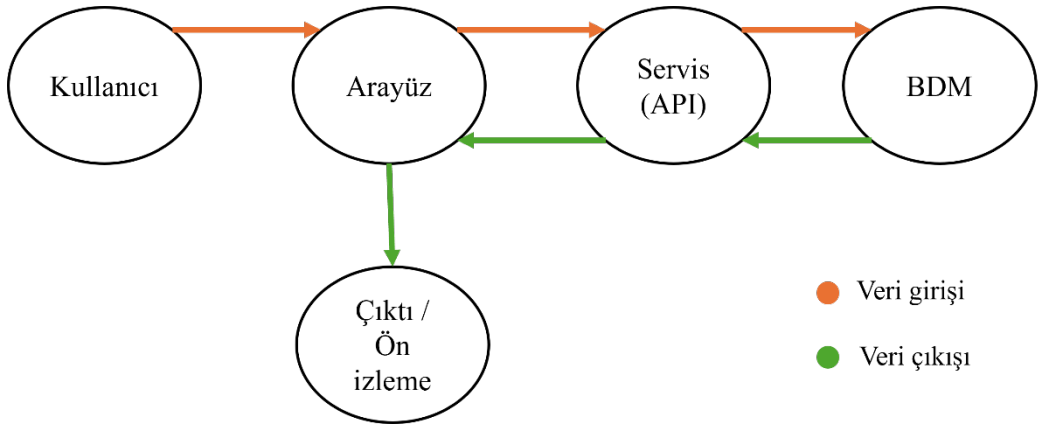
Web geliştirme, günümüzde kullanıcı deneyiminin (User Experience – UX) başarı için merkezi bir konumda olduğu kritik bir alan haline gelmiştir. UX tasarımı, kullanılabilirlik, görsel tasarım ve insan faktörleri gibi çeşitli disiplinleri entegre ederek bütünsel ve etkileyici bir kullanıcı deneyimi oluşturmayı amaçlar. Kullanıcı odaklı bir yaklaşım benimseyen işletmeler, yalnızca müşteri memnuniyetini artırmakla kalmaz, aynı zamanda önemli rekabet avantajları da elde edebilir. Kullanıcı deneyimi, temelde

kullanıcıların hedeflerini, arzularını ve davranışlarını anlamak ve bu doğrultuda ihtiyaçlarını karşılamak üzerine odaklanır. Literatüre göre [96], iyi bir kullanıcı deneyimi üç temel ilkeyle tanımlanır; kullanışlılık, kullanılabilirlik ve arzu edilebilirlik. Bir web sitesi, kullanıcıların hedeflerine ulaşmasına yardımcı olmalı (kullanışlılık), bunu verimli ve etkili bir şekilde gerçekleştirmeli (kullanılabilirlik) ve aynı zamanda duygusal olarak tatmin edici bir deneyim sunmalıdır (arzu edilebilirlik) [96], [97]. Bu unsurlar, kullanıcıların bir web sitesine dair genel algısını şekillendirir ve markaya bağlılıklarını önemli ölçüde etkileyebilir.

Kullanıcı deneyimi tasarımında insan psikolojisi kritik bir öneme sahiptir. İnsanların nasıl algıladığını, okuduğunu ve düşündüğünü anlamak, tasarımcıların kullanıcılarla uyumlu arayüzler oluşturmasına olanak tanır. Örneğin, görsel hiyerarşi, kullanıcıların dikkatini kritik unsurlara yönlendirmeye yardımcı olurken, renk seçimleri markanın mesajıyla uyumlu duygular uyandırır. Tasarımcılar ayrıca kullanıcıların hafıza sınırlamalarını da göz önünde bulundurarak, sezgisel ve düşük bilişsel yük gerektiren arayüzler tasarlamalıdır [97]. Kullanıcı odaklı bir tasarım yaklaşımı, derinlemesine kullanıcı araştırması ile başlar. Persona oluşturma ve bağlamsal gözlem gibi teknikler, tasarımcıların kullanıcıların ihtiyaçlarını ve hayal kırıklıklarını anlamasına olanak tanır. Gualtieri'ye göre [96], başarılı bir kullanıcı deneyimi tasarımı, kullanıcıların yerine geçmeyi gerektirir; geri bildirimlerini dinlemek, davranışlarını gözlemlemek ve geliştirme süreci boyunca hipotezleri sürekli olarak test etmek bu sürecin temel unsurlarıdır. UX'in bir diğer önemli yönü, yazılım geliştirme yaşam döngüsüne (SDLC) entegrasyonudur. UX ilkelerinin geliştirme sürecinin erken aşamalarında benimsenmesi, maliyetli yeniden tasarımların önüne geçer ve nihai ürünün kullanıcı beklentileriyle uyumlu olmasını sağlar. Sürekli kullanılabilirlik testleri ve yinelemeli tasarım, kullanıcı deneyimini iyileştirmek ve değişen kullanıcı ihtiyaçlarına uyum sağlamak için kritik öneme sahiptir [96].

3. MATERYAL VE METOT

Model mimarisi açısından, BDM'de kullanılanlar gibi dönüştürücü tabanlı bir model bu görev için avantajlı olmakla birlikte, bu modeller insan benzeri metinleri anlama ve üretme konusunda olağanüstü yetenekler göstermektedir. Bu yetenek de onları kod üretme görevleri için uygun hale getirmektedir [103]. Dönüştürücü mimarisinin dikkat mekanizmaları, modelin girdi isteminin ilgili kısımlarına odaklanmasını sağlamaktadır. Bu özellik, doğru ve bağlama uygun HTML kodu üretmek için gereklidir. Ayrıca, transfer öğrenmenin kullanılması, modelin önceden eğitilmiş modellerden gelen bilgilerden yararlanmasına izin vererek modelin performansını artırabilmektedir; bu da özellikle daha küçük veri setleriyle çalışırken fayda sağlamaktadır [103]. Sistemin operasyonel akışı, ZOA platformunun veri akışı gösterimini sağlayan Şekil *3.1*'de gösterilmektedir. Diyagram, kullanıcı girdisiyle başlayıp kullanıcı arayüzü (UI), uygulama programlama arayüzü (API) ve BDM üzerinden ilerleyerek oluşturulan HTML çıktısını ve canlı ön izlemeyi kullanıcıya döndürmeden önce platformun bileşenleri arasındaki etkileşimi geçerken girdi verilerinin akışını temsil ederken, yeşil oklar, oluşturulan HTML ve canlı ön izlemesi de dahil olmak üzere işlenmiş verilerin geri dönüşünü göstermektedir. Bu görselleştirme, sistemin sıralı ve çift yönlü yapısını vurgulamakta ve kullanıcı girdisinin platform içinde nasıl eyleme dönüştürülebilir çıktılara dönüştürüldüğünün net bir şekilde anlaşılmasını sağlamaktadır.



Şekil 3.1. ZOA platformu veri akışı diyagramı.

Veri seti seçimi, dil modelinin eğitim sonuçlarının kalitesi konusunda oldukça önem arz etmektedir. Literatürde [103], alana özgü veri setlerinin kullanılmasının, dil modellerinde

kod üretme performansının önemli ölçüde artırdığını göstermektedir. ZOA platformu için Tailwind CSS çerçevesi ile oluşturulan HTML ve CSS bileşenlerinden oluşan bir veri seti idealdir. Tailwind CSS, geliştiricilerin HTML kodlarında modern ve özel tasarımlar oluşturmaya olanak tanıyan bir CSS çerçevesidir. Bu veri seti, zengin bir bileşen çeşitliliği sağlayarak modelin çeşitli web tasarımları üretme yeteneğini artırmaktadır.

Eğitim süreci, GPT-4, Llama veya benzer mimariler gibi var olan bir büyük dil modellerinin, seçilen veri seti üzerinde ince ayarının yapılarak yeniden eğitilmesini içermektedir. Bu yaklaşım, kontrol edilebilirliği ve yorumlanabilirliği artırmak için büyük dil modellerini geleneksel programlama paradigmalarıyla birleştirmenin avantajlarını vurgulayan çalışmanın [103] bulgularıyla uyumludur. Model, özellikle HTML ve Tailwind CSS üzerinde eğitilerek, geçerli web kodu üretmek için gerekli sözdizimini ve yapıyı öğrenebilir, böylece çıktıdaki hata olasılığını azaltabilmektedir.

Bu bölümde, çalışmanın gerçekleştirilmesi için kullanılan veri setleri, makine öğrenmesi ve derin öğrenme yöntemleri ile ilgili detaylı açıklamalar sunulmaktadır. Bu kapsamda, ZOA platformunun geliştirilme sürecinde izlenen adımlar ve kullanılan araçlar detaylandırılmıştır.

3.1. EĞİTİM VE DEPOLAMA PLATFORMU

Bu çalışmada dil modellerine erişip kullanabilmek için Hugging Face (HF) platformu, eğitim işlemleri için de Google Colab tercih edilmektedir. Colab'ta eğitim için A100 GPU donanımı tercih edilerek, modeli HF'den çekmek ve yeniden eğitmek için de transformers kütüphanesi kullanılmaktadır. Colab için ücretli abonelik kapsamında sunulan A100 GPU donanımı ile eğitim işlemleri yürütülmüştür. Ayrıca epoch ve loss değerleri gibi eğitim verilerini tutmak için, Colab'ın önerisiyle, Wandb AI isimli bir platform kullanılmaktadır. Platforma ücretsiz üye olup elde edilen erişim anahtarı Colab üzerine ekleyerek çalışma yapılabilmektedir. Birkaç günlük ücretsiz deneme sonrası platform, bir abonelik istemekte fakat araştırma ve akademik çalışmalar kullanan kullanıcılar için, akademisyenler ve öğrencilere ücretsiz bir paket sunmaktadır. Bunun için üniversitenin sunduğu e-mail adresini hesaba kaydedip doğrulama yapmak gerekmektedir. Böylece eğitim sırasında anlık olarak çeşitli grafikler ile performans takip edilebilmekte ve tüm gerekli istatiski veriler buradan elde edilebilmektedir. Kaydedilen verileri dışa aktar seçeneği ile görsel olarak veya csv gibi uzantılar ile veri olarak elde etmek mümkündür.

Bu çalışma kapsamında, eğitim performans metrikleri Wandb (Weights & Biases) [104] platformu ile takip edilmektedir. Colab ile çalışabilir bir Python kütüphanesi bulunan Wandb, BDM için eğitim sırasındaki verileri tutarak raporlama, grafik, gösterge paneli gibi imkanlar sunmaktadır. Bu verileri csv, png gibi çeşitli uzantılarda dışa aktarmak mümkündür.

3.2. VERİ SETİ OLUŞTURMA

ZOA platformunun başarısını belirleyen en kritik unsurlardan biri, kullanılan veri setlerinin kalitesidir. Veri setleri, yapay zeka modellerinin öğrenme sürecinin temelini oluşturur ve modellerin görevlerini etkin bir şekilde yerine getirebilmesi için gerekli bilgiyi sağlar. Bu çalışmada, HTML ve CSS bileşenlerinden oluşan bir veri seti, ZOA platformunun dil modelini eğitmek amacıyla kullanılmıştır. Veri setinin hazırlanması ve işlenmesi aşamaları hem model performansı hem de kullanıcı deneyimi üzerinde doğrudan etkili olmuştur. ZOA platformu için kullanılan veri seti, açık kaynak kodlu projelerden ve özel olarak yazılmış HTML kodlarından elde edilmiştir. Bu veri setine, Hugging Face platformunda yayımlanan ve Tailwind CSS çerçevesi ile oluşturulan "*jacob9706/tailwind-dataset*" veri seti de eklenmiştir. Çizelge *3.1*'de de gösterildiği üzere, veri seti toplamda 2524 satırdan oluştu, bu satırların 641'i açık kaynak kodlu projelerden ve özel yazılmış kodlardan, geri kalan 1883'ü ise Hugging Face veri setinden alınmıştır.

Çizelge 3.1. Eğitilecek veri setindeki kaynak dağılımı (satır toplam).

Kaynak	**Veri Adeti**
jacob9706/tailwind-dataset	1883
Açık kaynak kod ve manuel oluşturma	641

Veri seti, kullanıcılardan alınan doğal dil girdilerini anlamlandırmak ve bunlara uygun HTML kodu üretmek amacıyla düzenlenmiştir. Bu kapsamda, veriler girdi (input) ve çıktı (output) olarak iki bilgiye sahip olacak şekilde düzenlendi. Girdi (input) alanı, kullanıcı tarafından yapılan metin tabanlı girdileri içermektedir. Bu girdiler kullanıcıların oluşturmak istedikleri web sayfalarının yapısı ve içerik gereksinimlerini belirtmektedir. Çıktı (output) alanı ise, girdilere karşılık gelen HTML ve CSS kodlarını içermektedir. Bu

kodlar, modern ve kullanıcı dostu bir arayüz tasarımı sağlayacak şekilde yapılandırılmıştır. Kodlar, Tailwind CSS çerçevesini temel alır ve estetik olarak modern web sayfaları oluşturmayı hedeflemektedir.

Eğitilecek dil modelinin İngilizce dilinde daha iyi performans göstermesi [108] ve hazır olarak kullanılan veri setinin İngilizce olmasından dolayı, veri setindeki girdi ve çıktılar da İngilizce dilinde hazırlanmıştır. Bu verilerden bazı örnekler Çizelge *3.2*'de gösterilmektedir.

Çizelge 3.2. Çalışmada kullanılan veri setinden örnekler.

