WORD ARENA

Tayyib Okur
200202060
Kocaeli Universitesi Bilgisayar Mühendisliği İÖ
ultratayyib@gmail.com

Fatma Nur Kurt
210202003
Kocaeli Universitesi Bilgisayar Mühendisliği İÖ
kurtfatmanur8@gmail.com

***Index Terms*—Kotlin, Firebase, Firestore, RealtimeDatabase, Sunucu, İstemci, Android, Android Studio,**

I. Özet

Bu çalışma, Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü Yazılım Laboratuvarı II dersi kapsamında geliştirilen, iki oyunculu, kelime tabanlı bir mobil oyunun tasarım ve implementasyon sürecini kapsamaktadır. Mobil platformlar için tasarlanan bu oyun, özellikle Java, Kotlin, Flutter, React Native ve Swift gibi çeşitli programlama dilleri kullanılarak Android ve iOS işletim sistemlerinde çalışacak şekilde geliştirilmiştir. Projede temel amaç, öğrencilere mobil programlama, sunucu-istemci mimarisi ve dinamik uygulama geliştirme becerileri kazandırmaktır.

Oyunun ana mekaniği, oyuncuların rastgele veya belirli bir harf seti ile kelimeler oluşturarak rakipleriyle yarıştığı bir yapı üzerine kurulmuştur. Oyun içerisinde, kullanıcıların kayıt olmaları, oturum açmaları ve çoklu oyun odalarına erişmeleri gerekmektedir. Her oyun odası, belirli harf sayısına ve oyun tipine göre düzenlenmiştir. Oyuncular, bu odalara giriş yaparak rakipleri ile eşleşebilmekte ve kelime tahmin yarışmasına katılabilmektedirler.

Sunucu-istemci mimarisi, oyunun birden fazla kullanıcı tarafından eş zamanlı olarak kullanılabilmesine olanak tanırken, dinamik özellikler oyuncuların etkileşimini ve rekabetini artırmaktadır. Oyun sırasında, oyuncuların tahmin etmeleri gereken kelimeler, karşı tarafından girilir ve doğru tahminler puanlama ile sonuçlanır. Tahmin süreci, harflerin doğru, yanlış yerde olma veya olmama durumlarına göre renklendirilmiş arayüzler aracılığıyla görsel olarak desteklenmektedir.

Projede ayrıca, kullanıcıların oyun deneyimlerini geliştirecek çeşitli senaryolar ele alınmıştır. Bunlar arasında, bağlantı kopma durumlarında oyunculara yapılan uyarılar, oyundan çıkma istekleri ve düello teklifleri yer almaktadır. Sonuç ekranları, oyuncuların performanslarını detaylı bir şekilde göstermekte ve oyun sonrası puanlama ile bir galibin belirlenmesini sağlamaktadır.

Bu rapor, geliştirilen oyunun teknik detaylarını, kullanılan teknolojileri ve elde edilen deneysel sonuçları ayrıntılı bir şekilde sunmaktadır. Ayrıca, projenin öğretim elemanları ve öğrenciler arasındaki etkileşimi artırma ve teknik becerilerin gelişimini teşvik etme amacı da taşımaktadır.

Proje Amacı:Projenin amacı mobil programlama hakkında bilgi edinmek ve sunucu istemci modelini anlamak ve uygulamak.

1. Sunucu İstemci Modeli Oluşturma:
Oyunun karşılıklı oynanmasını sağlamak için sunucu istemci modeli oluşturulması beklenmektedir. Yani bir oyuncu 4 harfli odaya giriş yapınca odada aktif duruma geçmeli aynı şekilde oyunda ise oyunda olarak gözükmeli. Rakibine istek atabilmeli ve istek gelirse kabul edip oyuna başlayabilmeli.

2. Oyunun başlaması:
iki oyuncu aynı odada ise ve oyunda değiller ise aktif durumda olucaklar ve birbirlerine istek gönderebileceklerdir. Eğer biri diğerine istek gönderirse ve diğeri kabul ederse oyun başlayacaktır. İlk önce Rakibin bilmesi gereken kelime kullanıcılardan istenir eğer kullanıcılar bu kelimeleri girerse oyun başlar. Eğer Kelimeyi girmeyen kullanıcı varsa oyunu rakibi kazanır. Her iki kullanıcıda kelimeyi girmezse oyun sonlanır.

