Teknoloji Fakültesi

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

YAPAY ZEKA DESTEKLİ MOBİL EĞİTİM PLATFORMU UYGULAMASI

BİTİRME PROJESİ

Bilgisayar Mühendisliği Bölümü

DANIŞMAN

Dr. Öğr. Üyesi Timur İNAN

İSTANBUL, 2025

Teknoloji Fakültesi

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

YAPAY ZEKA DESTEKLİ MOBİL EĞİTİM PLATFORMU UYGULAMASI

BİTİRME PROJESİ

Bilgisayar Mühendisliği Bölümü

DANIŞMAN

Dr. Öğr. Üyesi Timur İNAN

İSTANBUL, 2025

MARMARA ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Öğrencileri Emirhan Aydın, Bekir Başar Noyan ve Doğukan Kızıltepe tarafından "**YAPAY ZEKA DESTEKLİ MOBİL EĞİTİM PLATFORMU UYGULAMASI**" başlıklı proje çalışması, xxx tarihinde savunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Dr. Öğr. Üyesi xxx xxx Marmara Üniversitesi	**(Danışman)**	(İMZA)……..
Prof. Dr. Xxx xxx Marmara Üniversitesi	(Üye)	(İMZA)……..
Prof. Dr. Xxx xxx Marmara Üniversitesi	(Üye)	(İMZA)……..

ÖNSÖZ

Proje çalışmamız süresince karşılaştığımız bütün problemlerde sabırla yardım ve bilgilerini esirgemeyen, tüm desteğini sonuna kadar yanımızda hissettiğimiz değerli hocamız sayın Dr. Öğr. Üyesi Timur'a, bu proje çalışması fikrinin oluşması ve ortaya çıkmasındaki önerisi ve desteğinden dolayı en içten teşekkürlerimizi sunarız.

İÇİNDEKİLER

ÖZET

AI DESTEKLİ MOBİL EĞİTİM PLATFORMU UYGULAMASI

Bu proje, yabancı öğrencilerin Türkçeyi öğrenmesini kolaylaştıran yapay zeka destekli bir eğitim platformu geliştirmeyi amaçlamaktadır. iOS için Swift ile geliştirilen uygulama, kullanıcıların dil seviyesine göre çeşitli içerikler sunacak, ilerlemelerini takip edecek ve anlık geri bildirim sağlayacaktır. Yapay zeka ile öğrenciler interaktif alıştırmalar yapabilecek, sorular sorarak gerçek zamanlı yanıtlar alabileceklerdir. Kullanıcı deneyimi ve öneri sistemi sürekli olarak değerlendirilerek, öğrenme sürecinin daha verimli ve etkili hale gelmesi sağlanacaktır.

Haziran, 2025 **Öğrenciler**

ABSTRACT

DEVELOPMENT OF AN AI-ASSISTED EDUCATION PLATFORM USING SWIFT

This study presents the design and development of an AI-assisted education platform for iOS using Swift, specifically aimed at helping foreign students learn Turkish. With advancements in artificial intelligence and natural language processing, personalized and interactive learning experiences have become increasingly important. This project enhances learning efficiency by providing customized educational content based on students' progress and preferences, while enabling real-time question-and-answer interactions with AI.

The proposed system utilizes artificial intelligence to analyze user inputs, track learning habits, and deliver tailored study materials according to each student's needs. AI-driven interactions allow students to ask and answer questions in real time, receive instant feedback, and engage with interactive exercises. The Swift programming language is used for application development, while API integration ensures smooth communication between the mobile application and the AI model.

The effectiveness of the system will be evaluated using user engagement metrics and response accuracy, ensuring continuous improvement.

By integrating artificial intelligence into language education, this project bridges the gap between traditional and technology-driven learning methods, providing an intelligent, accessible, and efficient platform for learners of Turkish. Future improvements include expanding AI capabilities, integrating voice-based interactions, and enhancing content recommendation algorithms to offer an even more immersive learning experience.

June, 2025 **Students**

KISALTMALAR

AI: Artificial Intelligence

API: Application Programming Interface

iOS: iPhone Operating System

1. GİRİŞ

Teknolojinin hızla gelişmesi, eğitim sistemlerini de dijital dönüşüme yönlendirmiş ve bireyselleştirilmiş öğrenme yaklaşımlarının önemini artırmıştır. Geleneksel eğitim yöntemleri genellikle tüm öğrencilere aynı müfredatı sunarken, öğrencilerin bireysel öğrenme hızları, seviyeleri ve ilgi alanları çoğu zaman göz ardı edilmektedir.

Bu proje kapsamında, OpenAI tarafından geliştirilen ChatGPT modeli entegre edilerek yapay zekâ destekli bir eğitim platformu geliştirilmiştir. Swift programlama dili ile geliştirilen bu iOS tabanlı platform, kullanıcıdan başta aldığı seviye bilgisine göre eğitim içeriklerini sunmakta ve böylece öğrencilerin seviyelerine uygun, daha etkili bir öğrenme deneyimi sağlamaktadır.

Uygulama; Reading, Listening, Writing ve Speaking olmak üzere dört temel dil becerisine odaklanmakta ve her beceri için seviyelere ayrılmış dersler sunmaktadır. Kullanıcı, uygulama içerisinde yapay zekâya gerçek zamanlı olarak soru yöneltebilmekte ve aldığı geri bildirimlerle öğrenme sürecine aktif şekilde katılım sağlayabilmektedir.

Bu bağlamda, proje geleneksel öğrenme ile teknolojiyi birleştirerek, öğrenciye hem yapay zekâ destekli etkileşim hem de seviye bazlı içerik sunan bütünleşik bir dil öğrenme platformu ortaya koymaktadır.

1.1. Proje Çalışmasının Amacı ve Önemi

Bu çalışmanın temel amacı, öğrencilerin öğrenme süreçlerini kişiselleştiren ve yapay zeka ile desteklenmiş bir mobil eğitim platformu geliştirmektir. Özellikle yabancı öğrencilerin Türkçeyi öğrenmesini kolaylaştırmak için tasarlanan bu platform, kullanıcıların ihtiyaçlarına göre şekillenen dinamik bir öğrenme deneyimi sunmayı hedeflemektedir.