Girdi	**Çıktı**
The design of the component is a simple, clean layout with a white background and black text. The text is organized into paragraphs and bullet points, making it easy to read and understand. The component includes a search bar at the top, allowing users to quickly find specific information within the content. The overall design is minimalistic and user-friendly, focusing on delivering the information in a clear and concise manner.	```html<section class="py-20"> <ul class="space-y-4"> <li class="rounded-lg bg-gray-50 px-6 py-8"> <button class="font-heading group flex w-full items-center justify-between text-left font-bold"> Fusce eget nunc et libero accumsan rutrum quis nec lectus? <svg class="ml-2 h-4 w-4 flex-shrink-0 text-indigo-500" xmlns="http://www.w3.org/2000/svg" fill="none" viewbox="0 0 24 24" stroke="currentColor"> <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M5 15l7-7 7 7"></path> </svg> </button> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam euismod orci sed tristique placerat. Fusce in ligula urna. Fusce eget nunc et libero accumsan rutrum quis nec lectus. Quisque luctus sem nibh, quis ornare neque consectetur varius. Maecenas rhoncus consectetur rutrum. <li class="rounded-lg bg-gray-50 px-6 py-8"> <button class="font-heading group flex w-full items-center justify-between text-left font-bold"> Donec sed leo sit amet ante ornare laoreet in quis est? <svg class="ml-2 h-4 w-4 flex-shrink-0 text-indigo-500" xmlns="http://www.w3.org/2000/svg" fill="none" viewbox="0 0 24 24" stroke="currentColor"> <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M19 9l-7 7-7-7"></path> </svg> </button> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus quis ultricies est. Duis nec hendrerit magna. Ut vel orci gravida, hendrerit enim non, gravida turpis. Praesent non lectus porttitor, scelerisque nulla nec, ornare neque. Integer massa libero, ornare ut leo nec, scelerisque rutrum elit. Morbi et massa id ipsum lobortis rhoncus. Aenean sit amet porta nulla, quis varius risus. Suspendisse ultrices ut lectus non laoreet. Etiam ornare laoreet tortor quis porttitor. Suspendisse tempus erat non dui volutpat eleifend. <li class="rounded-lg bg-gray-50 px-6 py-8"> <button class="font-heading group flex w-full items-center justify-between text-left font-bold"> Mauris vitae ex ut lectus cursus ornare? <svg class="ml-2 h-4 w-4 flex-shrink-0

Çizelge 3.2 (devamı). Çalışmada kullanılan veri setinden örnekler.

Girdi	**Çıktı**
	text-indigo-500" xmlns="http://www.w3.org/2000/svg" fill="none" viewbox="0 0 24 24" stroke="currentColor"> <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M19 9l-7 7-7-7"></path> </svg> </button> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus quis ultricies est. Duis nec hendrerit magna. Ut vel orci gravida, hendrerit enim non, gravida turpis. Praesent non lectus porttitor, scelerisque nulla nec, ornare neque. Integer massa libero, ornare ut leo nec, scelerisque rutrum elit. Morbi et massa id ipsum lobortis rhoncus. Aenean sit amet porta nulla, quis varius risus. Suspendisse ultrices ut lectus non laoreet. Etiam ornare laoreet tortor quis porttitor. Suspendisse tempus erat non dui volutpat eleifend. <li class="rounded-lg bg-gray-50 px-6 py-8"> <button class="group flex w-full items-center justify-between text-left"> Nam consequat, augue sed rutrum faucibus? <svg class="ml-2 h-4 w-4 flex-shrink-0 text-indigo-500" xmlns="http://www.w3.org/2000/svg" fill="none" viewbox="0 0 24 24" stroke="currentColor"> <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M19 9l-7 7-7-7"></path> </svg> </button> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam euismod orci sed tristique placerat. Fusce in ligula urna. Fusce eget nunc et libero accumsan rutrum quis nec lectus. Quisque luctus sem nibh, quis ornare neque consectetur varius. Maecenas rhoncus consectetur rutrum. <li class="rounded-lg bg-gray-50 px-6 py-8"> <button class="group flex w-full items-center justify-between text-left"> Cras at ante non ligula pharetra elementum? <svg class="ml-2 h-4 w-4 flex-shrink-0 text-indigo-500" xmlns="http://www.w3.org/2000/svg" fill="none" viewbox="0 0 24 24" stroke="currentColor"> <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M19 9l-7 7-7-7"></path> </svg> </button> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam euismod orci sed tristique placerat. Fusce in ligula urna. Fusce eget nunc et libero accumsan rutrum quis nec lectus. Quisque luctus sem nibh, quis ornare neque consectetur varius. Maecenas rhoncus consectetur rutrum. </section> ```

Çizelge 3.2 (devamı). Çalışmada kullanılan veri setinden örnekler.

Girdi	**Çıktı**
The design of the component is a combination of blue and purple colors, with a curved shape that gives it an abstract appearance. The background color is dark blue, providing a stark contrast to the vibrant colors of the design. The design itself appears to be made up of different shapes and lines, creating a sense of depth and complexity. Overall, the component has a modern and dynamic look to it.	```html <section class=”relative bg-gray-800 py-20 2xl:py-40 overflow-hidden”> <div class=”relative max-w-4xl px-4 lg:px-0 mx-auto py-10”> <div class=”max-w-xl mb-14 lg:mb-28”> <span class=”text-lg text-blue-400 font-semibold”>Our Works <h2 class=”mt-8 text-5xl text-white font-bold font-heading”>More than 20 years in the game</h2> <div class=”hidden lg:block absolute top-0 left-0 -ml-80”> <img class=”-ml-24 w-112 h-96 object-cover” src=”https://images.unsplash.com/photo-1618005198919-d3d4b5a92ead?ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&ixlib=rb-1.2.1&auto=format&fit=crop&w=967&q=80” alt=””> <div class=”hidden lg:block absolute top-0 right-0 -mr-80”> <img class=”-mr-52 w-112 h-96 object-cover” src=”https://images.unsplash.com/photo-1618005198919-d3d4b5a92ead?ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&ixlib=rb-1.2.1&auto=format&fit=crop&w=967&q=80” alt=””> <div class=”relative max-w-4xl mx-auto”> <img class=”w-full h-96 mb-12 object-cover” src=”https://images.unsplash.com/photo-1617791160505-6f00504e3519?ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&ixlib=rb-1.2.1&auto=format&fit=crop&w=1100&q=80” alt=””> <div class=”flex flex-wrap -mx-5”> <div class=”w-full lg:w-4/5 px-5 mb-6 lg:mb-0”> <div class=”relative py-12 px-10 lg:px-20 bg-gray-600 rounded-xl”> <div class=”absolute top-0 -mt-6 left-0 ml-16 w-14 h-14 bg-gray-600” style=”clip-path: polygon(50% 0%, 100% 50%, 50% 100%, 0% 50%);”> <h3 class=”mb-6 text-4xl text-white font-bold font-heading”>Experience design for your products</h3> <p class=”text-lg text-gray-300”>The brown fox jumps over the lazy dog.</p> <div class=”w-auto mx-auto lg:w-1/5 px-5”> <button class=”inline-flex mr-2 items-center justify-center w-14 h-14 bg-blue-500 hover:bg-blue-600 rounded-full”> <svg class=”w-4 h-4” width=”7” height=”13” viewbox=”0 0 7 13” fill=”none” xmlns=”http://www.w3.org/2000/svg”> <path d=”M6.84708 12.1077C7.05097 12.3133 7.05097 12.6436 6.84708 12.8476C6.64319 13.0517 6.31377 13.0525 6.10988 12.8476L0.152917 6.8708C-0.0509739 6.66673 -0.0509738 6.33645 0.152917 6.13087L6.10988 0.154027C6.31377 -0.0500387 6.64319 -0.0500386 6.84708 0.154027C7.05098 0.358848 7.05098 0.689887 6.84708 0.893952L1.4143 6.50121L6.84708 12.1077Z” fill=”white”></path> </svg> </button> <button class=”inline-flex items-center justify-center w-14 h-14 bg-blue-500 hover:bg-blue-600 rounded-full”> <svg class=”w-4 h-4” width=”7” height=”13” viewbox=”0 0 7 13” fill=”none” xmlns=”http://www.w3.org/2000/svg”> <path d=”M0.152917 0.894235C-0.0509742 0.688658 -0.0509742 0.358375 0.152917 0.15431C0.356808 -0.0497557 0.686228 -0.0505119 0.89012 0.15431L6.84708 6.13116C7.05097 6.33522 7.05097 6.6655 6.84708 6.87108L0.890119 12.8479C0.686227 13.052 0.356807 13.052 0.152916 12.8479C-0.0509753 12.6431 -0.0509753 12.3121 0.152916 12.108L5.5857 6.50074L0.152917 0.894235Z” fill=”white”></path>

Çizelge 3.2 (devamı). Çalışmada kullanılan veri setinden örnekler.

Girdi	**Çıktı**
	</svg> </button> </section> ```

Veri setindeki bileşenler platform tarafında html kodları içerisine yerleştirilerek kullanıcıya sunulmaktadır. Bu veri seti, ZOA platformunun dil modelini eğitmek için kullanılan temel materyalidir ve modelin doğruluğu üzerinde önemli rol oynamaktadır. Veri setinin genişliği ve çeşitliliği, modelin farklı kullanıcı girdilerini anlaması ve doğru HTML kodları üretmesi açısından büyük bir avantaj sağlamaktadır. Özellikle, Tailwind CSS çerçevesi ile hazırlanan kodlar, modern web standartlarına uygun, estetik ve işlevsel sayfalar oluşturulmasına olanak tanımıştır. ZOA platformunun başarısında veri setlerinin doğru seçimi ve etkin işlenmesi kritik bir öneme sahiptir. Bu veri seti, platformun kullanıcı dostu ve erişilebilir bir web geliştirme aracı olmasını mümkün kılmıştır. Gelecekte, daha büyük ve çeşitli veri setlerinin entegrasyonu, model performansını ve kullanıcı deneyimini daha da iyileştirme potansiyeline sahiptir.

3.3. EĞİTİM TEKNİKLERİ VE KULLANIMI

ZOA platformunun dil modeli eğitiminde ve kullanıcı girdilerini anlayarak doğru HTML kodları üretmesinde makine öğrenmesi ve derin öğrenme teknikleri kritik bir rol oynamaktadır. Bu teknikler, verilerin işlenmesinden modelin optimize edilmesine kadar geniş bir yelpazede uygulanmış ve platformun başarısına önemli ölçüde katkıda bulunmuştur. Makine öğrenmesi, veri setinde bulunan giriş – çıkış eşleştirmelerini kullanarak modelin kullanıcı girdilerini daha iyi anlamasına olanak tanımış ve doğal dil işleme teknikleri ile bu girdiler analiz edilmiştir. Doğal dil işleme süreçleri, dil bilgisi analizi, lemmatizasyon ve sözcük vektörlerinin çıkarımı gibi yöntemlerle modelin performansını artırmıştır.

Derin öğrenme, makine öğrenmesinin bir alt dalı olarak, ZOA platformunun teknik altyapısında öne çıkan bir başka önemli bileşendir. Derin öğrenme modelleri, büyük veri setlerinden öğrenim sağlarken karmaşık yapıları çözümlemek ve doğru sonuçlar üretmek için güçlü algoritmalar sunar. Bu bağlamda, konvolüsyonel sinir ağları (CNN) gibi teknikler, HTML ve CSS bileşenlerinin yapısal özelliklerini öğrenmek ve sınıflandırmak

için kullanılmıştır. Ayrıca, büyük dil modelleri, doğal dil girdilerinden anlam çıkarma sürecini optimize etmiş ve kullanıcı ihtiyaçlarına uygun kodlar üretmiştir. Hugging Face platformunda mevcut dil modelleri, ZOA platformunun özel veri setine göre ince ayar yapılarak yeniden eğitilmiş ve böylece hem doğruluk hem de verimlilik artırılmıştır.

Makine öğrenmesi ve derin öğrenme süreçleri, veri ön işleme aşamasından başlayarak model seçimi ve performans değerlendirmesine kadar birbirini tamamlayan bir dizi adım içermektedir. Veri setleri, kullanıcı girdilerinin çeşitliliğini temsil edecek şekilde temizlenmiş, normalleştirilmiş ve tutarlı bir formata dönüştürülmüştür. Model eğitimi sırasında, belirlenen dil modelleri ZOA platformu için optimize edilmiş veri setleri üzerinde eğitilmiş ve eğitim sürecinde doğruluk, kayıp oranı ve işlem süreleri gibi performans metrikleri sürekli izlenmiştir. Bu süreçte, hem modelin kullanıcı girdilerini anlamadaki başarısını hem de kod üretim hızını artırmak amacıyla transfer öğrenimi gibi modern tekniklerden faydalanılmıştır.

ZOA platformunda uygulanan makine öğrenmesi ve derin öğrenme teknikleri, kullanıcı deneyimini iyileştirmek için kritik bir altyapı sağlamıştır. Bu teknikler sayesinde, kullanıcılar tarafından doğal dilde ifade edilen girdiler doğru bir şekilde analiz edilmiş ve hızlı bir şekilde işlevsel HTML kodlarına dönüştürülmüştür. Gelecekte, daha karmaşık dil modellerinin ve derin öğrenme algoritmalarının entegrasyonu ile platformun yeteneklerinin daha da artırılması hedeflenmektedir. Bu süreç, ZOA'nın hem daha geniş bir kullanıcı kitlesine hitap etmesine hem de web geliştirme süreçlerini daha demokratik hale getirmesine katkı sağlayacaktır.

3.3.1. Dil Modeli ve Eğitim Yöntemi Seçimi

ZOA platformunun temelini oluşturan dil modeli, HF üzerinde yer alan ve Transformer tabanlı mimarilere dayanan bir model olarak seçilmiştir. Bu modelin seçimi sırasında, modelin özellikle HTML ve CSS gibi yapılandırılmış dilleri üretme yeteneği göz önünde bulundurulmuştur. Platform, kullanıcı girdilerine göre Tailwind CSS çerçevesi destekli web sayfaları oluşturmayı hedeflediğinden, seçilen dil modelinin bu tür çıktılar üretme kapasitesine sahip olması kritik bir öneme sahiptir.