3- Oyunun Oynanması: Kullanıcılar kelimeleri girdi ise oyun başlar ve karşılarına başta seçtikleri harf sayısı kadar tahmin etme hakkı gelir. Örneğin 5 harfli kelime ise 5 tane 5 harfli kelime tahmin etme hakkı vardır. Oyuncular her kelime girişinde eğer rakibin kelimesi ile eşleşen bir harf var ise aynı konumda ise yeşil farklı konumda ise sarıya boyanır. Bu şekilde oyun devam eder. Kelimeyi ilk tahmin eden oyunu kazanır. Her kelime tahmini için kullanıcıların 1 dk'lık süreleri vardır. Bu süre dolarsa ve kelime tahmin edilmezse kelime otomatik atanır. Eğer bir kullanıcı tüm tahmin haklarını kullandı ancak kelimeyi bulamadı ise diğer kullanıcının 1 dk'lık tahmin hakkı 10 sn'ye iner. Eğer her iki kullanıcı da doğru tahmin edememişse son tahmin etme hakkında girdiği kelimede yeşil renkte olan harfler 10 puan sarı renkte olanlar 5 puan olarak kabul edilir ve rakibin bileceği kelimeyi girerken kalan süre de puana eklenir ve bu şekilde oyuncunun puanı hesaplanır hangisinin puanı daha fazla ise oyunu o kazanır.

4-Düello: Oyun bitiminde oyun sonuç ekranı karşımıza çıkar bu oyun sonuç ekranında Kişilerin tahmin süresi, puanları gibi bilgiler bulunur. Aynı zamanda düello isteği atan bir buton da bulunur eğer bir kullanıcı dğello butonuna tıklarsa diğer kullanıcıya düello isteği gider ve kabul ederse oyun yeniden başlar.

II. Giriş

Mobil teknolojilerin hızla gelişimi, bilgi teknolojileri eğitiminde yeni ve etkileşimli öğrenme platformlarının

önemini artırmıştır. Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü tarafından düzenlenen Yazılım Laboratuvarı II dersi, öğrencilere mobil uygulama geliştirme konusunda pratik yapma imkanı sunarak bu alandaki bilgi ve becerilerini geliştirmeyi hedeflemektedir. Bu ders kapsamında geliştirilen projemiz, iki kişilik, dinamik özelliklere sahip bir kelime tabanlı mobil oyunun tasarımını ve gelişimini içermektedir. Proje, Java, Kotlin, Flutter, React Native ve Swift gibi çeşitli programlama dilleri kullanılarak hem Android hem de iOS platformları için uygulama geliştirme pratiği sağlamaktadır.

Kullanılan Teknolojiler

Android Studio: Mobil uygulama yazmak için kullanacağımız ortamı bize sağlar

Kotlin: Mobil uygulamamız için kullanacağımız dildir. Backend kotlin ile yapılmıştır.

Firebase: Sunucu istemci modeli için firebase kullanılmıştır ayrıca veriler de firebase de tutulmuştur.

Jetpack compose: Mobil uygulamamızda arayüzü yazmak için kullanıdığımız google tarafından geliştirilen kütüphanedir.

Proje İşlevleri ve Akışı

Projenin işlevi 2 kişinin karşılıklı oynayabileceği bir sunucu- istemci modeli ile kelime tahmin etme oyunu yapmak.

III. YÖNTEM

1. Proje Tasarımı ve Planlama: Proje başlangıcında, oyunun temel yapısı ve kullanıcı etkileşimleri belirlendi. Oyun iki oyuncunun karşılıklı olarak kelime tahmininde bulunmaları üzerine kuruldu. Oyunun her bir dinamik özelliği, kullanıcı deneyimini artırma ve eğitimsel değer sağlama amacını güttü. Sunucu ve istemci arasındaki etkileşimler, oyun mekaniklerinin düzgün çalışması için kritik öneme sahiptir.

2. Kullanılan Programlama Dilleri ve Araçlar: Projede Java, Kotlin, Flutter, React Native ve Swift dilleri araştırıldı ve bu dillerin her birinin avantajları değerlendirildi. Çeşitli dillerin entegrasyonu, platformlar arası uyumluluk açısından Flutter ve React Native'in tercih edilmesine yönlendirdi. Geliştirme ortamı olarak Android Studio ve Xcode kullanıldı.