Platformun başlıca hedefleri şunlardır:

- **Öğrencilerin ihtiyaçlarına göre öğrenme yolları oluşturmak:** Yapay zeka, öğrencilerin seviyelerine uygun ders içerikleri önererek onların öğrenme sürecini optimize edecektir.
- **Anlık geri bildirim sağlamak:** Yapay zeka destekli etkileşimli sistem sayesinde öğrenciler, yapay zekaya anında sorular sorabilecek ve gerçek zamanlı yanıtlar alarak öğrenme süreçlerini hızlandırabileceklerdir.
- **Öğrenme sürecini kullanıcı seviyesine göre geliştirmek:** Platform, kullanıcıdan başta aldığı seviye bilgisine göre uygun içerikleri sunar ve bu doğrultuda sabit bir öğrenme yolu sağlar.

Bu proje, geleneksel eğitim yöntemlerinden farklı olarak öğrencilerin başlangıç seviyelerine göre yapılandırılmış, kişiselleştirilmiş bir öğrenme süreci sunmayı hedeflemektedir. Yapay zeka destekli etkileşimli sistem sayesinde öğrenciler, gerçek zamanlı olarak sorular sorup yanıtlar alarak öğrenme sürecine aktif şekilde katılabilecek ve dil öğrenimini daha etkili ve motive edici bir şekilde sürdürebilecektir.

1.2. Projenin Kapsamı

Bu proje, yapay zekâ destekli bir iOS mobil uygulaması aracılığıyla yabancı öğrencilerin Türkçe öğrenme süreçlerini desteklemeyi amaçlamaktadır. Swift programlama dili kullanılarak geliştirilen uygulama, MVVM-C (Model-View-ViewModel-Coordinator) mimarisiyle yapılandırılmış olup modüler, test edilebilir ve sürdürülebilir bir sistem sunmaktadır.

Proje kapsamında geliştirilen mobil uygulama, dört temel dil becerisine odaklanmaktadır: **Reading (Okuma), Listening (Dinleme), Writing (Yazma)** ve **Speaking (Konuşma)**. Kullanıcılardan ilk girişte seviye bilgisi alınmakta ve bu doğrultuda içerikler belirlenmektedir. Her kurs, A1'den C2'ye kadar farklı seviyelere ayrılmış dersler içermektedir. Ders içerikleri; metin, ses, çoktan seçmeli soru ve serbest metin gibi çeşitli formatlarda sunulmaktadır.

Yapay zekâ entegrasyonu sayesinde kullanıcılar, özellikle yanlış yanıt verdiklerinde veya serbest metin girişi yaptıklarında anlık geri bildirim alabilmekte ve bu geri bildirimler ChatGPT tabanlı doğal dil işleme modeli ile sağlanmaktadır. Ayrıca kullanıcılar, uygulama içindeki özel AI sohbet ekranı aracılığıyla doğrudan yapay zekâ ile iletişim kurabilmektedir.

Projenin kapsamı yalnızca mobil uygulama arayüzü ile sınırlı olmayıp; aynı zamanda .NET 8 ile geliştirilen bir back-end servisi, PostgreSQL veritabanı, Entity Framework tabanlı veri yönetimi gibi teknolojilerin entegrasyonunu da içermektedir. Kullanıcı bilgileri, seviye tercihleri ve ilerlemeleri güvenli bir şekilde saklanmakta ve gerektiğinde analiz için kullanılabilmektedir.

Bu yönüyle proje, yapay zekâ desteği ile dil öğrenimini kolaylaştıran, kişisel seviyeye göre şekillenen, teknik olarak sürdürülebilir ve kullanıcı dostu bir dijital eğitim platformu sunmayı hedeflemektedir.

1.3. Proje Katkıları

Bu proje, dil eğitimi alanında mobil teknolojiler ve yapay zekânın entegre edilmesiyle hem teknik hem de pedagojik açıdan çeşitli katkılar sunmaktadır:

- **Yapay zekâ destekli etkileşim:** OpenAI entegrasyonu sayesinde kullanıcılar uygulama içinde gerçek zamanlı olarak yapay zekâ ile iletişim kurabilmekte ve dil öğrenimini daha etkileşimli hâle getirmektedir.
- **Seviye bazlı içerik sunumu:** Kullanıcının girişte seçtiği seviye bilgisine göre özelleştirilmiş içerikler sunularak, geleneksel tekdüze müfredat yapısından uzak, daha kişisel bir öğrenme deneyimi sağlanmıştır.
- **Mobil erişilebilirlik:** iOS uyumlu olarak geliştirilen uygulama, kullanıcılara her yerden ulaşılabilir bir öğrenme ortamı sunarak zaman ve mekândan bağımsız eğitim imkânı sağlamaktadır.
- **Modüler mimari yapı:** MVVM-C tasarım deseni ile geliştirilen yapı, uygulamanın sürdürülebilirliğini, test edilebilirliğini ve genişletilebilirliğini artırmakta; benzer projeler için örnek bir yazılım mimarisi sunmaktadır.
- **Çok yönlü dil becerisi gelişimi:** Uygulama; okuma, dinleme, yazma ve konuşma olmak üzere dört temel dil becerisini kapsayan yapısıyla bütünsel bir Türkçe öğrenme süreci sunmaktadır.
- **Anlık geri bildirimle öğrenme:** Kullanıcının yanlış yanıtlarına karşı yapay zekâ tarafından verilen açıklayıcı geri bildirimler, öğrenme sürecini destekleyici nitelikte yapılandırılmıştır.

1.4. Kullanılan Yöntemler

Bu çalışmada, yapay zekâ destekli bir mobil eğitim uygulaması geliştirmek amacıyla yazılım geliştirme yaşam döngüsü adımları takip edilmiştir. Uygulama, bireysel dil öğrenme sürecini destekleyen bir yapı üzerine kurulmuş ve kullanıcı deneyimini artıracak şekilde modüler olarak tasarlanmıştır.