Modelin eğitimi, ince ayar yöntemi kullanılarak gerçekleştirilmiştir. İnce ayar, önceden eğitilmiş bir dil modelinin belirli bir görev için optimize edilmesi sürecini ifade eder. Bu süreçte, modelin performansını artırmak amacıyla özel olarak hazırlanmış bir veri seti kullanılmıştır. Veri seti, HTML, CSS ve JavaScript gibi web geliştirme dillerine ait kod

parçacıklarını içermektedir. Veri seti hazırlanırken, çeşitlilik ve gerçek dünya senaryolarını yansıtma gibi faktörler dikkate alınmıştır. Bu sayede, modelin kullanıcı girdilerine uygun ve işlevsel web sayfaları üretebilme yeteneği geliştirilmiştir. Eğitim sürecinde kullanılan veri setinin işlenmesi ve modelin optimize edilmesi için Adam optimizer ve öğrenme hızı gibi hiperparametreler dikkatle seçilmiştir. Modelin performansı, eğitim sırasında doğruluk oranı ve kayıp oranı gibi metriklerle sürekli olarak izlenmiştir. Modelin aşırı öğrenme gibi sorunlarla karşılaşmaması için erken durdurma (early stopping) tekniği uygulanmış ve eğitim sürecinde düzenli olarak validasyon testleri gerçekleştirilmiştir. Bu süreç, literatürde dil modellerinin görev odaklı performansını artırmak için önerilen yöntemlere dayanmaktadır. Örneğin, ince ayar yöntemi, dil modellerinin farklı görevler için optimize edilmesinde sıklıkla kullanılan ve etkili sonuçlar veren bir yaklaşım olarak kabul edilmektedir (Howard & Ruder, 2018) [49]. Ayrıca, hiperparametre optimizasyonu ve erken durdurma teknikleri, modelin genelleme kapasitesini artırmada önemli bir rol oynamaktadır (Goodfellow, Bengio, & Courville, 2016) [51]. Bu yöntemlerin entegrasyonu, ZOA platformunun kullanıcı girdilerine daha hassas ve etkili yanıtlar verebilmesini sağlamıştır.

Önceki çalışmalarda [3], [105] model seçimiyle ilgili detaylıca araştırılmış ve çeşitli dil modelleri üzerinde bazı testler uygulandı. Bu kapsamda, kodun başlangıç ve bitiş etiketlerinin geçerli olup olmadığı, tüm etiketlerin doğru şekilde açılıp kapandığı, kodun yeterli tasarım öğeleri içerip içermediği gibi sorulara odaklanılmıştır. Bu yaklaşımla, seçilen modelin temel gereksinimleri karşıladığı ve web geliştirme için uygun olduğu belirlenmiştir. Bu yöntemin uygulanabilirliği akademik bir çerçeveye dayanmaktadır. Bu sorulara dayalı bir değerlendirme yaklaşımı, Göreve Özgü Değerlendirme Çerçeveleri (Task-Specific Evaluation Frameworks) kapsamında da ele alınabilir. Örneğin, Jurafsky ve Martin'in (2020) "Konuşma ve Dil İşleme" (Speech and Language Processing) adlı çalışmasında [9], dil modellerinin belirli görevler için optimize edilmesi sürecinde hedefe uygunluk ve çıktının kalitesinin değerlendirilmesi için kriter bazlı soru setlerinin önemi vurgulanmıştır. Aynı şekilde, Wang ve Chen'in (2023) [106] BDM üzerine yaptıkları inceleme, görev odaklı çıktılar için spesifik metriklerin ve soruların performans analizi için nasıl temel oluşturabileceğini detaylandırmaktadır. Bu tür bir yaklaşım, model seçim sürecinin sistematik ve uygulama odaklı bir şekilde yürütülmesini sağlar. ZOA platformu gibi kullanıcı girdilerine dayalı çıktılar üreten bir sistemde, bu tür sorular üzerinden yapılan değerlendirmeler, modelin sadece teorik olarak değil, pratikte de hedeflenen

işlevleri yerine getirme kapasitesini ölçmeye olanak tanımıştır.

Bu kapsamda, model seçimi için HF'de bulunan çeşitli modeller üzerinde bu testler uygulandı. Donanım sınırlamaları nedeniyle, kişisel bir bilgisayar yüksek parametreli dil modellerini eğitmek için gerekli ortamı sağlayamadı. Daha küçük dil modelleriyle yapılan ilk eğitim denemeleri yetersiz sonuçlar verdiğinden dolayı HF Space'te ZeroGPU [107] adlı sanal bir sunucu üzerinde çalışma denendi. ZeroGPU, Transformers kütüphanesini desteklemek için tasarlanmıştır ve iş yükü başına 40 GB VRAM'e sahip Nvidia A100 donanımına sahiptir; bu da onu belirli parametrelerle büyük dil modellerini çalıştırmak için uygun kılmaktadır. Ancak, ZeroGPU sunucuları öncelikle eğitim işlemlerinden ziyade ön eğitimli modelleri çalıştırmak için tasarlandığından, HF tarafından getirilen sınırlamalar nedeniyle eğitim denemeleri başarısız oldu. Sonuç olarak, çalışma, ücretli abonelik planı aracılığıyla yüksek GPU destekli donanımla model eğitimi için daha güvenilir bir ortam sunan Google Colab'a ile yapıldı. Bu çalışmada eğitim için Colab'ın A100 GPU donanımı kullanılmıştır.

Çizelge 3.3. Dil modellerinin veri seti ile yeniden eğitim başarı sonuçları.

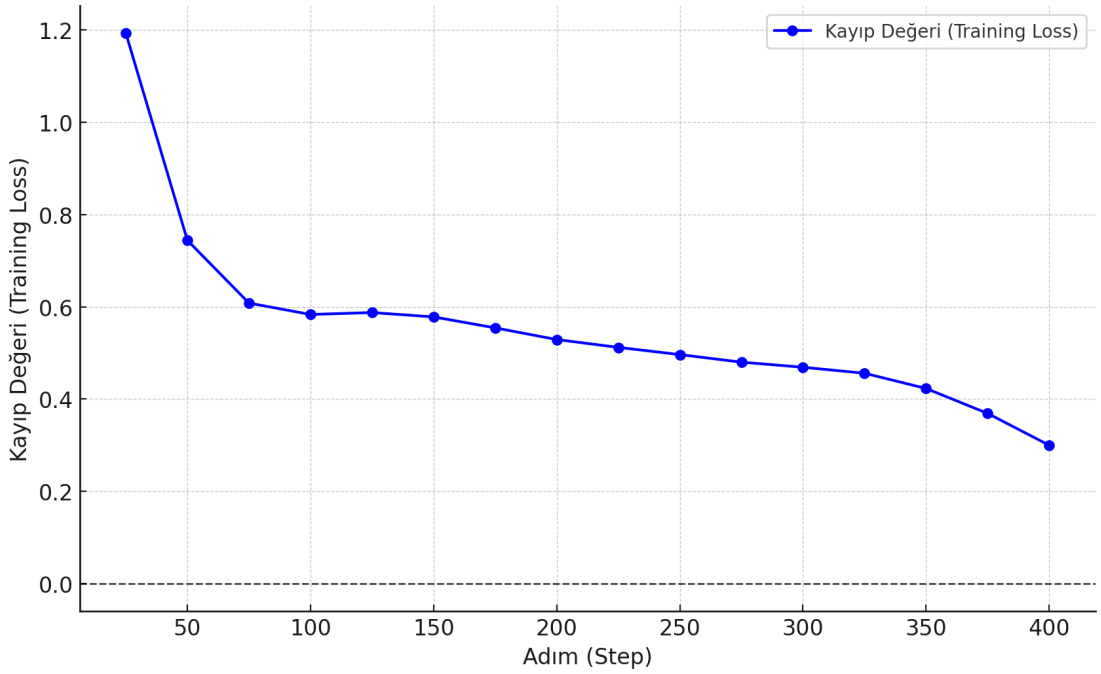
Model Adı	**Eğitim Süresi (saat)**	**Test Başarı Sonucu**
Salesforce/codet5-base	5	1/5
NousResearch/Llama-2-7b-chat-hf	8	4/5
HuggingFaceTB/SmolLM2-1.7B-Instruct	8	2/5
Qwen/Qwen2.5-1.5B-Instruct	7	2/5
google-t5/t5-base	4	1/5

Test başarı sonucu için yapılan değerlendirme kriterleri:

1. Geçerli HTML başlangıç ve bitiş etiketleri varlığı
2. Tailwind CSS’in doğru kullanımı (referans eklemeleri)
3. İstemle uyumluluk
4. Yeterli tasarım öğeleri kullanımı
5. Hız

Kod oluşturmada yetenekli en uygun modeli belirlemek üzere HF üzerindeki birkaç popüler BDM analiz edilmiştir. Seçilen modelin Transformers kütüphanesiyle uyumlu olması ve 83,5 GB sistem RAM'i, 40 GB GPU RAM'i ve 235,7 GB disk alanına sahip Colab'ın A100 GPU donanımında verimli bir şekilde çalışabilmesi gerekiyordu. Birçok BDM bu kriterlere uygun olsa da performansları önemli ölçüde farklılık gösteriyordu. En uygun modeli belirlemek için birden fazla modele ince ayar yapıldı ve 2524 satır içeren aynı veri kümesi üzerinde test edildi. Eğitim süresi, başarı oranları ve HTML etiketlerinin doğruluğu gibi temel ölçütler Çizelge *3.3* 'de özetlendiği gibi değerlendirilmiştir. Başarı oranları, geçerli HTML başlangıç ve bitiş etiketlerinin varlığı, Tailwind CSS'nin doğru kullanımı, istemlerle uygunluk ve yeterli tasarım öğelerinin dahil edilmesi gibi kriterlere göre hesaplanmıştır.

Çalışma kapsamında seçilen "*NousResearch/Llama-2-7b-chat-hf*" isimli model, Meta'nın açık kaynaklı Llama2 modelinin [47] 7 milyar ila 70 milyar parametre arasında değişen ince ayarlı bir versiyonudur [108]. Çok dilli desteği, yüksek performansı, açık kaynak yapısı ve Transformers kütüphanesiyle uyumluluğu seçilmesindeki temel faktörlerdi. Bu modelin seçilmesinin ardından, veri kümesi bu model kullanılarak hassas bir şekilde ayarlanmıştır. Eğitim sürecinde, ayarlanması gereken parametrelerin sayısını azaltarak büyük dil modellerinin verimli bir şekilde ince ayarlanması için tasarlanmış olan QLoRA [109] tekniği kullanılmıştır. Öğrenme oranı 2e-4 ve epoch değeri 1 olarak ayarlanmıştır. İşlem sırasında oluşturulan bir eğitim kaybı grafiği Şekil *3.2*'de gösterilmekte ve her adımdaki kayıp oranını göstermektedir.



Şekil 3.2. Yeniden eğitilen model için kayıp oranı grafiği.

Yeniden eğilip oluşturulan yeni model, "*osmankoc/llama-2-7b-zoa*" [110] adıyla HF platformuna yüklendi; Python ile geliştirilen ZOA API ile entegre edildi. API içerisinde yer alan metot ile, dışarıdan gelen girdi (prompt) isteğine karşılık olarak HTML web sayfası kodunu sonuç (response) olarak verecek şekilde bir akış kurgulandı. API, bu çalışma için geliştirilen arayüzde kullanılmaktadır.

3.4. VERİ İŞLEME SÜREÇLERİ

ZOA platformunun başarısını belirleyen temel unsurlardan biri, veri işleme süreçlerinin etkinliğidir. Veri işleme süreçleri, modelin eğitimi ve kullanıcı girdilerinin doğru bir şekilde anlaşılması için gerekli olan temel adımları içermektedir. Bu süreçler, veri temizleme, veri normalizasyonu, etiketleme ve veri dönüşümü gibi aşamalardan oluşur ve her bir adım modelin performansı üzerinde doğrudan bir etkiye sahiptir. Veri işleme sürecinin ilk adımı olan veri temizleme, veri setlerinde yer alan hatalı veya eksik bilgilerin ayıklanmasıyla başlar. Bu işlem, veri setinin tutarlılığını ve güvenilirliğini artırarak modelin daha doğru sonuçlar üretmesini sağlamaktadır.

Veri temizleme adımının ardından gelen veri normalizasyonu, veri setindeki farklı formatların ve ölçü birimlerinin standart bir forma dönüştürülmesini amaçlar. Bu süreç, özellikle HTML ve CSS kodlarının yapılandırılmasında büyük önem taşır. ZOA

platformunda kullanılan veri setleri, farklı kaynaklardan elde edildiği için normalizasyon işlemi, veri tutarlılığı sağlama açısından kritik bir rol oynamıştır. Örneğin, HTML ve CSS kodlarının tek bir biçime dönüştürülmesi, modelin girdileri daha kolay işlemesine olanak tanımış ve eğitimin verimliliğini artırmıştır. Veri setlerinde yer alan girdilerin ve çıktının anlamlı bir şekilde eşleştirilmesi için etiketleme işlemi gerçekleştirilmiştir. Bu aşama, dil modeli eğitiminde kullanılacak veri çiftlerinin oluşturulmasını sağlamış ve modelin, kullanıcı girdilerine uygun cevaplar üretmesine olanak tanımıştır.