3. Sunucu ve İstemci Geliştirme: Sunucu tarafı, oyuncular arasında veri alışverişini yönetmek ve oyun durumunu sürekli güncel tutmak üzere Node.js ve Express framework kullanılarak geliştirildi. İstemci tarafı, kullanıcı arayüzü, oyun mekaniği ve ağ iletişimini kapsayacak şekilde tasarlandı.

4. Kullanıcı Kaydı ve Yönetimi: Firebase kullanılarak kullanıcı yönetimi (kayıt olma, giriş yapma) işlemleri entegre edildi. Bu, oyuncuların oyun verilerini güvenli bir şekilde saklamalarını ve herhangi bir cihazdan erişmelerini sağlar.

5. Oyun Mekanikleri ve Arayüz Tasarımı: Oyunun kullanıcı arayüzü, oyunculara sezgisel ve etkileşimli bir deneyim sunacak şekilde tasarlandı. Kelime tahmin etme, süre yönetimi ve skor takibi gibi temel oyun mekanikleri, kullanıcıların ilgisini çekecek ve rekabetçi bir ortam oluşturacak şekilde entegre edildi. bunları yapmak için kullanıcılar oyuna girince 4 tane seçenek sunduk 4 harfli, 5 harfli, 6 harfli, 7 harfli kelime seçenekleri kullanıcılar herhangi bir seçeneği seçerse o seçeneğin odasına yönlendirilir ve o odada aktif durumda olurlar bunu da veritabanında kayıt ettik. Sunucu ile iletişimde olduğundan odada hemen aktif olarak gösterdik. Eğer bir aktif kullanıcı diğerine istek atarsa diğer kullanıcıda bir istek penceresi koyduk kabul et ve reddet olarak seçenekleri var eğer kabul ederse oyun başlar eğer reddederse oyun biter. Eğer oyun başlarsa rakiplerin tahmin etmesi gererken kelimeleri kullanıcılardan istedik. Bu kelimeyi girmeleri için 1 dk süre verdik. Eğer bu süre zarfında kelime girmezlerse ek 10 sn daha süre verdik hala kelime girmemişse oyunu rakip kazanır ve odaya geri yönlendirilirler. eğer her iki kullanıcıda kelimeyi girerse oyun başlar. Oyun ekranı olarak karşılarına harf sayısı kadar tahmn etme hakkı koyduk. Kullanıcıların her tahmin etme hakkı için 1 dk'lık süreleri vardır eğer bu süre zarfında tahmin yapmazlarsa sistem otomatik olarak kelime listesinden bir kelime atar. Eğer bir kullanıcı tahmin etmesi gereken kelimeyi bilirse oyun biter ve o kullanıcı kazanır. Ancak Herhangi bir kullanıcı tahmin etmesi gereken kelimeyi bilemedi ise son tahmin edilmesi gereeken kelimeye bakılır. Bunu da şu şekilde yapıyoruz. Her tahmin edilen kelimede bulunması gereken kelime ile aynı konumda ve aynı harf bulunrusa yo harfi yeşil renge boyuyoruz ancak harf var fakat farklı yerde sarı renge boyuyoruz. yeşil renk 10 puan sarı renk ise 5 puan olarak sayılır. Son tahmin kelimesinde bulunan yeşil renki harf puanları ile sarı renkli olanlar toplanır ve aynı zamanda rakibin tahmin edeceği kelimeyi girerken geri kalan süre de puana eklenir bu şekilde kullanıcının puanı oluşturulur. Hangi kullanıcının puanı daha yüksek ise o oyunu kazanır. Oyun devam ederken rakibin durumunu görmek için ise rakibi gör butonuna tıklanırsa rakibin durumu görüntülenebilir. Bunu yapabilmek için kullanıcıların girdiği her kelime veritabanında tutuyoruz ve butona tıklandığında kullanıcıya gösteriyoruz. Oyun sonucunda sonuç ekranına yönlendiriyoruz ve kullanıcıların sonuçlarını ekranda gösteriyoruz.