Projede izlenen yöntemsel yaklaşım şu şekilde özetlenebilir:

- **Analiz ve Planlama:** Projenin hedef kitlesi belirlenmiş, kullanıcı ihtiyaçları analiz edilmiştir. Özellikle yabancı öğrencilerin Türkçe öğrenme sürecinde karşılaştığı sorunlara odaklanılmıştır.
- **Katmanlı Mimari Kullanımı (Back-End):** Sunucu tarafı geliştirmelerinde katmanlı mimari tercih edilmiştir. Veri erişimi için Entity Framework'ün Code-First yaklaşımı kullanılmış ve PostgreSQL veritabanı ile bağlantı sağlanmıştır.
- **Swift ile Uygulama Geliştirme:** iOS uygulaması Swift programlama dili ile geliştirilmiş, kullanıcı arayüzleri UIKit ve SwiftUI bileşenleri ile oluşturulmuştur.
- **MVVM-C Tasarım Deseni:** Uygulama mimarisinde MVVM-C (Model-View-ViewModel-Coordinator) yapısı benimsenmiştir. Bu yapı sayesinde katmanlar arası ayrım sağlanmış, modülerlik ve test edilebilirlik artırılmıştır.
- **OpenAI API Entegrasyonu:** ChatGPT tabanlı yapay zekâ sisteminin entegrasyonu yapılmıştır. Kullanıcı, uygulama içerisindeki soru-cevap ekranlarında doğal dil ile sorular sorarak yapay zekâdan gerçek zamanlı yanıtlar alabilmektedir.
- **Kütüphane ve Bileşen Kullanımı:** Uygulamada Lottie kütüphanesi ile animasyonlar eklenmiş, kullanıcı ilerlemesini göstermek için Circular Progress Bar gibi üçüncü parti kütüphaneler entegre edilmiştir.
- **Sayfa ve Akış Tasarımı:** Kullanıcı arayüzü; giriş, kayıt, seviye seçimi, ana sayfa, AI sohbet ekranı, profil ekranı ve ders ekranları şeklinde yapılandırılmıştır. Sayfalar arası yönlendirme Coordinator yapısı ile sağlanmıştır.

Bu yöntemsel yaklaşım, hem yazılım mühendisliği ilkelerine uygun bir yapı ortaya koymuş hem de kullanıcı merkezli bir öğrenme deneyimi sunmayı mümkün kılmıştır.

1.5. Çalışmanın Sınırlılıkları

Bu çalışmada geliştirilen yapay zekâ destekli eğitim platformu, kullanıcıya seviye bazlı içerik sunarak Türkçe öğrenme sürecini kolaylaştırmayı amaçlamaktadır. Ancak mevcut sürümde bazı teknik ve içeriksel sınırlılıklar mevcuttur:

- **Uyarlamalı (adaptif) öğrenme desteği yoktur:** Uygulama, kullanıcıdan başta alınan seviye bilgisine göre sabit içerikler sunmaktadır. Öğrencinin ilerleyişine veya performansına göre otomatik olarak içerik güncelleme veya zorluk ayarı yapılmamaktadır.
- **Geri bildirimler sabit formatlıdır:** Yanlış cevaplara karşı verilen yapay zekâ yanıtları etkili olsa da, sistem kullanıcı geçmişine dayalı kişiselleştirilmiş geri bildirim üretmemektedir.
- **Çoklu dil desteği sınırlıdır:** Uygulama yalnızca Türkçe öğrenen kullanıcıları hedeflemektedir. Arayüz dili veya çoklu dil destek mekanizması bulunmamaktadır.
- **Sesli komutlar sınırlı seviyede değerlendirilmektedir:** Speaking modülünde kullanıcıdan alınan sesli yanıtlar temel doğruluk kontrolüne dayalıdır. Derinlemesine telaffuz analizi veya ses tonlaması değerlendirmesi yapılmamaktadır.
- **Sadece iOS platformu desteklenmektedir:** Uygulama yalnızca iOS işletim sistemine sahip cihazlarda çalışacak şekilde geliştirilmiştir. Android veya web sürümü bulunmamaktadır.
- **Kapsamlı kullanıcı verisi analiz edilmemektedir:** Kullanıcının uzun vadeli performansı, öğrenme davranışları ya da istatistiksel ilerlemesi sistem tarafından analiz edilmemektedir.

Bu sınırlılıklar, projenin ilerleyen aşamalarında giderilerek daha gelişmiş bir öğrenme deneyimi sunulması hedeflenmektedir.

2. GELİŞTİRME SÜRECİ VE TEKNİK DETAYLAR

Mobil erişilebilirliği artırmak: Swift kullanılarak geliştirilen bu platform, her zaman ve her yerden erişilebilecek şekilde tasarlanacaktır.

Bu hedefler doğrultusunda geliştirilecek eğitim platformu, kullanıcıların seviyelerine göre yapılandırılmış, erişilebilir ve etkileşimli bir öğrenme deneyimi sunmayı amaçlamaktadır.

2.1. Swift ile Mobil Uygulama Geliştirme

Proje, Swift programlama dili kullanılarak geliştirilecektir. Swift, iOS uygulama geliştirme için güçlü ve optimize bir dil olup, eğitim platformunun performanslı çalışmasını sağlayacaktır.

Uygulamanın temel bileşenleri şunlardır:

- Kullanıcı Arayüzü (UI): SwiftUI veya UIKit kullanılarak tasarlanacaktır.
- Veri Depolama: Kullanıcı verileri PostgreSQL üzerinde saklanacaktır.
- API Bağlantıları: ChatGPT entegrasyonu için OpenAI API'si kullanılacaktır.

2.1.1 MVVM-C Tasarım Deseni

MVVM-C (Model-View-ViewModel-Coordinator), bir tasarım deseni olarak kullanılan bir yapıdır. Bu yapı, Model-View-ViewModel (MVVM) desenini geliştirir ve uygulamaya bir Coordinator ekleyerek sayfa yönetimini daha etkili bir şekilde ele alır.

2.1.1.1 Model Katmanı:

- Model, uygulamanın temel veri yapısını ve iş mantığını temsil eder.
- Verilerin saklanması, işlenmesi ve uygulama kurallarının uygulanması bu sınıfta gerçekleşir.
- Kullanıcı arayüzü (View) veya kullanıcı etkileşimi (Controller) ile doğrudan etkileşime girmez.