Veri dönüşümü, veri setinin model eğitimine uygun bir formata getirilmesi için yapılan işlemleri içerir. Bu süreçte, kullanıcı girdileri doğal dil girdilerinden makine tarafından işlenebilir bir forma dönüştürülmüş, HTML kodları ise daha optimize bir şekilde depolanmıştır. Ayrıca, veri seti işleme sırasında kullanılan veri artırma teknikleri, modelin çeşitliliği anlamasında ve genelleştirme yeteneğini geliştirmesinde etkili olmuştur. Örneğin, veri artırma teknikleriyle veri setine yeni kombinasyonlar eklenmiş ve bu sayede model, farklı kullanıcı girdilerine daha etkili yanıtlar verebilmiştir.

Veri işleme süreçleri, sadece modelin doğruluğunu artırmakla kalmamış, aynı zamanda eğitim sürecinin hızlı ve verimli bir şekilde tamamlanmasına da olanak tanımıştır. ZOA platformunda kullanılan bu süreçler, kullanıcı girdilerinin doğru bir şekilde anlamlandırılmasını sağlamış ve üretilen kodların modern web standartlarına uygun olmasını garanti etmiştir. Gelecekte, daha büyük ve çeşitli veri setlerinin işlenmesiyle bu süreçlerin daha da iyileştirilmesi hedeflenmekte ve böylece ZOA platformunun yeteneklerinin daha da genişletilmesi amaçlanmaktadır. Böylece hem modelin performansını artıracak hem de kullanıcı deneyimini daha da geliştirecektir.

3.5. PERFORMANS VE DEĞERLENDİRME

ZOA platformunun performansı ve değerlendirme süreçleri, modelin doğruluk oranını, hızını ve kullanıcı deneyimini optimize etmek için kritik bir öneme sahiptir. Performans değerlendirmesi, modelin eğitim süreci sırasında ve sonrasında çeşitli metriklerin incelenmesiyle gerçekleştirilmiştir. Bu metrikler arasında doğruluk, kayıp oranı, işlem süresi ve modelin genelleştirme kapasitesi yer almaktadır [111], [112]. Modelin kullanıcı girdilerine verdiği yanıtların doğruluğu, eğitim verileriyle karşılaştırılarak test edilmiştir. Bu süreçte kullanılan metrikler, modelin farklı senaryolarda ne kadar başarılı olduğunu ölçmek ve eksikliklerini belirlemek amacıyla tasarlanmıştır.

Performans değerlendirme sürecinde doğruluk oranı, modelin ürettiği HTML kodlarının kullanıcı girdilerine uygunluğunu ölçmek için temel bir kriter olarak kullanılmıştır. Bu konuda geliştirilen modelin doğruluk oranı ve kayıp oranı değerlendirmesi, Flach'ın (2019) önerdiği yaklaşıma dayanmaktadır [113]. Bu doğrultuda, çalışma kapsamında geliştirilen ZOA dil modelinin ürettiği HTML kodlarının kullanıcı girdilerine uygunluğu, eğitim ve test verileri üzerinde kapsamlı analizlerle değerlendirilmiştir. Eğitim ve test verileri üzerinde yapılan analizlerde, doğruluk oranının yüksek olduğu, ancak bazı nadir durumlarda modelin kullanıcı girdilerini yanlış anlamlandırdığı gözlemlenmiştir. Bu durum, modelin eğitimi sırasında kullanılan veri setlerinin genişletilmesi ve daha fazla çeşitlilik içermesi gerektiğini ortaya koymuştur. Buna ek olarak, kayıp oranı, modelin öğrenme sürecindeki hatalarını ve bu hataların nasıl azaldığını anlamak için incelenmiştir. Eğitim süreci boyunca kayıp oranının kademeli olarak azaldığı, bu durumun modelin zaman içinde daha iyi bir performans sergilemesini sağladığı görülmüştür.

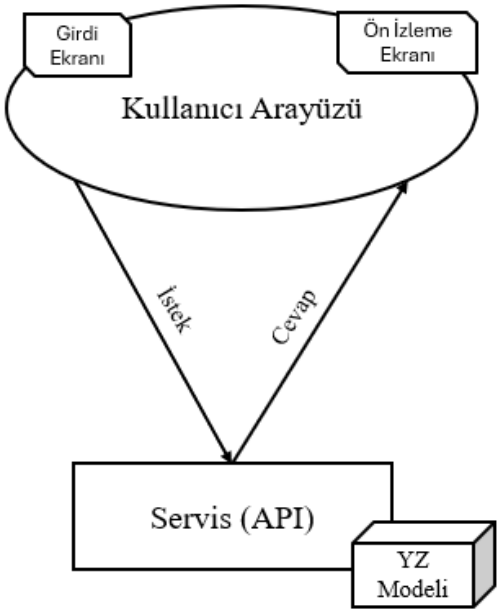
Geliştirilen sistemin işlem hızı ve kullanıcı deneyimi değerlendirmesi, Liu ve arkadaşlarının (2019) önerdiği performans değerlendirme çerçevesine dayanmaktadır [114]. Bu çalışma, makine öğrenimi modellerinin gerçek dünya uygulamalarındaki performansını ölçmek için işlem süresi ve kullanıcı memnuniyeti gibi metriklerin kullanılmasını önermektedir. Platformun performansı, sadece doğruluk oranıyla değil, aynı zamanda işlem hızıyla da değerlendirilmiştir. Bu yaklaşım, Nebius'un (2024) önerdiği YZ model performans metriklerine dayanmaktadır [115]. Kullanıcı girdilerinin analiz edilmesi ve uygun HTML kodlarının üretilmesi sürecinde düşük gecikme süresi, kullanıcı deneyimini doğrudan etkileyen bir faktör olmuştur. ZOA platformu, optimize edilmiş dil modelleri ve verimli bir API altyapısı sayesinde kullanıcı girdilerine hızlı yanıt verebilmiştir. Bu süreçte, API yanıt süreleri sürekli olarak izlenmiş ve platformun yüksek bir performans standardı yakaladığı görülmüştür. ZOA platformunun sürekli iyileştirilmesi için kullanıcı geri bildirimlerinin değerlendirilmesi, Fiddler ve arkadaşlarının (2018) çalışmasında önerilen yöntemlere dayanmaktadır [116]. Bu yaklaşım, model performansının gerçek kullanıcı deneyimleriyle sürekli olarak karşılaştırılmasını ve iyileştirilmesini önermektedir. Kullanıcı memnuniyetini artırmak için gerçekleştirilen anketler ve geri bildirimler, platformun performans değerlendirme sürecine önemli bir katkı sağlamıştır. Kullanıcı geri bildirimleri, modelin eksikliklerini anlamak ve bu eksiklikleri gidermek için önemli bir kaynak olmuştur.

ZOA platformunun performansına ilişkin değerlendirmeler, sadece mevcut durumun

analiz edilmesiyle sınırlı kalmamış, aynı zamanda gelecekte yapılacak iyileştirmelere de rehberlik etmiştir. Gelecekte, model performansını artırmak ve daha geniş bir kullanıcı kitlesine hitap etmek amacıyla daha karmaşık metriklerin ve test senaryolarının entegre edilmesi planlanmaktadır. Bu yöntem, Harmon ve arkadaşlarının (1997) çalışmasında önerilen çok boyutlu performans değerlendirme yaklaşımına dayanmaktadır [117]. Bu yaklaşım, makine öğrenimi modellerinin performansını çeşitli açılardan değerlendirmeyi ve sürekli olarak iyileştirmeyi amaçlamaktadır. Bu süreç, ZOA platformunun hem teknik kapasitesini hem de kullanıcı memnuniyetini sürekli olarak artırmasına olanak tanıyacaktır. Performans değerlendirme yöntemlerinin literatürdeki en güncel yaklaşımlarla uyumlu olması, platformun gelişimini ve etkinliğini artırmada kritik bir rol oynamaktadır.

4. GELİŞTİRİLEN SİSTEM

YZ destekli kodsuz geliştirme olanağı sunan ZOA platformu, kodlama sürecini basitleştirmek için yeniden eğitilmiş olan YZ dil modelinden yararlanarak kodsuz web geliştirmede önemli bir ilerlemeyi temsil etmektedir. Platform, HTML ve CSS bileşenlerinden oluşan özel bir veri seti kümesi üzerinde bir dil modelini eğiterek doğru, verimli ve kullanıcı dostu kod üretimi sunmayı amaçlamaktadır. Kodsuz çözümlere olan talep artmaya devam ederken, ZOA platformu geleneksel programlama becerileri olmadan web uygulamaları oluşturmak isteyen kullanıcıların ihtiyaçlarını karşılamak için minimalist bir çözüm olarak ortaya konmaktadır. Platformun potansiyel uygulamaları, bireysel kullanıcıların ötesinde web geliştirme süreçlerini kolaylaştırmak isteyen kuruluşlara kadar uzanmaktadır. Platform, ekiplerin kapsamlı kodlama bilgisi olmadan web uygulamaları oluşturmasını sağlayarak geliştirme süresini ve maliyetlerini önemli ölçüde azaltabilmektedir. Bu, üretkenliği artırdığı ve kuruluşlarda dijital dönüşümü kolaylaştırdığı gösterilen kodsuz geliştirme platformlarının artan eğilimiyle uyumludur [57]. Ayrıca, modern web standartlarına uygun kod üretme yeteneği, ZOA platformunu hızla gelişen web geliştirme ortamında değerli bir araç olarak konumlandırmaktadır. Platform mimarisi Şekil *4.1*'te gösterilmektedir.

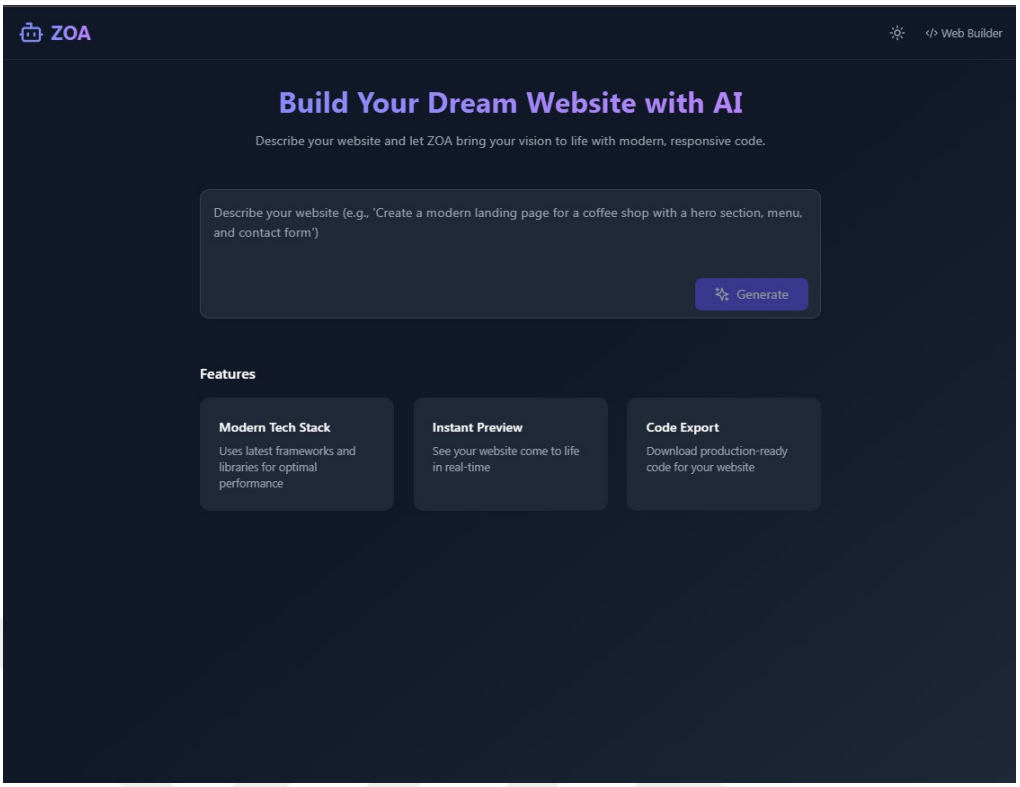


Şekil 4.1. Geliştirilen ZOA platformunu mimarisi.

