
# T.C

# SAKARYA ÜNİVERSİTESİ

# BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

MOBİL UYGULAMA GELİŞTİRME DERSİ ÖDEV RAPORU

QUİZAPP UYGULAMASI

Bağlantı Adresi: https://drive.google.com/file/d/1wSPN1xRicLE-TvKMF-Kz-QWOzSXkFxgy/view?usp=sharing

Grup Üyeleri:

**B181210068 – Betül Nur Güner (1A)**

**B181210026 – Gülinsu Özturan (1A)**

**SAKARYA**

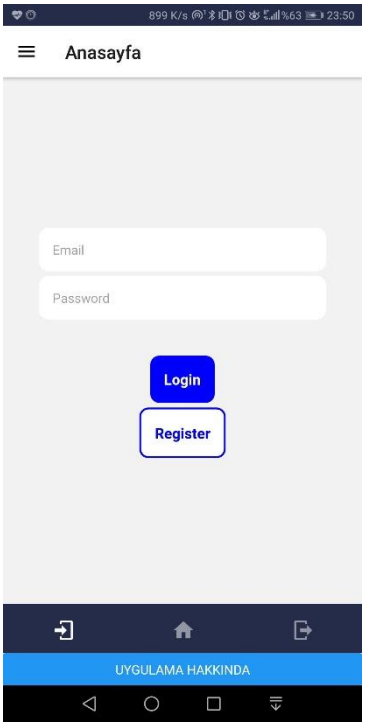
**Aralık, 2021**

# İçindekiler

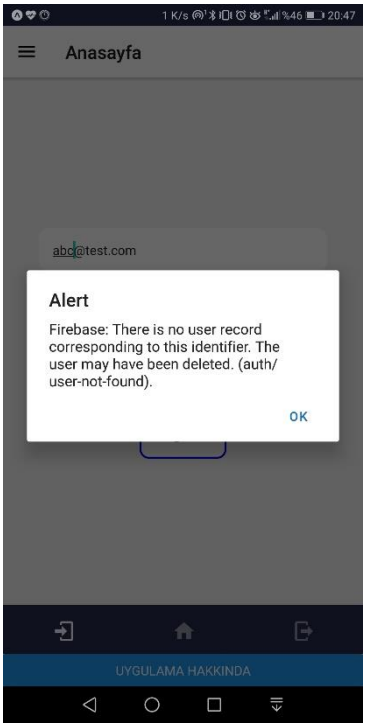
# PROJENİN TANITIMI

## 1-Login Ekranı

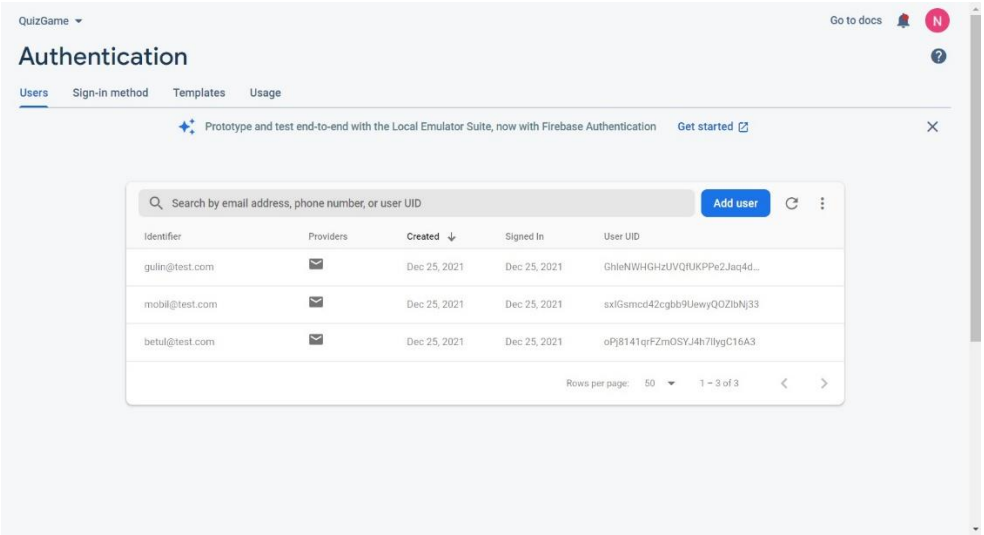
Projemizde ilk olarak kullanıcıyı bir login ekranı karşılamaktadır. Login ve doğrulama işlemleri için Firebase teknolojisinden faydalandık. Kullanıcı daha önceden kayıtlı değilse email ve şifre bilgisini girerek “Register” butonuna tıklamalıdır, böylece hem kolay şekilde kaydolacak hem de otomatik olarak Quiz ekranına yönlendirilecektir. Eğer kullanıcı daha önceden kayıtlı ve firebase veritabanında bilgileri bulunuyorsa bu bilgileri girerek “Login” butonuna tıklaması yeterlidir. Eğer kullanıcı kayıtlı olmadığı halde login olmaya çalışırsa aşağıdaki uyarı yazısıyla karşılaşır:




*Şekil1*




*Şekil2*




*Şekil3*

Şekil2' deki ekran görüntüsünde abc@test.com kullanıcısı Şekil3'te ekran görüntüsü bulunan veri tabanımızda daha önce kayıtlı olmadığı için Login olmasına izin verilmez Register olması gerektiği uyarısını verir. Register işlemi tamamlanan kullanıcılar veritabanında görüntülenir.

Firebase konfigürasyon ayarları aşağıdaki şekilde yapılmıştır:

```
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyBNFKQ3KltVFPN3kh9BZLshs17brZpGav8",
  authDomain: "quizgame-fc683.firebaseapp.com",
  projectId: "quizgame-fc683",
  storageBucket: "quizgame-fc683.appspot.com",
  messagingSenderId: "30544961468",
  appId: "1:30544961468:web:e828db36324b9c6c2f55b9"
};

// Initialize Firebase
let app;
if(firebase.apps.length ===0){
    app=firebase.initializeApp(firebaseConfig);
}
else{
    app=firebase.app()
}

const auth=firebase.auth()
export {auth};
```

*Şekil4*

Giriş ve Kayıt işlemleri için screen klasörü altında LoginScreen.js dosyası oluşturuldu. Firebase ile authentication işlemlerini gerçekleştirmek için import { auth } from '../firebase' kütüphanesi eklenir. Bu dosyanın içinde import ettiğimiz auth bileşeni kullanılarak aşağıdaki fonksiyonlar yardımıyla kullanıcı kayıt ve giriş denetimi sağlanır:

```
const handleSignUp = () =>{
    auth
        .createUserWithEmailAndPassword(email,password)
        .then(userCredentails => {
            const user=userCredentails.user;
            console.log('Registered with: ',user.email);
        })
        .catch(error=>alert(error.message))
}

const handleLogin = () =>{
    auth
        .signInWithEmailAndPassword(email,password)
        .then(userCredentails => {
            const user=userCredentails.user;
            console.log('Login with: ',user.email);
        })
        .catch(error=>alert(error.message))
}
```