2.1.1.2 ViewModel Katmanı:

- View ile Model arasında arabuluculuk yapar.
- ViewModel, View için gerekli olan veriyi sağlar, kullanıcı etkileşimlerini işler ve Model ile olan iletişimi yönetir

- Genellikle View'den bağımsız olarak test edilebilir ve tekrar kullanılabilir olmalıdır.
- Servis ve View arasındaki bağlantıyı sağlar. Servisten gelen veriler ViewModel tarafından yönetilen Observable değişkenlere atanır ve View istediği zaman bu değişkenleri kullanarak güncellemeleri alır.

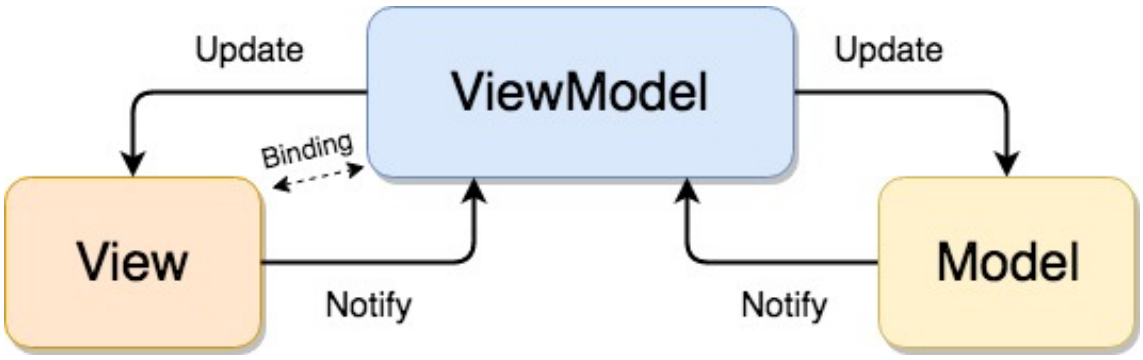
2.1.1.3 View Katmanı:

- Kullanıcı arayüzünü temsil eder ve kullanıcının gördüğü grafiksel veya metinsel öğeleri içerir.
- ViewModel'e istenen işlemleri bildirir ve ViewModel, bu işlemleri gerçekleştirir.
- View, tembel çalışma prensibiyle daha az iş yapar ve uygulama mantığını ViewModel'a bırakır.

2.1.1.4 Coordinator Katmanı:

- Coordinator sınıfı, sayfa yönetimini sağlar.
- ViewModel, View'den aldığı bilgiler doğrultusunda Coordinator'a haber verir, ardından Coordinator, sayfa değişikliklerini yönetir.
- Coordinator kullanımı, test edilebilirliği arttırır ve View'in sorumluluklarını azaltarak kodun daha temiz olmasını sağlar.

Bu yapıyı kullanmanın avantajlarından biri, her bileşenin belirli bir sorumluluğa odaklanarak, kodun daha okunabilir, bakımı daha kolay ve test edilebilir olmasını sağlamaktır. [1]



Görsel 1 - Örnek MVVM Mimarisi

2.1.1.5 Nib/Xib Yapısı:

XIB ve NIB dosyaları, iOS uygulamalarında görsel tasarımı yönetmek için güçlü araçlar sunar. Modüler yapıları ve hızlı geliştirme olanakları sayesinde, her iki dosya türü de küçük, bağımsız arayüz bileşenlerini yönetmek için uygundur. Bu yapılar, özellikle karmaşık kullanıcı arayüzlerinin daha düzenli bir şekilde geliştirilmesine olanak tanır.

NIB dosyaları, iOS ve macOS uygulamalarında görsel arayüz tasarımlarını içeren dosyaların derlenmiş hâlidir. XIB dosyası, Interface Builder kullanılarak oluşturulan ve tasarım öğelerinin yerleştirildiği bir dosyadır. Ancak XIB dosyaları doğrudan çalıştırılamaz; bu dosyalar, derlenerek bir NIB dosyasına dönüştürülür. NIB dosyası, uygulama çalışırken daha hızlı yüklenmesi ve daha verimli olması açısından önemli bir avantaj sağlar.

NIB dosyalarının kullanımı, özellikle arayüz bileşenlerinin modüler hale getirilmesini ve yeniden kullanılabilirliğini sağlar. Tasarlanan bir kullanıcı arayüzü, NIB dosyasına dönüştürüldükten sonra, başka yerlerde de aynı tasarım kolayca yüklenip kullanılabilir. Bu, uygulamanın farklı bölümlerinde tekrar kullanılabilen özel view'ler veya view controller'lar gibi bileşenlerin oluşturulmasına olanak tanır.

2.2 Uygulama Mimarisi ve Ekran Yönlendirme Yapısı

Bu bölümde, uygulamanın genel mimari yapısı ve ekranlar arası geçişlerin nasıl yönetildiği açıklanacaktır. Proje, MVVM-C (Model-View-ViewModel-Coordinator) tasarım deseni kullanılarak modüler, test edilebilir ve sürdürülebilir bir yapıda geliştirilmiştir. Sayfa geçişleri, Coordinator yapısı üzerinden merkezi bir şekilde kontrol edilmekte ve kullanıcı etkileşimleri sonucunda ilgili ekranlara yönlendirme sağlanmaktadır. Bu yapı sayesinde, kullanıcı arayüzü bileşenleri yalnızca görsel sunumla ilgilenirken, uygulama akışı ayrı sınıflar aracılığıyla düzenli bir biçimde yönetilmektedir.

2.2.1. ApplicationCoordinator Yapısı ve Ekran Yönlendirme Sistemi

Uygulamanın genel akışının kontrol edilmesi ve ekranlar arası geçişin yönetilmesi amacıyla merkezi bir ApplicationCoordinator sınıfı kullanılmıştır. MVVM-C mimarisinde Coordinator yapısı, sayfa geçişlerini View ve ViewModel katmanlarından ayırarak yazılımın modülerliğini, test edilebilirliğini ve sürdürülebilirliğini artırmaktadır.