ZOA platformunun uygulanması, kullanıcı arayüzü, model etkileşimi için servis (API) ve temel modelin kendisi dahil olmak üzere birkaç bileşen içermektedir. Kullanıcı arayüzü sezgisel ve duyarlı olmakla birlikte, kullanıcıların gereksinimlerini de kolayca girebilmelerini sağlamaktadır. Bu amaçla, modern ve kullanışlı bir arayüz geliştirildi. API ise, kullanıcı arayüzü ile eğitilmiş model arasında köprü görevi görerek kullanıcı istemlerinin HMTL koduna dönüştürülmesini kolaylaştırmaktadır. Şekil *4.1*'te gösterilen bu süreç, yalnızca kodun üretilmesini değil, aynı zamanda üretilen çıktı hakkında anında geri bildirim sağlayarak kullanıcı deneyimini geliştiren canlı bir ön izleme sağlamayı da içermektedir. Kullanıcılar istenen web sayfasını tanımlayan doğal dil istemlerini arayüz üzerinden girdiğinde geliştirilen arayüz ve API birbiri ile konuşarak, ilgili HTML ve CSS kodunu oluşturmaktadır. Sistemin işleyişini gösteren sözde (pseudo) kod aşağıdaki gibidir.

```
Başlat

prompt = KullanıcıdanGirdiAl()

Eğer prompt NULL ise
   HataMesajıGöster("Geçersiz bir değer girdiniz. Lütfen bir değer girin.")
   Bitir

ServisYanıtı = ServistenKodÜretimi (prompt)

Eğer APIYanıtı BAŞARILI ise
   KullanıcıArayüzündeGöster("Üretilen Kod", APIYanıtı.Kod)
   KullanıcıArayüzündeGöster("Önizleme", APIYanıtı.Önizleme)
Aksi halde
   HataMesajıGöster("Kod oluşturulurken bir hata oluştu. Lütfen tekrar deneyin.")

Bitir
```

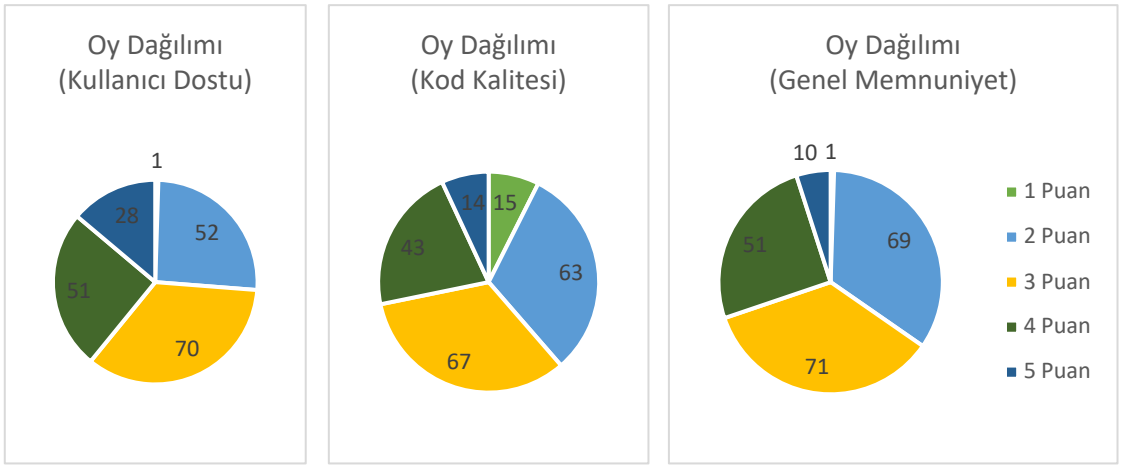
Platform arayüzü için modern ve kullanımı kolay olması hedeflenen bir tasarım geliştirildi. Arayüz geliştirilmesinde "Shadcn UI" isimli, Tailwind CSS çerçevesi ile geliştirilmiş, açık kaynak kodlu hazır bileşen seti içeren bir tasarım tercih edildi. React ile geliştirilen arayüzde, koyu ve açık tema desteği ile mobil uyumluluk (responsive) özellikleri bulunmaktadır. Kullanıcı girdi ve ön izleme ekranı olmak üzere iki ana ekrandan oluşmaktadır. Platform, koyu ve açık tema olmak üzere iki tema desteklemektedir. Koyu tema için olan ekran görüntüsü Şekil *4.2*'de gösterilmektedir.



Şekil 4.2. ZOA platformu kullanıcı arayüzü – Girdi ekranı (koyu tema).

4.1. PLATFORM PERFORMANS ANALİZİ

Geliştirilen platformun performansını değerlendirmek için çeşitli ölçütler kullanılmaktadır. Bunlar arasında kod doğruluğu, üretim hızı ve kullanıcı memnuniyeti yer almaktadır. Kod doğruluğu, oluşturulan HTML ve CSS kodlarının önceden tanımlanmış bir dizi standart ve en iyi uygulama ile karşılaştırılmasıyla değerlendirilecektir. Oluşturma hızı kullanıcı deneyimi için kritik öneme sahiptir; bu nedenle platform API yanıtlarında düşük gecikme sürecini hedeflemektedir. Kullanıcı memnuniyeti, kullanım kolaylığı ve üretilen çıktının kalitesine odaklanan anketler ve kullanılabilirlik testleri yoluyla ölçülmektedir. Buna ek olarak, kodun sürdürülebilirliği ve okunabilirliği analiz edilmektedir. Bu faktörler, web geliştirmede uzun vadeli kullanılabilirlik için gereklidir. Platformun temiz ve anlaşılabilir kod üretme kabiliyeti, kodsuz geliştirme ortamındaki başarısının önemli bir belirleyicisi olmaktadır.



Şekil 4.3. Kullanıcı oylarına göre derecelendirmelerin dağılımı.

Platform, minimum 3 yıl yazılım geliştirme tecrübesi bulunan 50 uzman yazılım geliştirici tarafından test edildi. Kullanıcılar toplamda 202 kod oluşturma talebi oluşturdu. Kullanıcıların deneyimlerine göre dağılımı ve istek sayıları Çizelge *4.1*'de gösterilmektedir.

Çizelge 4.1. Testi gerçekleştiren kullanıcı dağılımı.

Kullanıcı Sayısı	**Yazılım Geliştirme Tecrübesi (yıl)**	**İstek Sayısı**
22	3 – 5	88
20	5 – 10	84
8	10+	30

Test eden kullanıcılara platform kulanım kolaylığı (User Friendly), istek sonucunda oluşturulan kodun kalitesi (Code Quality) ve genel memnuniyet (Overall Rating) olmak üzere üç başlıkta 1 – 5 arası puanlama yapması istendi; 1 en kötü ve 5 en iyi olmak üzere değerlendirildi. Yapılan değerlendirmeler için verilen oy dağılımları Şekil *4.3*'de gösterilmektedir. Grafik değerleri Çizelge *4.2*'de gösterilmektedir.

Çizelge 4.2. Kullanıcı oylarına göre derecelendirmelerin dağılımı.

Değerlendirme	Kullanıcı Dostu (toplam oy sayısı)	Kod Kalitesi (toplam oy sayısı)	Genel Memnuniyet (toplam oy sayısı)
1 Puan	1	15	1
2 Puan	52	63	69
3 Puan	70	67	71
4 Puan	51	43	51
5 Puan	28	14	10

Çizelge 4.3. Kategoriye göre ortalama oylama değerleri.

	Kullanıcı Dostu (toplam oy sayısı)	Kod Kalitesi (toplam oy sayısı)	Genel Memnuniyet (toplam oy sayısı)
Ortalama Oy Değeri	3,26	2,89	3

Kullanıcı oylamalarında verilen puanlamaların ortalamaları Çizelge *4.3*'te gösterilmektedir. Değerlerin genel ortalaması ise 5 üzerinden 3,05 olarak hesaplanmakta ve buna göre platformun ortalama seviyede bir memnuniyet sağladığı görülmektedir. Bu da ilk kullanım için ideal bir derecelendirme olmakla birlikte, gelişmeye açık yönlerin de olduğunu da göstermektedir. Özellikle kod kalitesi ve üretilen kodun çıktısının daha iyi olması gerektiği açıktır. Daha büyük bir dil modeli kullanımı, daha büyük bir veri kümesi ile çalışma ve eğitim parametrelerinde iyileştirmeler yapma gibi farklı gelişim test senaryoları mevcuttur. Bu senaryolar ile, gelecek çalışmalarda daha yüksek bir memnuniyet ve kalite oranı hedeflenmektedir.

4.2. KULLANICI DENEYİMİ

ZOA platformunun kullanıcı deneyimi tasarımı, kullanıcıların ihtiyaçlarını hızlı, etkili ve sezgisel bir şekilde karşılamaya yönelik olarak geliştirilmiştir. Platform, hem teknik bilgiye sahip olmayan kullanıcılar hem de profesyonel geliştiriciler için kolay anlaşılabilir bir arayüz sunmaktadır. Bu yönüyle, yazılım geliştirme süreçlerini demokratikleştirerek daha geniş bir kullanıcı kitlesine hitap etmektedir. Kullanıcı arayüzü, modern tasarım

prensipleri göz önünde bulundurularak oluşturulmuştur. React ve Tailwind CSS kullanılarak geliştirilen bu arayüz, kullanıcıların girdilerini kolayca iletebileceği bir giriş ekranı ile üretilen çıktıları hızlıca görüntüleyebileceği bir ön izleme ekranından oluşmaktadır. Arayüz hem koyu hem de açık tema desteği sunarak kullanıcıların kişisel tercihlerine uygun bir deneyim sağlamaktadır. Ayrıca, platformun tamamen mobil uyumlu olması, kullanıcıların her yerden erişim sağlayabilmesine imkan tanımaktadır. Özellikle "Shadcn UI" ile desteklenen bileşenler, arayüzün işlevselliğini ve görsel kalitesini artırmıştır.

Platformun kullanıcı deneyimi üzerine yapılan testlerde, kullanıcıların girdilerine karşılık alınan yanıtların hızlı ve doğru olduğu görülmüştür. Beta testlerine katılan kullanıcılar, arayüzü sezgisel bulduklarını ve kullanım sürecinde herhangi bir zorluk yaşamadıklarını ifade etmişlerdir. Bununla birlikte, kullanıcı geri bildirimlerinde, belirli girdiler için daha fazla tasarım seçeneği sunulmasının deneyimi daha da zenginleştirebileceği belirtilmiştir. Kullanıcı deneyimi tasarımı, yalnızca teknik performansla sınırlı kalmamış, aynı zamanda kullanıcı memnuniyetini artıracak detaylara da odaklanmıştır. Örneğin, kullanıcıların girdilerine göre üretilen çıktıları anlık olarak görebilmeleri, etkileşim sürecini güçlendirmiştir. Ayrıca, çıktıların görsellik ve içerik açısından profesyonel standartlara uygun olması, platformun hem bireysel hem de kurumsal kullanıcılar için cazip bir seçenek haline gelmesini sağlamıştır. ZOA'nın kullanıcı deneyimi yaklaşımı, sürekli iyileştirme prensibiyle desteklenmektedir. Kullanıcıların geri bildirimleri, sistemin daha iyi hale getirilmesi için düzenli olarak analiz edilmekte ve bu geri bildirimlere dayalı güncellemeler yapılmaktadır. Bu sayede, ZOA platformu hem mevcut kullanıcıların memnuniyetini artırmakta hem de potansiyel kullanıcılar için cazip bir alternatif sunmaktadır. ZOA platformu, kullanıcı dostu tasarımı, hızlı ve etkili geri dönüşleri ve kişiselleştirme seçenekleriyle kullanıcı deneyimini en üst düzeye taşımayı başarmıştır. Platformun bu yönü, yalnızca yazılım geliştirme süreçlerini kolaylaştırmakla kalmayıp, aynı zamanda kullanıcıların yaratıcı projelerini hayata geçirmelerini de destekleyen güçlü bir araç olmasını sağlamaktadır.

4.3. KARŞILAŞTIRMALI ANALİZ

ZOA platformu, yapay zeka destekli kodsuz yazılım geliştirme alanında mevcut diğer platformlarla karşılaştırıldığında, belirgin avantajlar ve bazı geliştirme alanları sunmaktadır. Platformun özelliklerini daha iyi anlamak ve bu alanlardaki konumunu

değerlendirmek için, geleneksel kodsuz platformlar ve yapay zeka destekli alternatiflerle bir karşılaştırma yapılmıştır. ZOA, özellikle kullanıcı girdilerinden HTML ve CSS çıktıları üretme konusunda üstün bir performans göstermektedir. Kullanıcıların doğal dilde verdiği talimatları işleyerek kod üretmesi, bu platformu geleneksel kodsuz platformlara göre daha erişilebilir hale getirmektedir. Geleneksel platformlar genellikle sürükle-bırak yöntemine dayanırken, ZOA'nın DDİ teknolojisi ile desteklenmiş yapısı, kullanıcı deneyimini daha dinamik ve verimli hale getirmiştir. Ayrıca, Tailwind CSS çerçevesini kullanarak üretilen çıktılar, modern ve kullanıcı dostu tasarımlara olanak tanımaktadır.

Çizelge 4.4. ZOA, Mistral ve GPT modellerinin performans ölçüm değerleri.

Metrik	**Llama-2-7B-ZOA**	**Mistral-7B**	**GPT-4**
Çıkarım Hızı (sn)	3,2	2,8	4,1
Blue Puanı	0,78	0,65	0,82
Bellek Kullanımı (GB)	5,1	4,8	8,2

Bu çalışmada, "*osmankoc/llama-2-7b-zoa*" modelinin performansı, özellikle Tailwind CSS ile HTML kod üretimi bağlamında, çeşitli metrikler kullanılarak değerlendirilmiştir. Modelin performansını ölçmek için kullanıcı değerlendirmesi sonuçları ile ilgili bilgiler paylaşıldı. Bu değerlendirmeler, geliştiriciler tarafından üretilen kodun görsel doğruluğu, temizliği ve Tailwind CSS'in en iyi uygulamalarına uygunluğu açısından puanlanarak kaliteyi ölçer. Diğer metrikler de Blue puanı, çıkarım (inference) hızı ve bellek kullanımı gibi değerlerdir. Blue puanı, modelin ürettiği kodun referans çözümlerle olan benzerliğini ölçerek dil modelinin doğruluğunu niceliksel olarak ifade etmektedir. Çıkarım hızı, modelin bir girdiye ne kadar sürede yanıt üretebildiğini gösterirken, bellek kullanımı modelin donanım kaynaklarını ne kadar verimli kullandığını ortaya koymaktadır. Elde edilen sonuçlar Çizelge *4.4*'te gösterilmektedir. Bu sonuçlara göre geliştirilen ZOA modeli, özellikle Tailwind CSS ile kod üretimi gibi spesifik bir alanda uzmanlaşmış olması nedeniyle dikkat çekicidir. Model 3,2 saniyelik çıkarım (inference) hızı ve 5,1 GB bellek kullanımı ile kaynak verimliliği açısından öne çıkmaktadır. Ayrıca 0,78 Blue puanı ve kullanıcı değerlendirmeleri baz alındığında modelin hem doğruluk hem de kalite açısından kullanışlı olduğu görülmektedir. Bu performans, modelin özellikle gerçek

zamanlı uygulamalarda ve kaynak kısıtlı ortamlarda kullanımını cazip kılmaktadır. Açık kaynaklı olması ve tam kontrol imkanı sunması, modelin güvenilirliğini artırmakta ve özelleştirilebilir yapısıyla geliştiricilere esneklik sağlamaktadır. Bu nedenle, Tailwind CSS tabanlı web bileşenleri geliştiren ekipler için bu model hem maliyet hem de performans açısından etkili bir çözüm sunmaktadır. Bu metrikler, modelin hem teknik hem de pratik açıdan üstünlüklerini ortaya koyarken, aynı zamanda kullanıcıların ihtiyaçlarına uygun bir çözüm olarak öne çıkmasını sağlamaktadır. Bu çalışma, spesifik bir alanda uzmanlaşmış dil modellerinin, genel amaçlı modellere kıyasla nasıl daha verimli ve etkili olabileceğini göstermektedir.