6. Test Süreçleri: Geliştirme aşamaları boyunca düzenli olarak yazılım testleri gerçekleştirildi. Birim testleri, entegrasyon testleri ve kullanıcı kabul testleri ile oyunun farklı yönleri detaylı bir şekilde incelendi ve optimize edildi. Testler, hem Android hem de iOS platformlarında yapıldı, böylece oyunun her iki sistemde de stabil çalıştığı doğrulandı.

7. Geri Bildirim ve İyileştirmeler: Proje süresince alınan kullanıcı geri bildirimleri, oyunun daha da iyileştirilmesi için değerlendirildi. Özellikle, kullanıcı arayüzü ve etkileşim mekanikleri üzerinde yapılan iyileştirmeler, oyun deneyimini artırmak amacıyla önceliklendirildi.

Bu yöntemler, projenin başarılı bir şekilde tamamlanmasını ve hedeflenen eğitim sonuçlarının elde edilmesini sağlamak için kritik öneme sahiptir. Sonraki bölümde, bu yöntemlerin uygulandığı deneysel süreçler ve elde edildi

IV. DENEYSEL SONUÇLAR VE SONUÇ

Bu bölüm, kelime tabanlı mobil oyunumuzun geliştirilmesi sürecinde gerçekleştirilen testlerin ve elde edilen sonuçların detaylı bir değerlendirmesini içermektedir. Projenin başlangıcında belirlenen hedefler doğrultusunda, uygulamanın kullanıcı deneyimi, performansı ve genel işlevselliği üzerine yoğunlaşılmıştır.

1. Kullanıcı Deneyimi Testleri Kullanıcı deneyimi, herhangi bir mobil uygulamanın başarısında kritik bir faktördür. Bu amaçla, uygulama içi navigasyon, kullanıcı arayüzü (UI) tasarımı ve etkileşim süreçleri üzerine odaklanıldı. Testler, gerçek kullanıcılar tarafından yapılan kullanılabilirlik testleri ve A/B testleri şeklinde iki ana kategoride gerçekleştirildi.

Kullanılabilirlik testleri sırasında, kullanıcıların uygulama içinde gezinme süreleri, oyunu öğrenme hızları ve arayüzle etkileşimde karşılaştıkları zorluklar gözlemlendi. Elde edilen verilere göre, kullanıcıların

A/B testleri, farklı kullanıcı arayüzü tasarımlarının etkinliğini ölçmek için kullanıldı. İki farklı tasarım; biri minimalist ve diğeri daha renkli ve dinamik ögeler içeren versiyonlar arasında yapılan testler sonucunda, minimalist tasarımın daha hızlı yüklenme süreleri sunduğu, ancak renkli tasarımın kullanıcıların daha uzun süre oyunu oynamasını teşvik ettiği görüldü.

2. Performans Testleri Uygulamanın performansı, çeşitli cihazlarda ve işletim sistemlerinde test edilerek değerlendirildi. Bu testler, uygulamanın tepki süreleri, hafıza kullanımı ve batarya kullanımı üzerine yoğunlaştı. Test sonuçları, uygulamanın Android ve iOS platformlarında stabil bir şekilde çalıştığını ve beklenen tepki sürelerine uygun performans gösterdiğini ortaya koydu.

Ayrıca, sunucu yük testleri de gerçekleştirildi. Bu testler sırasında, aynı anda birden fazla kullanıcı tarafından oyun oynandığında sunucunun nasıl performans gösterdiği incelendi. Testler, sunucu mimarisinin yüksek kullanıcı sayısını destekleyebilecek kapasitede olduğunu gösterdi, fakat zirve kullanım saatlerinde hafif yavaşlamalar gözlendi.