ApplicationCoordinator sınıfı, tüm uygulama boyunca tekil (singleton) bir nesne olarak çalışmakta ve kullanıcı akışını aşağıdaki şekillerde yönetmektedir:

- **Tab Bar Başlatma:** initTabBar fonksiyonu ile uygulamanın ana navigasyon yapısı olan TabBarCoordinator başlatılmakta ve kullanıcı varsayılan olarak ana sayfa sekmesine yönlendirilmektedir.
- **Ekran Geçişleri:** Kullanıcının etkileşimleri sonucunda farklı modüllere geçiş yapmak gerektiğinde pushFromTabBarCoordinator veya pushFromTabBarCoordinatorAndVariables fonksiyonları aracılığıyla ilgili Coordinator'lar başlatılmakta ve ViewController'lar navigation stack'e eklenmektedir.
- **Giriş Akışı Yönetimi:** Kullanıcının kayıt olma ya da giriş yapma işlemleri özel olarak pushToRegisterScreen ve pushToMainLoginScreen metodları ile kontrol altına alınmıştır. Bu sayede, girişle ilişkili ekranlar yalnızca LoginScreenCoordinator navigasyon yapısı altında yönetilmektedir.
- **Kurs Giriş Yönlendirmesi:** handleCourseEntry fonksiyonu, kullanıcının seçtiği kurs ve seviyeye uygun olarak ilgili CourseScreenCoordinator'u başlatır ve ViewModel ile birlikte ilgili derse yönlendirme gerçekleştirir.

Coordinator yapısında kullanılan bu merkezi kontrol mekanizması sayesinde, ViewController'lar herhangi bir doğrudan yönlendirme sorumluluğu taşımaz. Bu durum, **ViewController'ların yalnızca kullanıcı arayüzü mantığıyla ilgilenmesini sağlar** ve sayfa geçişlerinin test edilebilirliğini artırır.

Ayrıca projede tanımlanan Coordinator protokolü, tüm Coordinator sınıflarının ortak davranış kalıplarını belirlemektedir. Bu yapı sayesinde her ekranın kendi yöneticisi vardır, ancak bu yöneticiler ApplicationCoordinator tarafından merkezi biçimde koordine edilir.

2.2.2 TabBarCoordinator Yapısı

Uygulamanın üç temel ekranını barındıran ana sekme (tab bar) yapısı, TabBarCoordinator sınıfı aracılığıyla yönetilmektedir. Bu yapı, **kullanıcı deneyimini sadeleştirmek ve farklı modüllere hızlı erişim sağlamak amacıyla** kullanılmaktadır.

TabBarCoordinator, UITabBarController üzerinden; ana sayfa (MainScreenCoordinator), yapay zekâ sohbet ekranı (AIScreenCoordinator) ve profil ekranı (ProfileScreenCoordinator) olmak üzere üç ana bölümü yönetmektedir. Her sekme, kendi bağımsız Coordinator yapısıyla entegre bir şekilde çalışır. Böylece, her sekmeye ait navigation yapısı ayrı ayrı tanımlanabilir ve gerektiğinde bağımsız olarak yönetilebilir.

Uygulama başlatıldığında start() fonksiyonu ile:

- Üç farklı coordinator başlatılır,
- Her biri kendi ViewController'ını hazırlar ve birer UINavigationController içine yerleştirilir,
- Bu navigation controller'lar UITabBarItem ile etiketlenir ve tabBarController'a atanır.

Bu yapı sayesinde:

- Uygulamanın ana ekran yapısı modüler ve genişletilebilir şekilde kurulmuş olur.
- Her sekmenin bağımsız navigasyon geçmişi korunur.
- MVVM-C mimarisi kapsamında **ekran ayrışması ve yönlendirme yönetimi** başarıyla sağlanmış olur.

TabBarCoordinator, ApplicationCoordinator tarafından çağrılarak uygulamanın temel yapısını ayağa kaldırmakta ve kullanıcıyı varsayılan olarak ana sayfa sekmesine yönlendirmektedir.

2.2.3 Diğer Coordinator Yapıları

Uygulama içerisindeki her temel ekran, kendi bağımsız Coordinator sınıfı aracılığıyla yönetilmektedir. Bu yapı, MVVM-C mimarisine uygun olarak ekran geçişlerinin ViewController'lardan tamamen ayrılmasını ve yönetimin merkezi bir yapıya devredilmesini sağlar.

Tüm bu coordinator yapıları benzer bir mantıkla çalışmaktadır:

- getInstance metodu ile singleton olarak erişilir.
- start() fonksiyonu içerisinde ilgili ViewController örneği oluşturulur ve ViewModel ile ilişkilendirilir.

2.3. ViewController Yapıları ve Ekran Fonksiyonları

Bu bölümde, uygulamanın kullanıcı arayüzünü oluşturan ViewController'lar gruplandırılarak açıklanacaktır. Her bir ekran, MVVM-C mimarisine uygun olarak ViewModel ve Coordinator katmanlarıyla birlikte çalışmakta, böylece ekranın yalnızca kullanıcı arayüzünden sorumlu olması sağlanmaktadır.

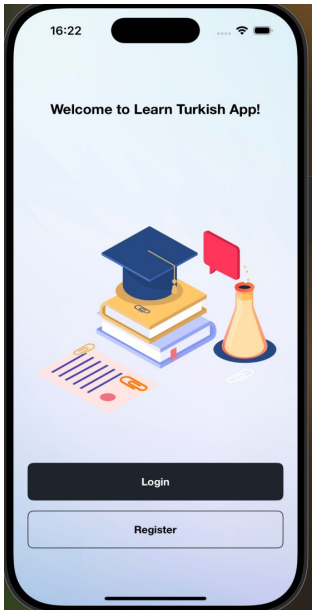
2.3.1 Uygulama Açılış Ekranı – LoginScreenViewController

LoginScreenViewController, uygulamanın açılışında karşılaşılan ilk ekrandır ve kullanıcıyı “Login veya "Register" seçeneklerine yönlendirmek amacıyla tasarlanmıştır. Bu ekran, herhangi bir veri işleme veya kullanıcı doğrulama işlemi gerçekleştirmez; yalnızca kullanıcıyı uygun ekrana yönlendiren bir geçiş noktası olarak görev yapar.

Ekran, MVVM-C mimarisine uygun biçimde yapılandırılmıştır. Kullanıcı tarafından butonlara tıklanması durumunda, ApplicationCoordinator sınıfı aracılığıyla ilgili ekranlara geçiş yapılır:

- “Login” butonu, MainLoginScreenViewController ekranına yönlendirir.
- “Register” butonu ise RegisterScreenViewController ekranına geçişi sağlar.