*Şekil5*

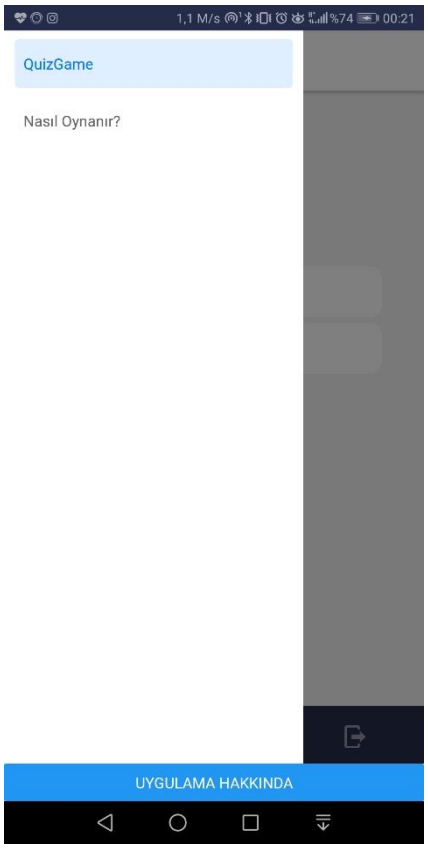
Kullanıcı giriş yaptıktan sonra quiz sorularının görüntülendiği HomeScreen dosyasına yönlendirilme yapılması gerekmektedir. Navigation.navigate() fonksiyonu kullanarak bu yönlendirme işlemini gerçekleştirdik.

```
const navigation=useNavigation()

useEffect(() => {
    const unsubscribe = auth.onAuthStateChanged(user=>{
        if(user){
            navigation.navigate("Home")

        }
    })

    return unsubscribe
}, [])
```

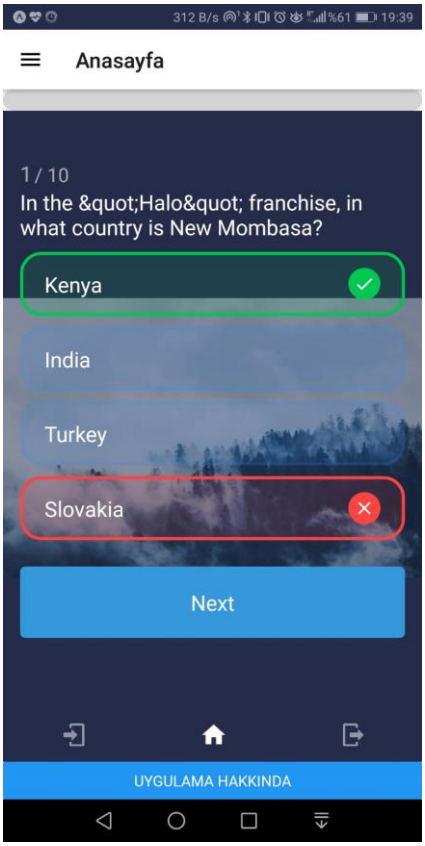
*Şekil6*

## 2- QuizGame

Solda kullandığımız drawer menümüzde Anasayfa olarak QuizGame menüsü ayarlanmıştır. Bu sayfanın içinde quiz soruları ve 3 adet tab bar yönlendirmesi mevcuttur. Tab iconlarından biri seçildiğinde hangisinin seçildiğini belirtmek amacıyla parlak renkte olmaktadır. En soldaki simge login ekranına, ortadaki home simgesi quiz soruları ekranına, en sağdaki simge logout ekranına kullanıcıyı yönlendirmektedir.