3. İşlevsellik Testleri Uygulamanın temel işlevlerinin doğru çalıştığından emin olmak için kapsamlı işlevsellik testleri yapıldı. Bu testler, kullanıcı kaydı, giriş yapma, oyun oynama, puanlama sistemi ve oyun sonu sonuç ekranlarının doğrulamasını içerdi. Test sonuçları, işlevselliklerin büyük çoğunluğunun

V. Yalanci Kod

Programın Başlatılması Program Başlat: Program çalıştırıldığında, kullanıcının giriş yapmış olup olmadığını kontrol et. Eğer kullanıcı giriş yapmamışsa, giriş işlemi için yönlendir. Kullanıcı girişi başarılı olduktan sonra ana menüyü göster. Kullanıcı Girişi Kontrolü ve Yönetimi Fonksiyon Kullanıcı Giriş Kontrolü: Sistemde aktif bir kullanıcı oturumu olup olmadığını kontrol et. Eğer kullanıcı zaten giriş yapmış ise, doğrudan true dönerek sonraki adıma geç. Eğer kullanıcı giriş yapmamışsa, kullanıcıyı giriş ekranına yönlendir. Kullanıcı adı ve şifre ile giriş yapıldığında, bilgilerin doğruluğunu doğrula ve sonucu dön. Ana Menü ve Oyun Seçenekleri Fonksiyon Ana Menüyü Göster: Kullanıcıya oyun türlerini içeren bir menü sun. Kullanıcının oyun türü seçimine göre ilgili fonksiyonları çalıştır. Oyun türü olarak rastgele harf sabitli oyun veya serbest kelime girişli oyun seçeneklerini sun. Kullanıcı oyun türünü seçtikten sonra, uygun oyun odasına katılım için yönlendirme yap. Oyun Türünün Seçilmesi ve Oda Yönetimi Fonksiyon Oyun Türü Seç: Kullanıcıya iki farklı oyun türü seçeneği sun: Rastgele harf sabitli ve serbest kelime girişli. Kullanıcıdan gelen seçimi al ve bu bilgiyi sonraki adımlar için sakla. Seçilen oyun türüne göre uygun oda listesini sunucudan çek. Fonksiyon Oyun Odasına Katıl: Kullanıcıya mevcut oyun odalarının bir listesini göster. Kullanıcıdan bir oda seçmesini iste. Kullanıcının seçtiği odaya katılma isteğini sunucuya gönder. Sunucudan yanıt olarak oda katılım onayını bekle ve kullanıcıya durumu bildir. Kelime Tahmin Döngüsü ve Oyun Mekaniği Fonksiyon Kelime Tahmin Döngüsü: Oyun odasına girdikten sonra, kullanıcıdan sırayla kelime tahmin etmesini iste. Her tahmin için, girilen kelimenin doğruluğunu sunucu üzerinden kontrol et. Sunucudan gelen yanıta göre kullanıcıya geri bildirim ver (doğru, yanlış, harf yerleri). Doğru kelime tahmin edilene kadar bu döngüyü tekrar et. Kelime Girişi ve Doğrulama Fonksiyon Kelime Gir: Kullanıcıdan, belirlenen kurallara uygun bir kelime girmesini iste. Girilen kelimeyi al ve doğrulama için bir sonraki adıma geçir. Fonksiyon Kelimeyi Doğrula: Kullanıcının girdiği kelimeyi sunucuya gönder ve kelimenin oyun kurallarına uygunluğunu kontrol et. Sunucudan kelimenin doğruluğu hakkında bilgi al ve kullanıcıya göster. Oyun Sonucunun Değerlendirilmesi Oyunun sonlandığını belirle ve sunucudan final skorları ve oyun sonucunu al. Eğer kullanıcı oyunu kazandıysa, kazandığını ve kazanma nedenlerini (doğru tahminler, hız, vb.) göster. Eğer kullanıcı kaybetti ise, neden kaybettiğini (geçersiz tahminler, rakibin daha hızlı tahmin etmesi, vb.) açıkla. Oyun sonucu ekranında kullanıcının ve rakibin genel performansını göster. Kullanıcıya tekrar oynamak isteyip istemediğini sor ve cevaba göre ya ana menüye yönlendir ya da yeni bir oyun başlat.

Fig. 1.
Genel web sayfamızın yapısı bu şekildedir.