Ekranda kullanıcı etkileşimini artırmak için bir **Lottie animasyonu** kullanılmıştır. loginAnimation adlı animasyon, ekrana görsel hareketlilik katarak kullanıcıya modern bir giriş deneyimi sunmaktadır.



Görsel 2 - Uygulama Açılış Ekranı

2.3.2 Giriş Ekranı - MainLoginScreenViewController

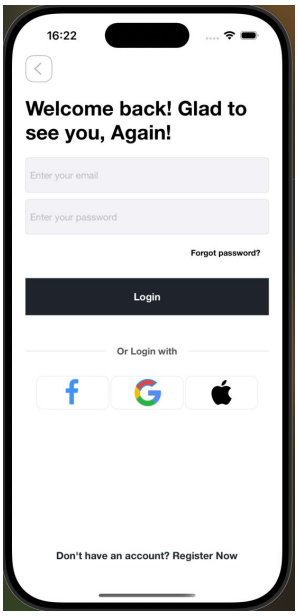
MainLoginScreenViewController, uygulamada kullanıcıdan e-posta ve şifre bilgilerini alarak kimlik doğrulama işleminin gerçekleştiği ekrandır. Bu ekran, kullanıcı girişine ait temel işlevselliği barındırmakta ve ViewModel ile etkileşim içinde çalışmaktadır.

Kullanıcı, login butonuna tıkladığında emailText ve passwordText alanlarındaki veriler alınmakta, geçerlilik kontrolü yapılmakta ve MainLoginScreenViewModel aracılığıyla sunucu tarafına gönderilmektedir. Doğrulama işlemi başarılı olursa, ApplicationCoordinator üzerinden initTabBar çağrılarak kullanıcı uygulamanın ana ekranına yönlendirilir. Hatalı girişlerde ise kullanıcıya showAlert fonksiyonu aracılığıyla hata mesajı gösterilmektedir.

Arayüzde ayrıca özel bir CustomBackButtonView kullanılmıştır. Bu bileşen, kullanıcının bir önceki ekrana (LoginScreenViewController) dönmesini sağlar. registerNowButton ise kullanıcıyı RegisterScreenViewController ekranına yönlendirmektedir.

Bu yapı sayesinde:

- Kullanıcıdan alınan veriler sadece UI katmanında tutulur.
- Giriş işlemleri ViewModel üzerinden gerçekleştirilir.
- Ekran yönlendirmeleri Coordinator katmanına devredilmiştir.



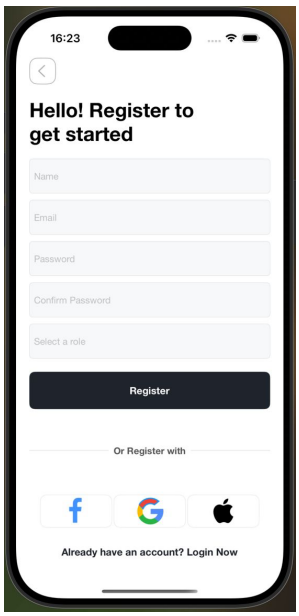
Görsel 3 - Login Ekranı

2.3.3 Kayıt Ekranı - RegisterScreenViewController

RegisterScreenViewController, uygulamaya yeni kullanıcı kaydı yapılmasını sağlayan ekrandır. Kullanıcıdan; ad, e-posta, şifre, şifre tekrarı ve kullanıcı rolü (Teacher veya Student) bilgileri alınır. Arayüz, UIKit bileşenleri ile oluşturulmuş olup; kullanıcıya daha anlaşılır ve modern bir deneyim sunmak amacıyla giriş alanlarının arka planları özelleştirilmiş, kayıt butonu stilize edilmiştir.

Kullanıcı rolü belirleme işlemi, UIPickerView ile gerçekleştirilir. PickerView'a bağlı olan roleTextField, kullanıcıya sadece ön tanımlı seçenekler sunar. Seçilen rol bilgisi, arka planda numerik bir değere çevrilerek sistemde kayıtlı olan Register modeline aktarılır.

Ekrandaki "Register" butonuna tıklanmasıyla birlikte, girilen bilgiler RegisterScreenViewModel aracılığıyla ilgili servis katmanına gönderilir. İşlem başarılı olursa, kullanıcıya "kayıt başarılı" mesajı gösterilir ve ApplicationCoordinator üzerinden MainLoginScreenViewController ekranına yönlendirilir. İşlem başarısız olursa, kullanıcıya hata mesajı gösterilir. "Login Now" butonu da doğrudan giriş ekranına geçişi sağlar.



Görsel 4 - Register Ekranı

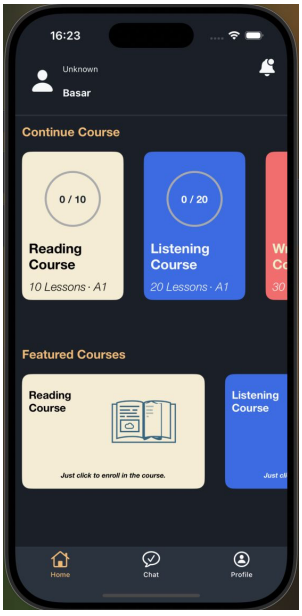
2.3.4 Ana Ekran - MainScreenViewController

MainScreenViewController, kullanıcının uygulamaya giriş yaptıktan sonra karşılaştığı ana ekrandır. Bu ekran, hem kullanıcının ismini hem de kayıtlı olduğu kurslardaki ilerlemelerini görsel olarak sunar. Arayüzde iki farklı UICollectionView kullanılmıştır:

- Birincisi, her bir kursun (Reading, Listening, Writing, Speaking) genel ilerleme durumunu göstermek için kullanılır. Burada ders sayısı, tamamlanan ders sayısı ve ilerleme yüzdesi gibi bilgiler gösterilmektedir. İlerleme verileri progressView aracılığıyla grafiksel olarak ifade edilmiştir.
- İkinci koleksiyon görünümünde (collectionView2), her bir kursu temsil eden kartlar yer alır. Bu hücrelerde Lottie kütüphanesi kullanılarak ilgili kurslara özel animasyonlar gösterilmekte ve kullanıcı etkileşimini artıran bir arayüz sunulmaktadır.