ZOA, kullanıcıların karmaşık teknik bilgi gerektirmeden yazılım geliştirme süreçlerine dahil olmalarını sağlarken, bazı yapay zeka destekli platformlarla kıyaslandığında daha hafif bir yapı sunmaktadır. Örneğin, Amazon SageMaker gibi ileri düzey YZ platformları, veri bilimi ve makine öğrenimi modellerini eğitmek için daha geniş kapsamlı araçlar sunarken, ZOA özellikle web geliştirme süreçlerine odaklanarak belirli bir alanda derinleşmiştir. Bu odaklanma, platformun kullanıcı dostu olmasını sağlamış, aynı zamanda öğrenme eğrisini de azaltmıştır. Platformun performansı ve kullanıcı deneyimi, diğer YZ destekli kodsuz platformlarla karşılaştırıldığında dikkate değer bir noktaya sahiptir. Örneğin, Google Teachable Machine ve Lobe AI gibi platformlar görüntü sınıflandırma ve model eğitimi gibi alanlarda öne çıkarken, ZOA'nın güçlü olduğu nokta, web geliştirme süreçlerini demokratikleştirerek HTML ve CSS üretimine odaklanmasıdır. Bununla birlikte, kullanıcı geri bildirimlerine dayalı tasarım seçeneklerinin çeşitliliği gibi alanlarda, daha fazla geliştirme yapılması gerektiği ifade edilmiştir.

ZOA'nın karşılaştırmalı avantajlarından biri de düşük donanım gereksinimleri ile çalışabilmesidir. Örneğin, Microsoft'un Copilot gibi yapay zeka tabanlı kod üretim araçları daha büyük veri setleri ve yüksek hesaplama gücü gerektirirken, ZOA, daha geniş bir kullanıcı kitlesine hitap edecek şekilde optimize edilmiştir. Bu özellik, özellikle bireysel kullanıcılar ve küçük işletmeler için ekonomik ve pratik bir çözüm sunmaktadır. ZOA platformu, mevcut YZ destekli ve geleneksel kodsuz platformlara kıyasla kullanıcı dostu arayüzü, doğal dil girdileri ile çalışma yeteneği ve web geliştirme süreçlerine odaklanmasıyla dikkat çekmektedir. Bununla birlikte, platformun geniş veri setleri ile desteklenmesi, kullanıcı geri bildirimlerine dayalı iyileştirmelerin yapılması ve daha fazla entegrasyon seçeneği sunması, gelecekteki geliştirmeler için önemli fırsatlar yaratabilir. Bu özellikleriyle ZOA, kodsuz yazılım geliştirme alanında güçlü bir alternatif olarak

konumlanmaktadır. Ayrıca, ZOA’nın baz aldığı Llama modeli sayesinde Türkçe isteklere de cevap verebilmektedir. Bu konuda servis kodları içerisinde sistem isteri de eklenerek istenilen dilde web sitesi oluşturması sağlanmaktadır.

Sonuç olarak, geliştirilen platformun yüksek doğruluk oranları ve hızlı yanıt süreleri, uygulanan metodolojinin etkinliğini kanıtlamaktadır. Veri setlerinin özenle seçimi, model eğitimi sırasında kullanılan fine-tuning yöntemleri ve API'nin performans optimizasyonu, bu başarıyı destekleyen temel unsurlar olmuştur. Diğer yandan, kullanıcı deneyimi değerlendirmelerinde, ZOA’nın kullanıcı dostu arayüzü ve hızlı yanıt verme yeteneği öne çıkmıştır. Bu özellikler, literatürdeki birçok kodsuz platformun karşılaştığı kullanım zorluklarını aşmayı mümkün kılmıştır. Ayrıca, kullanıcıların platformu sezgisel bulması ve geri bildirimlerle olumlu görüşler sunması, platformun pratik uygulamalarda geniş bir kitleye hitap edebileceğini göstermektedir.

5. SONUÇ

Bu tez çalışması, YZ destekli kodsuz yazılım geliştirme platformlarının gelecekteki potansiyelini inceleyerek, yazılım mühendisliği ve kullanıcı deneyimi alanında yenilikçi bir yaklaşımı temsil etmektedir. Tez kapsamında geliştirilen ZOA platformu, DDİ tekniklerini ve büyük dil modellerini bir araya getirerek, yazılım geliştirme süreçlerini demokratikleştirmeyi ve daha geniş bir kullanıcı kitlesine erişim sağlamayı hedeflemiştir. Bu bağlamda elde edilen sonuçlar, yapay zekanın yazılım geliştirme süreçlerindeki dönüştürücü etkisini açıkça göstermekte ve bu alandaki mevcut bilgiye önemli katkılar sunmaktadır.

ZOA platformunun geliştirilmesi sürecinde, özellikle kullanıcıların doğal dil girdileriyle işlevsel web sayfaları oluşturmasına olanak tanıyan bir altyapı tasarlanmıştır. Bu tasarım, yazılım geliştirme süreçlerinde teknik bilgisi olmayan kullanıcıların bile aktif bir rol almasını sağlayarak yazılım süreçlerini daha erişilebilir hale getirmiştir. DDİ tekniklerinin kullanımı, kullanıcı girdilerinin daha iyi anlaşılmasını ve bu girdilere uygun HTML, CSS ve diğer bileşenlerin oluşturulmasını mümkün kılmıştır. Platformun tasarımı, yazılım geliştirme sürecinin maliyetlerini düşürmek ve hızını artırmak amacıyla yapılandırılmıştır. Bu durum hem bireysel kullanıcılar hem de kurumsal düzeyde yazılım geliştirme ihtiyaçları için önemli bir avantaj sağlamıştır.

Platformun performansı, çeşitli test senaryolarıyla değerlendirilmiş ve kullanıcı deneyimi açısından yüksek bir başarı elde edilmiştir. Geliştirme sürecinde kullanılan veri setlerinin çeşitliliği, ZOA'nın farklı kullanıcı ihtiyaçlarına yanıt verme kapasitesini artırmıştır. Bu veri setleri, modern web standartlarına uygun ve kullanıcı dostu web sayfaları oluşturulması için gerekli altyapıyı sağlamıştır. Veri setlerinin genişliği ve çeşitliliği, yalnızca platformun doğruluğunu artırmakla kalmamış, aynı zamanda kullanıcı deneyimini iyileştirmiştir. Özellikle, Tailwind CSS çerçevesine dayalı bir veri seti kullanılması, platformun çıktılarının estetik ve işlevsel olmasını sağlamış ve bu yönüyle kullanıcı memnuniyetine katkıda bulunmuştur. ZOA platformunun geliştirilmesi sırasında doğal dil girdileriyle oluşturulan web bileşenlerinin kalitesi, kullanıcıların platformu etkin bir şekilde kullanmasını kolaylaştırmış ve daha önce karşılaşılan teknik engelleri ortadan kaldırmıştır.

Tez çalışması, YZ destekli kodsuz platformların yazılım geliştirme süreçlerinde sağlayabileceği katkılara dair önemli bulgular sunmaktadır. Platformun kullanıcı dostu arayüzü, yazılım geliştirme süreçlerini demokratikleştirirken, kullanıcıların taleplerine hızlı ve doğru bir şekilde yanıt verme kapasitesine sahiptir. Özellikle kullanıcıların doğal dilde ifade ettikleri gereksinimlerin doğru bir şekilde analiz edilmesi ve bu gereksinimlere uygun yazılım çıktılarının üretilmesi, ZOA platformunun en güçlü yönlerinden biri olarak ortaya çıkmıştır. Bununla birlikte, platformun performansı değerlendirildiğinde, kullanıcı deneyimini iyileştirme ve hata oranlarını azaltma gibi alanlarda da geliştirme potansiyeli bulunmaktadır. Kullanıcı geri bildirimleri, platformun özellikle güvenlik, veri gizliliği ve kullanıcı arayüzü tasarımı gibi alanlarda iyileştirilmesi gerektiğini göstermektedir.

Geleneksel yazılım geliştirme süreçlerinde yüksek maliyetli ve zaman alıcı olan aşamalar, ZOA gibi YZ destekli kodsuz platformlarla önemli ölçüde hızlandırılmış ve maliyet etkin bir şekilde yönetilmiştir. Platform, kullanıcıların doğal dil girdilerini analiz ederek bu girdilere uygun dinamik web sayfaları oluşturmasını sağlamış ve bu yönüyle yazılım geliştirme süreçlerinin verimliliğini artırmıştır. Bu durum, yalnızca bireysel kullanıcılar için değil, aynı zamanda işletmeler ve organizasyonlar için de önemli bir avantaj sağlamaktadır. Kurumsal düzeyde uygulamalar için ZOA platformunun potansiyeli, daha geniş veri setleriyle entegre edildiğinde ve kullanıcı geri bildirimlerine göre optimize edildiğinde daha da artacaktır.

Literatür taraması sırasında incelenen kodsuz yazılım geliştirme platformlarının sağladığı avantajlar ve sınırlamalar detaylı bir şekilde ele alınmış, bu doğrultuda ZOA platformunun sunduğu yenilikçi yaklaşımlar geliştirilmiştir. Özellikle DDİ tekniklerinin platforma entegre edilmesi, mevcut platformların kullanım kolaylığına dair öne çıkan sorunları çözmek için etkili bir yöntem olarak öne çıkmaktadır. Tez çalışmasında elde edilen bir diğer önemli bulgu, yapay zekanın etik ve güvenlik konularındaki potansiyel zorluklarıdır. ZOA platformunun geliştirilmesi sırasında, kullanıcı verilerinin güvenliği ve gizliliği ön planda tutulmuş ve modern şifreleme teknikleri entegre edilmiştir. Bununla birlikte, platformun güvenlik altyapısının daha da güçlendirilmesi ve kullanıcı verilerinin güvenli bir şekilde işlenmesi için ek önlemler alınması gerekmektedir. Özellikle kullanıcı verilerinin güvenli bir şekilde saklanması ve işlenmesi, platformun uzun vadeli başarısı için kritik bir öneme sahiptir. Gelecekte, ZOA platformunun güvenlik özelliklerinin geliştirilmesi ve kullanıcıların veri gizliliği konusundaki endişelerini giderecek çözümler sunulması önerilmektedir.

ZOA platformunun geliştirilmesi sırasında elde edilen bulgular, YZ destekli kodsuz platformların yazılım geliştirme süreçlerinde sahip olduğu potansiyeli ortaya koymuştur. Platform, kullanıcıların doğal dil girdileriyle işlevsel ve estetik web sayfaları oluşturmasını sağlarken, yazılım geliştirme süreçlerini hızlandırmış ve daha erişilebilir hale getirmiştir. Bu çalışma, YZ destekli kodsuz platformların yazılım mühendisliği ve kullanıcı deneyimi alanındaki potansiyelini göstermesi açısından önemli bir referans niteliğindedir. Özellikle dil modellerinin yazılım geliştirme süreçlerinde nasıl etkin bir şekilde kullanılabileceğini göstermesi bakımından değerli bir katkı sunmaktadır.

Gelecekte, ZOA platformunun daha geniş veri setleriyle eğitilerek kullanıcı girdilerine daha hassas ve doğru yanıtlar vermesi sağlanabilir. Ayrıca, çok dilli destek gibi özelliklerin eklenmesi, platformun farklı kullanıcı grupları için daha erişilebilir hale gelmesini sağlayabilir. Kullanıcıların geri bildirimlerine dayalı sürekli iyileştirme süreçleri, platformun performansını artıracak ve kullanıcı deneyimini iyileştirecektir. Bununla birlikte, ZOA platformunun otomatik dağıtım özellikleri eklenerek, kullanıcıların oluşturdukları web sayfalarını doğrudan yayımlamalarına olanak tanıyan bir altyapı geliştirilebilir. Bu tür yenilikler, platformun kullanımını daha da kolaylaştıracak ve kullanıcı memnuniyetini artıracaktır. Uygulama https://zoa.osmankoc.dev/ adresinde yayına alınarak kullanıma sunulmuştur. Kaynak kodları da GitHub üzerinde https://github.com/osman-koc/zoa-web-builder adresinde paylaşılmıştır.

Sonuç olarak, bu çalışma, YZ destekli kodsuz platformların yazılım geliştirme süreçlerinde sahip olduğu dönüştürücü potansiyeli göstermektedir. ZOA platformu, yazılım geliştirme süreçlerini demokratikleştirerek, teknolojiye erişimi daha geniş bir kitle için mümkün kılmayı hedeflemiştir. Bu bağlamda, ZOA platformunun geliştirilmesi sırasında elde edilen bulgular hem akademik hem de pratik anlamda önemli katkılar sunmaktadır. Platformun gelecekteki geliştirme süreçleri, daha fazla yenilik ve entegrasyon fırsatı sunarak yazılım geliştirme alanında yeni açılımlar sağlayacaktır. Bu tez, ZOA platformunun potansiyelini ve yapay zeka destekli kodsuz yazılım geliştirme süreçlerinin gelecekteki rolünü anlamak için önemli bir temel oluşturmaktadır.

6. KAYNAKÇA

[1] TÜSİAD Yazılım Çalışma Grubu, "Türkiye'de Yazılım Ekosisteminin Geleceği," *Deloitte*, Ocak 2021.