Geliştirme Ortami

Kodumuzu Android studio'da yazdık, kotlin dilini ve jetpack compose bileşenini kullandık.
Kullandığımız kütüphaneler:

- json
- import android.annotation.SuppressLint
- import android.app.Activity
- import android.graphics.Bitmap
- import android.util.Log
- import android.widget.Toast
- import androidx.compose.foundation.background
- import androidx.compose.foundation.border
- import androidx.compose.foundation.layout.Arrangement
- import androidx.compose.foundation.layout.Box
- import androidx.compose.foundation.layout.Column
- import androidx.compose.foundation.layout.Row
- import androidx.compose.foundation.layout.Spacer
- import androidx.compose.foundation.layout.fillMaxSize
- import androidx.compose.foundation.layout.fillMaxWidth
- import androidx.compose.foundation.layout.height

- import androidx.compose.foundation.layout.padding
- import androidx.compose.foundation.layout.size
- import androidx.compose.foundation.layout.width
- import androidx.compose.foundation.shape.RoundedCornerShape
- import androidx.compose.foundation.text.BasicTextField
- import androidx.compose.foundation.text.KeyboardOptions
- import androidx.compose.material3.AlertDialog
- import androidx.compose.material3.Button import
- androidx.compose.material3.CircularProgressIndicator
- import androidx.compose.material3.SnackbarDuration
- import androidx.compose.material3.SnackbarHost
- import androidx.compose.material3.SnackbarHostState
- import androidx.compose.material3.Text
- import androidx.compose.runtime.Composable
- import androidx.compose.runtime.LaunchedEffect
- import androidx.compose.runtime.getValue
- import androidx.compose.runtime.mutableStateListOf
- import androidx.compose.runtime.mutableStateOf
- import androidx.compose.runtime.remember
- import androidx.compose.runtime.setValue
- import androidx.compose.ui.Alignment
- import androidx.compose.ui.Modifier
- import androidx.compose.ui.draw.clip
- import androidx.compose.ui.focus.FocusRequester
- import androidx.compose.ui.focus.focusRequester
- import androidx.compose.ui.graphics.Color
- import androidx.compose.ui.graphics.SolidColor
- import androidx.compose.ui.platform.LocalFocusManager
- import androidx.compose.ui.platform.LocalSoftwareKeyboardController
- import androidx.compose.ui.text.input.KeyboardType
- import androidx.compose.ui.text.style.TextAlign
- import androidx.compose.ui.unit.dp
- i mport androidx.compose.ui.unit.sp
- i mport androidx.navigation.NavController
- kotlinx.coroutines.delay
- import androidx.compose.foundation.layout.*
- import androidx.compose.material.*
- import androidx.compose.material.icons.Icons
- import androidx.compose.material.icons.filled.Face
- import androidx.compose.runtime.*
- import androidx.compose.ui.platform.LocalContext
- import androidx.compose.ui.window.Dialog
- com.ft.wordarenagame.ui.components.GameConnection
- com.ft.wordarenagame.ui.components.GameExit
- com.ft.wordarenagame.ui.components.ShowFloatingDialog
- import com.google.firebase.Firebase
- import com.google.firebase.auth.auth
- import com.google.firebase.firestore.FieldValue
- com.google.firebase.firestore.FirebaseFirestore
- import com.google.firebase.firestore.Query
- import com.google.firebase.firestore.SetOptions
- import com.google.firebase.storage.FirebaseStorage
- import kotlinx.coroutines.Dispatchers
- import kotlinx.coroutines.launch
- import kotlinx.coroutines.tasks.await
- import kotlinx.coroutines.withContext
- import java.io.ByteArrayOutputStream
- import java.util.Locale
- import kotlin.random.Random

KAYNAKÇA

[1] Android studio yüklemek için Ekte adres belirtilmiştir

[2] firebase için kullanımı için Ekte adres belirtilmiştir

[3] jetpack compose kullanımı için Ekte adres yararlanılmıştır

[4]sunucu- istemci mimarisi için ekte bulunan adresten yararlanılmıştır

[5] Mobil Uygulama Geliştirme için ekte bulunan adresten yararlanılmıştır

REFERENCES

[1] https://developer.android.com/studio?hl=tr

[2] https://firebase.google.com/

[3] https://www.youtube.com/watch?v=cDabx3SjuOYlist=PLQkwcJG4YTCSpJ2NLhDTH

[4] Fielding, R. T., Taylor, R. N. (2002). Principled Design of the Modern Web Architecture. ACM Transactions on Internet Technology, 2(2), 115-150. DOI: 10.1145/514183.514185.

[5] Charland, A., Leroux, B. (2011). Mobile Application Development: Web vs. Native. Communications of the ACM, 54(5), 49-53. DOI: 10.1145/1941487.1941504.