Kullanıcı, bu hücrelerden birine tıkladığında, didSelectItemAt fonksiyonu aracılığıyla seçilen kursa uygun CourseScreenViewModel oluşturulmakta ve ApplicationCoordinator üzerinden ilgili CourseScreenCoordinator'a yönlendirme yapılmaktadır.

HamburgerMenuManager yardımıyla uygulamanın yan menü sistemi kontrol altına alınmıştır.

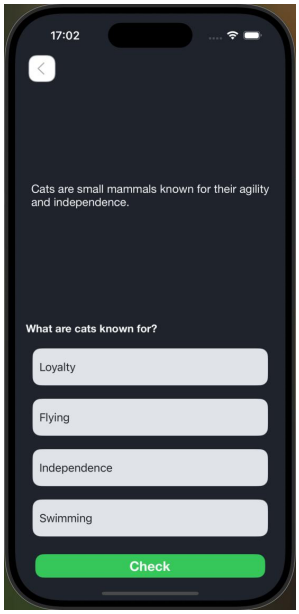


Görsel 5 - Ana Sayfa Ekranı

2.3.5 Reading Kursu Ekranı - ReadingScreenViewController

ReadingScreenViewController, kullanıcının seviyesine uygun metinleri okuyarak anlam çıkarma becerisini geliştirmesine yönelik olarak tasarlanmış bir derse ait ekran yapısıdır. Bu ekran, kullanıcının verilen bir paragrafa ait soruya çoktan seçmeli cevap vermesini sağlar ve kullanıcı cevabını kontrol ettikten sonra, yapay zekâ tarafından üretilmiş açıklayıcı geri bildirimleri popup olarak sunar.

Bu yapı sayesinde, klasik çoktan seçmeli test sisteminin ötesine geçilerek, yapay zekâ destekli açıklayıcı geri bildirimler aracılığıyla kullanıcıya daha derinlemesine öğrenme deneyimi sunulmaktadır.



Görsel 6 - Reading Kursu Ekranı

2.3.6 Listening Kursu Ekranı - ListeningScreenViewController

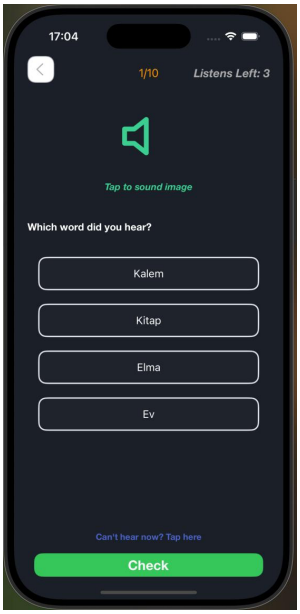
ListeningScreenViewController, kullanıcının dinlediği Türkçe sesleri anlama ve doğru yanıtı seçme becerisini geliştirmeyi hedefleyen dinleme etkinlik ekranıdır. Bu ekran, yapay zekâ destekli geri bildirim sistemi ile desteklenerek, geleneksel çoktan seçmeli sorulara yeni bir boyut kazandırmaktadır.

Kullanıcı her soruda:

- Bir ses dosyasını dinler ,
- Soruyu okur ve dört seçenekten birini seçer,
- Seçimini yaptıktan sonra checkButton ile cevabını gönderir.

Sesi dinlemek için kullanıcı, ekrandaki Lottie animasyonuna tıklayarak yapay zekâ destekli metinden sese sistemiaracılığıyla hedef sesi oynatır. Dinleme sayısı üçle sınırlandırılmıştır ve kalan hakkı listensLeftLabel ile gösterilmektedir.

Sonuç olarak, ListeningScreenViewController kullanıcıya aktif dinleme yaparak seçim yapma, dinleme becerisini ölçme ve yapay zekâ desteğiyle eksik yönlerini öğrenme fırsatı sunar.



Görsel 7 - Listening Kursu Ekranı

2.3.7 Writing Kursu Ekranı - WritingScreenViewController

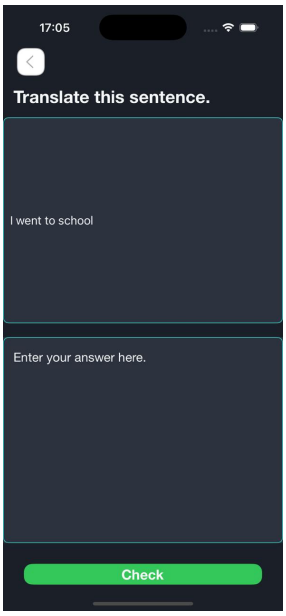
WritingScreenViewController, kullanıcının verilen bir cümleyi Türkçeye doğru şekilde çevirmesini amaçlayan yazma etkinliği ekranıdır. Bu ekran, yapay zekâ destekli geri bildirim mekanizması ile klasik çeviri etkinliklerine etkileşim kazandırır.

Kullanıcı arayüzünde:

- Metin alanında çevirisi yapılacak İngilizce cümle gösterilir,
- Kullanıcı, çevirisini metin kutusuna yazar,
- "Check" butonuna tıkladığında yapay zekâ destekli değerlendirme süreci başlatılır.

Geri bildirimin içeriği, yazım doğruluğu ve anlam açısından değerlendirme barındırır. Bu sayede kullanıcı, hatasını yargılayıcı olmayan bir dille öğrenir.

Bu yapı sayesinde yazma becerileri sadece doğru/yanlış kontrolüyle sınırlı kalmaz; yapay zekâ desteğiyle dilbilgisi, anlam ve yapı açısından geri bildirim alınabilir hâle gelir.