[2] M. R. DeLisi ve C. Howley. (2022, 13 Aralık). *Gartner Forecasts Worldwide Low-Code Development Technologies Market to Grow 20% in 2023*. [Online]. Gartner. Erişim: https://www.gartner.com/en/newsroom/press-releases/2022-12-13-gartner-forecasts-worldwide-low-code-development-technologies-market-to-grow-20-percent-in-2023.

[3] O. Koç, İ. Yücedağ, ve Ü. Şentürk, "The Impact of Artificial Intelligence Enhanced No-Code Software Development Platforms on Software Processes: A Literature Review," *DÜBİTED*, c.13, sy. 1, ss 383-401, 2025.

[4] S. Russell ve P. Norvig, *Artificial Intelligence: A Modern Approach*, *Pearson Education (3. basım)*, Ocak 2013.

[5] N. J. Nilsson, "Artificial Intelligence: A New Synthesis," *Morgan Kaufmann Publishers*, 1998.

[6] A. M. Turing, "Computing Machinery and Intelligence," *Mind*, c. 59, sy. 236, ss. 433-460, Oxford University Press on behalf of the Mind Association, Ekim 1950.

[7] J. McCarthy, M. L. Minsky, N. Rochester, ve C. E. Shannon, "A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence," *AI Magazine*, c. 27, 2006.

[8] A. A. Abro, M. S. H. Talpur, ve A. K. Jumani, "Natural Language Processing Challenges and Issues: A Literature Review," *Gazi Üniversitesi*, Aralık 2023, doi: 10.35378/gujs.1032517.

[9] Jurafsky, D. ve Martin, J. H., *Speech and Language Processing: An Introduction to Natural Language Processing*, Computational Linguistics, and Speech Recognition, *Pearson (2. Basım)*, 2008.

[10] S. Liu, H. La, A. Willms, ve R. E. Rhodes, "A No-Code App Design Platform for Mobile Health Research: Development and Usability Study," *JMIR Form Res*, c. 6, sy 8, Ağustos 2022.

[11] L. Sundberg ve J. Holmström, "Using No-Code AI to Teach Machine Learning in Higher Education", *Journal of Information Systems Education*, c. 35, sy 1, ss. 56-66, Aralık 2024.

[12] Z. Gong, P. Zhong ve W. Hu, "Diversity in Machine Learning," *IEEE Access*, c. 7, ss. 64323-64350, 2019.

[13] L. Sundberg ve J. Holmström, "Democratizing artificial intelligence: How no-code AI can leverage machine learning operations," *Bus Horiz*, c. 66, sy 6, ss. 777-788, Kasım 2023.

[14] H. Sidhpurwala, G. Mollett, E. Fox, M. Bestavros ve H. Chen. (2024, Kasım). *Building Trust: Foundations of Security, Safety and Transparency in AI*. [Online]. Erişim: https://arxiv.org/abs/2411.12275.

[15] N. Sido, E. A. Emon, E. Ahmed, E. Supervisor ve M. Falch, "Low/No Code Development & Generative AI," Yüksek Lisans Tezi, Elektronik Sistemler Bölümü, Aalborg Üniversitesi, Kopenhag, Danimarka, Mayıs 2024.

[16] Z. Yan. (2021, Aralık). *The Impacts of Low/No-Code Development on Digital Transformation and Software Development*. [Online]. doi: 10.48550/arXiv.2112.14073.

[17] A. Hanandeh, R. Hanandeh, F. Raheem, F. R. Y. Alazzawi, A. Al-Daradkah, …, ve S. Rateb, "The effects of big data, artificial intelligence, and business intelligence on e-learning and business performance: Evidence from Jordanian telecommunication firms," *International Journal of Data and Network Science*, c. 7, sy. 1, ss. 35-40, 2023.

[18] T. N. Fitria, "Artificial intelligence (AI) technology in OpenAI ChatGPT application: A review of ChatGPT in writing English essay," *ELT Forum: Journal of English Language Teaching*, c. 12, sy. 1, ss. 44-58, 2023.

[19] S. Sok ve K. Heng, "Opportunities, challenges, and strategies for using ChatGPT in higher education: A literature review," *Journal of Digital Educational Technology*, c. 4, sy. 1, s. ep2401, Aralık 2023.

[20] E. Loos, J. Gröpler ve M. L. S. Goudeau, "Using ChatGPT in education: Human reflection on ChatGPT's self-reflection," *Societies*, c. 13, sy. 8, Ağustos 2023.

[21] S. S. Biswas, "Role of Chat GPT in Public Health," *Annals of Biomedical Engineering*, c. 51, sy. 5, ss. 868-869, Mayıs 2023.

[22] S. Ruksakulpiwat, A. Kumar ve A. Ajibade, "Using ChatGPT in medical research: Current status and future directions," *Journal of Multidisciplinary Healthcare*, c. 16, ss. 1513-1520, 30 Mayıs 2023.

[23] S. B. Johnson, A. J. King, E. L. Warner, S. Aneja, B. H. Kann ve C. L. Bylund, "Using ChatGPT to evaluate cancer myths and misconceptions: artificial intelligence and cancer information," *JNCI Cancer Spectrum*, c. 7, sy. 2, Mart 2023.

[24] M. F. Rabbi, A. I. Champa, M. F. Zibran ve M. R. Islam, "AI Writes, We Analyze: The ChatGPT Python Code Saga," *2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR 2024)*, ss. 177-181, 2024.

[25] M. L. Siddiq, L. Roney, J. Zhang ve J. C. S. Santos, "Quality assessment of ChatGPT generated code and their use by developers," *2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR 2024)*, ss. 152-156, 2024.

[26] Ö. Aydın ve E. Karaarslan. (2022, 28 Aralık). *OpenAI ChatGPT generated literature review: Digital twin in healthcare.* [Online]. Erişim: https://ssrn.com/abstract=4308687.

[27] V. Mehta, A. Mathur, A. K. Anjali ve L. Fiorillo, "The application of ChatGPT in the peer-reviewing process," *Oral Oncology Reports*, c. 9, mak. 100227, Şubat 2024.

[28] M. Wermelinger, "Using GitHub Copilot to solve simple programming problems," *SIGCSE 2023: Proceedings of the 54th ACM Technical Symposium on Computer Science Education*, ss. 172-178, Mart 2023.

[29] M. Kytö, "Copilot for Microsoft 365: A comprehensive end-user training plan for organizations," Lisans tezi, İş Bilişim Teknolojileri Bölümü, Haaga-Helia Uygulamalı Bilimler Üniversitesi, Helsinki, Finlandiya, 2024.

[30] A. H. Ali, M. Alajanbi, M. G. Yaseen ve S. A. Abed, "ChatGPT4, DALL·E, Bard, Claude, BERT: Open possibilities," *Babylonian Journal of Machine Learning*, 2023, ss. 17-18, Mart 2023.

[31] Y. Sonoda vd., "Diagnostic performances of GPT-4o, Claude 3 Opus, and Gemini 1.5 Pro in 'Diagnosis Please' cases," *Japanese Journal of Radiology*, c. 42, sy. 11, ss. 1231-1235, Kasım 2024.

[32] K. Basu, R. Sinha, A. Ong ve T. Basu, "Artificial intelligence: How is it changing medical sciences and its future?," *Indian Journal of Dermatology*, c. 65, sy. 5, ss. 365-370, Eylül 2020.

[33] L. Josepth, (2023, 20 Kasım). *AI revolutionizing healthcare, finance, and transportation: A glimpse into the future.* [Online]. Erişim: https://www.linkedin.com/pulse/ai-revolutionizing-healthcare-finance-transportation-glimpse-joseph-tkhff.

[34] D. Leslie, (2019, Haziran). *Understanding artificial intelligence ethics and safety.* [Online]. The Alan Turing Institute. Erişim: https://doi.org/10.5281/zenodo.3240529.

[35] D. Jackson, S. A. Matei ve E. Bertino, (2023, Kasım). *Artificial intelligence ethics education in cybersecurity: Challenges and opportunities: A focus group report*. [Online]. Erişim: https://arxiv.org/abs/2311.00903.

[36] E. Adalı, "Doğal dil işleme," *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, c. 5, sy. 2, 2012.

[37] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain ve J. Gao. (2024, 20 Şubat). *Large Language Models: A Survey*. [Online]. Erişim: http://arxiv.org/abs/2402.06196.

[38] Z. Jiang, F. F. Xu, J. Araki ve G. Neubig, "How can we know what language models know?", *Transactions of the Association for Computational Linguistics*, c. 8, ss. 423-438, 2020.

[39] B. Akbulut, (2023, Ekim). *Büyük dil modelleri: Dilin evrimi ve yapay zekâ (Bölüm 2)*. [Online]. Erişim: https://medium.com/kodluyoruz/büyük-dil-modelleri-dilin-evrimi-ve-yapay-zekâ-bölüm-2-458ddea54f09.

[40] M. O. Topal, A. Baş ve I. van Heerden, "Exploring transformers in natural language generation: GPT, BERT, and XLNet," *International Conference on Interdisciplinary Applications of Artificial Intelligence (ICIDAAI) 2021*, ss. 172-178, Ocak 2021.

[41] L. Floridi ve M. Chiriatti, "GPT-3: Its nature, scope, limits, and consequences," *Minds and Machines*, c. 30, ss. 681-694, Kasım 2020.

[42] OpenAI, (2023, Mart). *GPT-4 Technical Report*. [Online]. Erişim: https://arxiv.org/abs/2303.08774.

[43] R. A. Poldrack, T. Lu ve G. Beguš, (2023, Nisan). *AI-assisted coding: Experiments with GPT-4*. [Online]. Erişim: http://arxiv.org/abs/2304.13187.

[44] A. Eriç, E. G. Özgür, Ö. F. Asker ve N. Bekiroğlu, "ChatGPT ve Sağlık Bilimlerinde Kullanımı", *Celal Bayar Üniversitesi Sağlık Bilimleri Enstitüsü Dergisi*, c. 11, sy. 1, ss. 176-182, Mart 2024.

[45] A. Erdem, "Akıllı Turizmin ChatGPT Tarafından Değerlendirilmesi (Evaluation of Smart Tourism by ChatGPT)," *Journal of Tourism and Gastronomy Studies*, c. 11, sy. 4, ss. 3298-3313, Aralık 2023.

[46] B. Hui, J. Yang, Z. Cui, J. Yang, D. Liu, …, ve J. Lin, (2024, Eylül). *Qwen2.5-Coder technical report*. [Online]. Erişim: http://arxiv.org/abs/2409.12186.

[47] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, …, ve T. Scialom, (2023, Temmuz). *Llama 2: Open foundation and fine-tuned chat models*. [Online]. Erişim: http://arxiv.org/abs/2307.09288.

[48] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, …, ve J. Vranes. (2024, Temmuz). *The Llama 3 Herd of Models*. [Online]. Erişim: http://arxiv.org/abs/2407.21783.

[49] J. Howard ve S. Ruder, (2018, Ocak). *Universal Language Model Fine-tuning for Text Classification*. [Online]. Erişim: http://arxiv.org/abs/1801.06146.

[50] S. Friederich, "A New Fine-Tuning Argument for the Multiverse," *Foundations of Physics*, c. 49, sy. 9, ss. 1011-1021, 2019.

[51] I. Goodfellow, Y. Bengio, ve A. Courville. (2016, 10 Kasım). *Deep Learning*. [Online]. MIT Press. Erişim: https://www.deeplearningbook.org.

[52] A. Xu, J. Smith, L. Wang, M. Brown, K. Lee, … ve S. Garcia. (2024, Ekim). "Generative AI and Retrieval-Augmented Generation (RAG) Systems for Enterprise," *33. ACM Uluslararası Bilgi ve Bilgi Yönetimi Konferansı (CIKM '24) Bildirileri*, Birmingham, Birleşik Krallık, ss. 5599-5602, 21-25 Ekim 2024.

[53] M. Arslan, H. Ghanem, S. Munawar, ve C. Cruz, "A Survey on RAG with LLMs", *Procedia Computer Science*, c. 246, ss. 3781-3790, Kasım 2024.

[54] M. Kulkarni, P. Tangarajan, K. Kim ve A. Trivedi. (2024, 10 Ocak). *Reinforcement Learning for Optimizing RAG for Domain Chatbots*. [Online]. Erişim: http://arxiv.org/abs/2401.06800.

[55] F. Rizzi, "Developing an Enterprise Chatbot using Machine Learning Models: A RAG and NLP Based Approach," Yüksek Lisans Tezi, Data Science and Engineering Programı, Politecnico di Torino, Torino, İtalya, 2024.

[56] P. van Lunteren, "EcoAssist: A no-code platform to train and deploy custom YOLOv5 object detection models," *Journal of Open Source Software*, c. 8, sy. 88, ss. 5581, 2023.

[57] M. Moskal, "No-Code Application Development on the Example of Logotec Ap Studio Platform," *Informatyka*, Automatyka, Pomiary w Gospodarce i Ochronie Srodowiska, c. 11, sy. 1, ss. 54-57, 2021.

[58] T. Koide, N. Fukushi, H. Nakano ve D. Chiba. (2023, 9 Haziran). *Detecting Phishing Sites Using ChatGPT*. [Online]. Erişim: https://arxiv.org/abs/2306.05816.

[59] S. S. Roy, K. V. Naragam ve S. Nilizadeh. (2023, 9 Mayıs). *Generating Phishing Attacks using ChatGPT*. [Online]. Erişim: https://arxiv.org/abs/2305.05133.

[60] S. S. Roy, K. V. Naragam ve S. Nilizadeh. *The Function of ChatGPT in Social Media: According to ChatGPT*. [Online]. Erişim: https://ssrn.com/abstract=4405389.

[61] A. González Sánchez, "Bölüm 3," *Azure OpenAI Service for Cloud Native Applications: Designing, Planning, and Implementing Generative AI Solutions*, 1. Baskı, Cambridge, MA: O'Reilly Media, Ağustos 2024.

[62] Ö. Aydin. (2024, 20 Aralık). *Google Bard Generated Literature Review: Metaverse*. [Online]. Erişim: https://bard.google.com.

[63] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, …, ve Z. Chen. (2023, 17 Mayıs). *PaLM 2 Technical Report*. [Online]. Erişim: http://arxiv.org/abs/2305.10403.

[64] T. R. McIntosh, T. Susnjak, T. Liu, P. Watters ve M. N. Halgamuge. (2023, 18 Aralık). *From Google Gemini to OpenAI Q (Q-Star): A Survey of Reshaping the Generative Artificial Intelligence (AI) Research Landscape*. [Online]. Erişim: https://arxiv.org/abs/2312.10868.

[65] İnnova. (2024, 24 Temmuz). *Kodlama İçin En İyi 5 Büyük Dil Modeli*. [Online]. Erişim: https://www.innova.com.tr/blog/kodlama-icin-en-iyi-5-buyuk-dil-modeli.

[66] Artificial Analysis. (2024, 18 Aralık). *Comparison of AI Models across Quality, Performance, Price*. [Online]. Erişim: https://artificialanalysis.ai/models.

[67] N. Rajaraman, J. Jiao ve K. Ramchandran. (2024, 12 Nisan). *Toward a Theory of Tokenization in LLMs*. [Online]. Erişim adresi: https://arxiv.org/abs/2404.08335.

[68] C. W. Schmidt, V. Reddy, H. Zhang, A. Alameddine, O. Uzan, Y. Pinter ve C. Tanner. (2024, 28 Şubat). *Tokenization Is More Than Compression*. [Online]. Erişim: https://arxiv.org/abs/2402.18376.

[69] A. Verma. (2017, 13 Ekim). *The Future Of Coding Is No Coding At All — Did GitHub CEO Predict Traditional Programming's Death?* [Online]. Erişim: https://fossbytes.com/coding-automation-programming-github-ceo.

[70] H. Liu, C. Zhang, S. Sun, H. Zhou, Y. Xie ve T. Jiang, "Universal No-code Web Design Tool for Substation Digital Monitoring System," *IMCEC 2024 – IEEE 6th Advanced Information Management*, Communicates, Electronic and Automation Control Conference, c., ss. 1391-1394, 2024.

[71] A. Jassy. (2024, 22 Ağustos). *One of the most tedious but critical tasks activity.. – Amazon Q*. [Online]. Erişim: https://www.linkedin.com/posts/andy-jassy-8b1615_one-of-the-most-tedious-but-critical-tasks-activity-7232374162185461760-AdSz.

[72] W. Filipek. (2024, 31 Aralık). *Top No-Code AI Tools of 2024: In-Depth Guide*. [Online]. Erişim: https://buildfire.com/no-code-ai-tools.

[73] BuildFire. (2024, 30 Aralık). *BuildFire App Builder Web Sitesi*. [Online]. Erişim: https://buildfire.com.

[74] M. Annamalai, K. Ravichandran, H. Srinivas, I. Zinkovsky, L. Pan, T. Savor, D. Nagle ve M. Stumm, "Sharding the Shards: Managing Datastore Locality at Scale with Akkio," *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, ss. 445-460, Carlsbad, CA, Ekim 2018.

[75] E. Huovila, "Koneoppimisen käyttö toimitusketjun hallinnassa – case-esimerkki DataRobotilla," Yüksek Lisans Tezi, Mühendislik Bilimleri Fakültesi, Endüstri Mühendisliği Bölümü, Lappeenranta-Lahti Teknoloji Üniversitesi LUT, 2021.

[76] D. Agustian, P. P. G. P. Pertama, P. N. Crisnapati ve P. D. Novayanti, "Implementation of Machine Learning Using Google's Teachable Machine Based on Android," *3rd International Conference on Cybernetics and Intelligent Systems (ICORIS 2021)*, ss. 220-225, 2021.

[77] Lobe. (2024, 30 Aralık). *Lobe AI GitHub Sayfası*. [Online]. Erişim: https://github.com/lobe.

[78] D. Nigenda, Z. Karnin, M. B. Zafar, R. Ramesha, A. Tan, M. Donini ve K. Kenthapadi, "Amazon SageMaker Model Monitor: A System for Real-Time Insights into Deployed Machine Learning Models," *Proceedings of the 28th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ss. 3671-3681, Ağustos 2022.

[79] T. Özdoğan. (2023, 3 Mayıs). *SageMaker Nedir?*. [Online]. Erişim: https://www.linkedin.com/pulse/sagemaker-nedir-tunahan-%C3%B6zdo%C4%9Fan/.

[80] Levity. (2024, 30 Aralık). *Levity AI Web Sitesi*. [Online]. Erişim adresi: https://levity.ai.

[81] K. Sukhanova. (2022, 25 Ekim). *Create No-Code Models with Levity*. [Online]. Erişim: https://techacute.com/no-code-models-levity/.

[82] AI Squared. (2024, 30 Aralık). *Data and AI Integration Platform*. [Online]. Erişim: https://squared.ai/.

[83] CallFluent. (2024, 30 Aralık). *Create Phone Calling AI Agents in 60 Seconds*. [Online]. Erişim: https://callfluent.com/.

[84] N. Kodali, "Tailwind CSS Integration in Angular: A Technical Overview," *International Journal of Innovative Research in Science*, Engineering and Technology, c. 13, sy. 9, ss. 1-14, Eylül 2024.

[85] H. Al Salmi, "Comparative CSS Frameworks," *Multi-Knowledge Electronic Comprehensive Journal for Education and Science Publications (MECSJ)*, c. 66, ISSN: 2616-9185, 2023.

[86] M. C. Klimm, Design Systems for Micro Frontends: An Investigation into the Development of Framework-Agnostic Design Systems Using Svelte and Tailwind CSS, Lisans Tezi, Bilgisayar ve Mühendislik Bilimleri Fakültesi, TH Köln – University of Applied Sciences, Gummersbach, Almanya, Şubat 2021.

[87] P. S. Maratkar ve P. Adkar, "React JS – An Emerging Frontend JavaScript Library," *IRE Journals*, c. 4, sy. 12, ss. 99-102, Haziran 2021.

[88] R. Wieruch. (2024, 12 Kasım). *The Road to React*. [Online]. Erişim: http://leanpub.com/the-road-to-learn-react.

[89] Yuill, S. ve Halpin, H. (2024, 13 Kasım). *Python*. [Online]. Erişim: https://citeseerx.ist.psu.edu/document?doi=1f2ee3831eebfc97bfafd514ca2abb7e2c5c86bb.

[90] G. van Rossum ve F. L. Drake, *An Introduction to Python,* Sürüm 2.5, Network Theory Ltd., ISBN: 978-0954161767, 2006.

[91] A. Rawat, "A Review on Python Programming," *International Journal for Research in Engineering, Science and Management*, c. 3, sy. 12, ss. 8–11, Aralık 2020.

[92] E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, U. Evci, ..., ve H. Larochelle, "Meta-Dataset: A Dataset of Datasets for Learning to Learn from Few Examples," *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[93] H. He ve E. A. Garcia, "Learning from Imbalanced Data," *IEEE Transactions on Knowledge and Data Engineering*, c. 21, sy. 9, ss. 1263–1284, Eylül 2009.

[94] K. Gauen, Y. Shimony, R. F. Dietterich, ve S. J. K. Smith, "Comparison of visual datasets for machine learning," *Proceedings – 2017 IEEE International Conference on Information Reuse and Integration*, IRI 2017, Institute of Electrical and Electronics Engineers Inc., ss. 346–355, Kasım 2017.

[95] J. J. Gómez-Navarro, J. P. Montvez, S. Jerez, P. Jiménez-Guerrero, ve E. Zorita, "What is the role of the observational dataset in the evaluation and scoring of climate models?", *Geophysical Research Letters*, c. 39, sy. 24, ss. 1–10, Aralık 2012.

[96] M. Gualtieri. (2024, 1 Aralık). *Best Practices in User Experience (UX) Design*. [Online]. Forrester. Erişim: https://forrester.com.

[97] M. Orlova. (2024, 1 Aralık). *User experience design (UX design) in a website development: Website redesign*. [Online]. Erişim: https://www.theseus.fi/handle/10024/120948.

[98] A. Ait, J. L. C. Izquierdo, ve J. Cabot, "HFCommunity: A Tool to Analyze the Hugging Face Hub Community," *Proceedings – 2023 IEEE International Conference on Software Analysis*, Evolution and Reengineering, SANER 2023, Institute of Electrical and Electronics Engineers Inc., ss. 728–732, 2023.

[99] Hugging Face. (2024, 3 Aralık). *Transformers Documentations*. [Online]. Erişim: https://huggingface.co/docs/transformers/v4.46.3/tr/index

[100] T. S. Gunawan, A. Ashraf, B. S. Riza, E. V. Haryanto, R. Rosnelly, M. Kartiwi ve Z. Janin, "Development of video-based emotion recognition using deep learning with Google Colab," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, c. 18, sy. 5, ss. 2463-2471, Ekim 2020.

[101] M. Kuroki, "Using Python and Google Colab to teach undergraduate microeconomic theory," *International Review of Economics Education*, c. 38, Kasım 2021.

[102] P. Kanani ve M. Padole, "Deep learning to detect skin cancer using Google Colab," *Int J Eng Adv Technol*, c. 8, sy. 6, ss. 2176–2183, Ağustos 2019.

[103] J. Wang ve Y. Chen, "A Review on Code Generation with LLMs: Application and Evaluation," *Proceedings – 2023 1st IEEE International Conference on Medical Artificial Intelligence*, MedAI 2023, Institute of Electrical and Electronics Engineers Inc., ss. 284–289, 2023.

[104] Wandb. (2024, 3 Aralık). *Weights & Biases*, [Online]. Erişim: https://wandb.ai/.

[105] O. Koç, İ. Yücedağ ve Ü. Şentürk, "Developing the ZOA Platform and Training a Language Model for No-Code Web Development," *Information Technology and Control*, incelenmek üzere gönderildi, 2025.

[106] F. Zhang, S. Tian, Z. Huang, Y. Qiao, ve Z. Liu. (2024, 1 Aralık). *Evaluation Agent: Efficient and Promptable Evaluation Framework for Visual Generative Models*. [Online]. Erişim: http://arxiv.org/abs/2412.09645

[107] ZeroGPU Explorers. (2024, Ekim). *ZeroGPU Spaces*. [Online]. Erişim: https://huggingface.co/zero-gpu-explorers.

[108] Meta. (2024, Kasım 10). *Language Model Page (meta-llama/Llama-2-7b-chat-hf)*. [Online]. Erişim: https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[109] T. Dettmers, A. Pagnoni, A. Holtzman, ve L. Zettlemoyer. (2024, 23 Haziran). *QLORA: Efficient Finetuning of Quantized LLMs*. [Online]. Erişim: https://github.com/TimDettmers/bitsandbytes.

[110] O. Koç. (2024, 12 Aralık). *Language Model Page (osmankoc/llama-2-7b-zoa)*. [Online]. Hugging Face. Erişim: https://huggingface.co/osmankoc/llama-2-7b-zoa

[111] MarkovML. (2023, 19 Aralık). *Model Evaluation Metrics: Methods & Approaches*. [Online]. MarkovML. Erişim: https://www.markovml.com/blog/model-evaluation-metrics

[112] Version1. (2023, 24 Kasım). *AI Metrics: The Science and Art of Measuring Artificial Intelligence*. [Online]. Version1. Erişim: https://www.version1.com/blog/ai-performance-metrics-the-science-and-art-of-measuring-ai/

[113] P. Flach, "Performance Evaluation in Machine Learning: The Good, the Bad, the Ugly, and the Way Forward," *Proceedings of the AAAI Conference on Artificial Intelligence*, c. 33, sy. 1, ss. 9808-9814, Temmuz 2019.

[114] O. Rainio, J. Teuho, ve R. Klén, "Evaluation metrics and statistical tests for machine learning", *Sci Rep*, c. 14, sy 1, Aralık 2024.

[115] Nebius Team. (2024, 13 Aralık). *AI model performance metrics: in-depth guide*. [Online]. Nebius. Erişim: https://nebius.com/blog/posts/ai-model-performance-metrics.

[116] M. Westphal ve W. Brannath, “Evaluation of multiple prediction models: A novel view on model selection and performance assessment”, *Stat Methods Med Res*, c. 29, sy 6, ss. 1728-1745, Haziran 2020.

[117] S. H. Lee, “Performance Evaluation of Machine Learning and Deep Learning-Based Models for Predicting Remaining Capacity of Lithium-Ion Batteries”, *Applied Sciences (Switzerland)*, c. 13, sy 16, Ağustos 2023.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Osman KOÇ

Yabancı Dili : İngilizce

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Y. Lisans	Elektrik-Elektronik ve Bilgisayar Mühendisliği	Düzce Üniversitesi	2025
Lisans	Bilgisayar Mühendisliği	Düzce Üniversitesi	2019
Lise	Bilişim Teknolojileri	Tuncay Azaphan A.T.M.Lisesi	2014

TEZDEN ÇIKAN YAYINLAR

O. Koç, İ. Yücedağ, ve Ü. Şentürk, “The Impact of Artificial Intelligence Enhanced No-Code Software Development Platforms on Software Processes: A Literature Review,” *DÜBİTED*, c.13, sy. 1, ss 383-401, 2025.

DİĞER YAYINLAR

O. Koç, İ. Yücedağ ve Ü. Şentürk, "Developing the ZOA Platform and Training a Language Model for No-Code Web Development," *Information Technology and Control*, incelenmek üzere gönderildi, 2025.