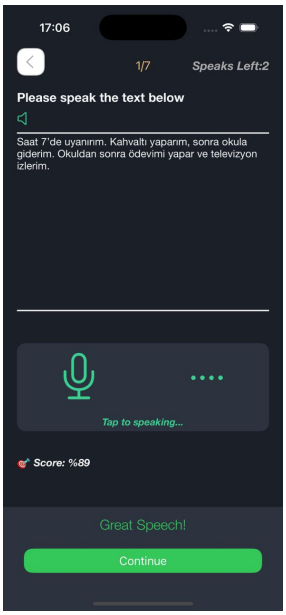
Görsel 8 - Writing Kursu Ekranı

2.3.8 Speaking Kursu Ekranı - SpeakingScreenViewController

SpeakingScreenViewController, kullanıcının sesli yanıtlar vererek konuşma pratiği yapmasını sağlayan ekrandır. Bu ekran, kullanıcıdan belirli bir Türkçe metni doğru telaffuzla seslendirmesini ister ve ardından yapay zekâ destekli bir sistem üzerinden geri bildirim sağlar.

Kullanıcı etkileşimi şu şekilde ilerler:

- Kullanıcı, ekranda gösterilen Türkçe cümleyi okur.
- Kullanıcının ses kaydı başlatıldıktan sonra kalan konuşma hakkı takip edilir.
- Ses kaydı tamamlandığında, kullanıcı checkButton ile sistemden değerlendirme alabilir.
- Sistem, stopRecordingAndEvaluate fonksiyonu ile konuşmayı analiz eder, benzerlik skorunu hesaplar ve bu oranı başarı yüzdesi olarak gösterir.
- Skor %50'nin üzerindeyse kullanıcı başarılı sayılır ve "Great Speech!" gibi olumlu bir geri bildirim alır.
- Altında kalırsa sistem kullanıcıya yapıcı geri dönüş verir.



Görsel 9 - Speaking Kursu Ekranı

3. BULGULAR VE TARTIŞMA

Bu çalışmada geliştirilen iOS tabanlı yapay zekâ destekli eğitim platformu, dil öğrenimini kişiselleştirmeyi, etkileşimi artırmayı ve kullanıcıya anlık geri bildirim sağlamayı hedeflemiştir. Uygulama, dört temel beceriyi (Reading, Listening, Writing, Speaking) kapsayan ekranlara sahiptir ve her biri kullanıcı merkezli bir yaklaşımla yapılandırılmıştır.

Uygulamanın teknik mimarisi MVVM-C (Model–View–ViewModel–Coordinator) desenine dayanarak, modülerlik, test edilebilirlik ve sürdürülebilirlik açısından güçlü bir yapı ortaya koymuştur. ViewController'lar yalnızca kullanıcı arayüzünden sorumlu tutulmuş; iş mantığı ViewModel'ler üzerinden yönetilmiş, yönlendirme ve sayfa geçişleri ise Coordinator yapısı ile gerçekleştirilmiştir. Bu yaklaşım, kod tekrarını azaltmış ve geliştiricilere kolaylık sağlamıştır.

Geliştirilen ekranlar, her beceriye özel olarak kullanıcı etkileşimini artıracak şekilde yapılandırılmıştır:

- **ReadingScreen** kullanıcının anlamlandırma becerilerini ölçerken, yanlış cevaplar sonrası yapay zekâ aracılığıyla yönlendirici geri bildirim sunmuştur.
- **ListeningScreen** kullanıcıya dinleme alıştırmaları yapma ve duyduğunu anlama üzerine yoğunlaşmış, sınırlandırılmış dinleme hakları ile dikkatli dinlemeye teşvik etmiştir.
- **WritingScreen** çeviri üzerinden üretim becerilerini ölçmüş ve doğru/yanlış ayrımı yerine açıklayıcı geri dönüşlerle kullanıcıyı yönlendirmiştir.
- **SpeakingScreen** ise konuşma kayıtları üzerinden benzerlik skoru hesaplayarak kullanıcıya özgü bir değerlendirme sistemi sunmuştur.

Uygulama içindeki görsel ve etkileşimli bileşenler (Lottie animasyonları, AI popup mesajları, progress bar'lar) kullanıcı deneyimini olumlu yönde etkilemiş, uygulamanın eğitici olduğu kadar motive edici bir yapıya kavuşmasını sağlamıştır.

Bu bulgular, klasik dil öğretim yöntemlerine göre daha bireyselleştirilmiş ve geri bildirime dayalı bir yaklaşımın, teknoloji desteğiyle uygulanabilirliğini ortaya koymuştur. Sistem, yalnızca bilgiyi sunmakla kalmamış; kullanıcıyı sürece aktif olarak dahil eden, değerlendiren ve yönlendiren bir öğrenme mekanizması kurmuştur.

4. SONUÇ

Bu çalışma kapsamında, Swift programlama diliyle geliştirilen ve OpenAI API entegrasyonu içeren yapay zekâ destekli bir mobil eğitim platformu başarılı bir şekilde tasarlanmış ve hayata geçirilmiştir. Kullanıcıların dört temel dil becerisi üzerinde çalışmasına olanak sağlayan uygulama, klasik eğitim uygulamalarından farklı olarak anlık geri bildirim, kişiselleştirme ve interaktif öğrenme deneyimi sunmuştur.

Kullanılan MVVM-C mimarisi sayesinde uygulama; okunabilir, genişletilebilir ve sürdürülebilir bir yapıya sahip olmuş, kodun farklı katmanlara ayrılması ile geliştirici açısından da yüksek yönetilebilirlik sağlanmıştır. Lottie animasyonları, özelleştirilmiş View bileşenleri ve kullanıcıdan alınan ses/metin girdilerinin değerlendirilmesi gibi unsurlar sayesinde etkileşim seviyesi artırılmıştır.

Yapay zekâ ile desteklenen bu yapı, öğrencilere yalnızca doğru cevabı göstermekle kalmayıp, onları neden-sonuç ilişkisi kurmaya yönlendirmiş, motivasyonlarını artırmış ve öğrenmeyi daha verimli hâle getirmiştir.

Sonuç olarak, geliştirilen bu platform; teknolojinin dil eğitimi süreçlerine başarılı bir şekilde entegre edilebileceğini ve kişiye özel eğitim modellerinin mobil cihazlar aracılığıyla erişilebilir kılınabileceğini göstermektedir. Bu yaklaşımın gelecekte farklı dillerde, daha geniş kullanıcı kitlesine hitap eden, daha gelişmiş yapay zekâ algoritmalarıyla desteklenen sürümleri geliştirilebilir.