*Şekil7*

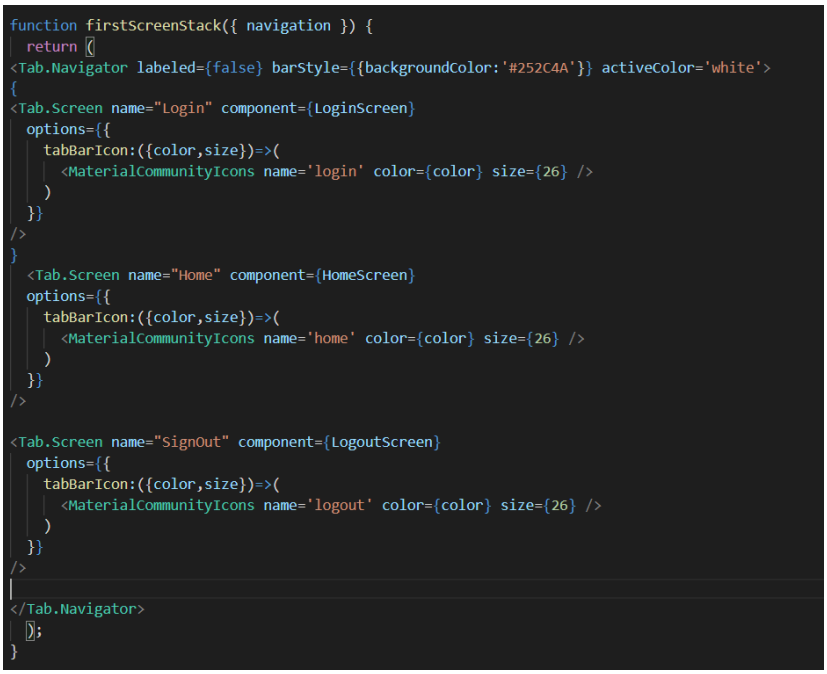



*Şekil8*

```
<NavigationContainer>

<Drawer.Navigator
  screenOptions={{
    activeTintColor: '#e91e63',
    itemStyle: { marginVertical: 5 },

  }}>
  <Drawer.Screen
    name="Anasayfa"
    options={{ drawerLabel: 'QuizGame' }}
    component={firstScreenStack}
    //component={HomeScreen}

    />
  <Drawer.Screen
    name="Nasıl Oynanır"
    options={{ drawerLabel: 'Nasıl Oynanır?' }}
    component={secondScreenStack}
    />

</Drawer.Navigator>
<Button
        title="Uygulama Hakkında"
        onPress={async () => {
          await sendPushNotification(expoPushToken);
        }}
      />
</NavigationContainer>
```

*Şekil8*

Şekil7’ deki drawer menüsünün oluşması için NavigationContainer Bileşeni içinde iki tane Drawer.Screen Bileşeni kullanılmıştır. Menüde QuizGame labeli ile görüntülenen bileşenin Anasayfaya yönlendirilmesi için firstScreenStack isimli bir fonksiyon yazılmış component olarak firstScreenStack verilmiştir:

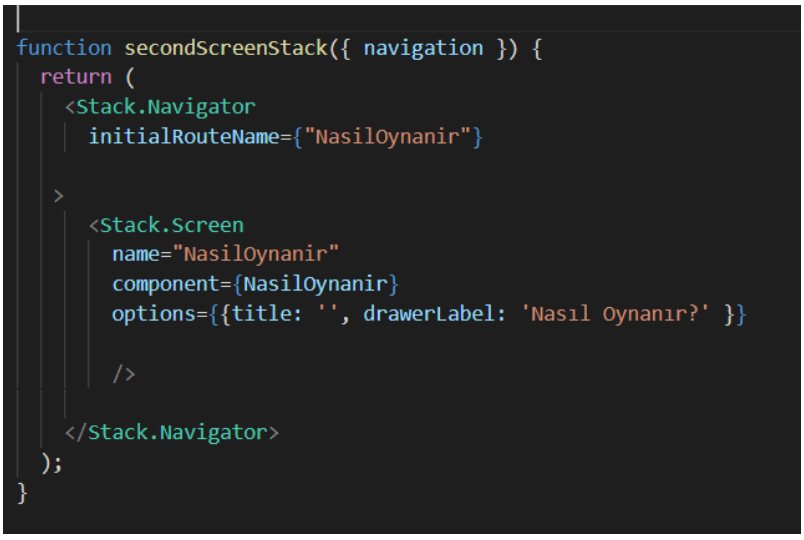



*Şekil9*

FirstScreenStack fonksiyonu içinde Tab Drawer için gereken Tab.Screen bileşenleri Tab.Navigator içerisinde yazılmıştır. Tab.Screen olarak Login, Home ve SignOut sayfaları oluşturulmuş olup screenlere gerçekleştirilen yönlendirmeler aşağıdaki gibidir:

Login -> LoginScreen Home -> HomeScreen SignOut -> LogoutScreen

Drawer menüsünde Nasıl Oynanır labeli ile görüntülenen bileşenin NasılOynanır sayfasına yönlendirilmesi için secondScreenStack isimli bir fonksiyon yazılmış component olarak secondScreenStack verilmiştir:




*Şekil9*

SecondScreenStack fonksiyonunda Nasıl Oynanır sayfasına yönlenme yapmak için Stack Navigation kullandık. Stack.Navigator bileşeni içerisinde Stack.Screen Bileşeni oluşturarak Nasıl Oynanır Sayfasına yönlendirdik.

HomeScreen.js sayfasında quiz uygulamamız için gereken sorular, cevapları ve bunların görüntü ayarları yer almaktadır. Uygulamamız içindeki soruları oluşturmak için aşağıdaki ekran görüntüsünde questionURL değişkenine atanan api kullanıldı.

```
const questionsURL= `https://opentdb.com/api.php?amount=10&type=multiple`;

export class QuizProps {
    static PropTypes={
        correct_answer: PropTypes.string,
        incorrect_answer: PropTypes.string,
        question: PropTypes.string,
        answers: PropTypes.string,
    }

};
```

*Şekil10*

{"response_code":0,"results":[{"category":"Science & Nature","type":"multiple","difficulty":"easy","question":"What is the hottest planet in the Solar System?","correct_answer":"Venus","incorrect_answers":["Mars","Mercury","Jupiter"]},{"category":"Geography","type":"multiple","difficulty":"medium","question":"What is the capital of Australia?","correct_answer":"Canberra","incorrect_answers":["Sydney","Melbourne","Brisbane"]},{"category":"Vehicles","type":"multiple","difficulty":"easy","question":"Which of the following car manufacturers had a war named after it?","correct_answer":"Toyota","incorrect_answers":["Honda","Ford","Volkswagen"]},{"category":"Science & Nature","type":"multiple","difficulty":"hard","question":"Which of the following is NOT a real element?","correct_answer":"Vitrainium","incorrect_answers":["Praseodymium","Hassium","Lutetium"]},{"category":"Entertainment: Video Games","type":"multiple","difficulty":"easy","question":"In the videogame Bully, what is the protagonist's last name?","correct_answer":"Hopkins","incorrect_answers":["Smith","Kowalski","Crabblesnitch"]},{"category":"General Knowledge","type":"multiple","difficulty":"hard","question":"Which of the following languages does NOT use gender as a part of its grammar?","correct_answer":"Turkish","incorrect_answers":["German","Danish","Polish"]},{"category":"Entertainment: Video Games","type":"multiple","difficulty":"medium","question":"Who voices the infamous Citadel Station A.I known as S.H.O.D.A.N, in the System Shock games?","correct_answer":"Terri Brosius","incorrect_answers":[" Jennifer Hale","Jenn Taylor","Lori Alan"]},{"category":"Science & Nature","type":"multiple","difficulty":"hard","question":"What element on the periodic table has 92 electrons?","correct_answer":"Uranium","incorrect_answers":["Sulfur","Hydrogen","Iron"]},{"category":"History","type":"multiple","difficulty":"easy","question":"What does the United States of America celebrate during the 4th of July?","correct_answer":"The signing of the Declaration of Independence","incorrect_answers":["The anniversary of the Battle of Gettysburg","The crossing of the Delaware River","The ratification of the Constitution"]},{"category":"Entertainment: Video Games","type":"multiple","difficulty":"hard","question":"In the Nintendo Game Splatoon 2, what is Marina's screen name?","correct_answer":"DJ_Hyperfresh","incorrect_answers":["MC.princess","Kidnotsquid123","I<3OffTh3H00k"]}]}

*Şekil11*

questionURL içindeki link üzerinden yukarıdaki json formatıyla yazılmış apiye ulaşılmaktadır.

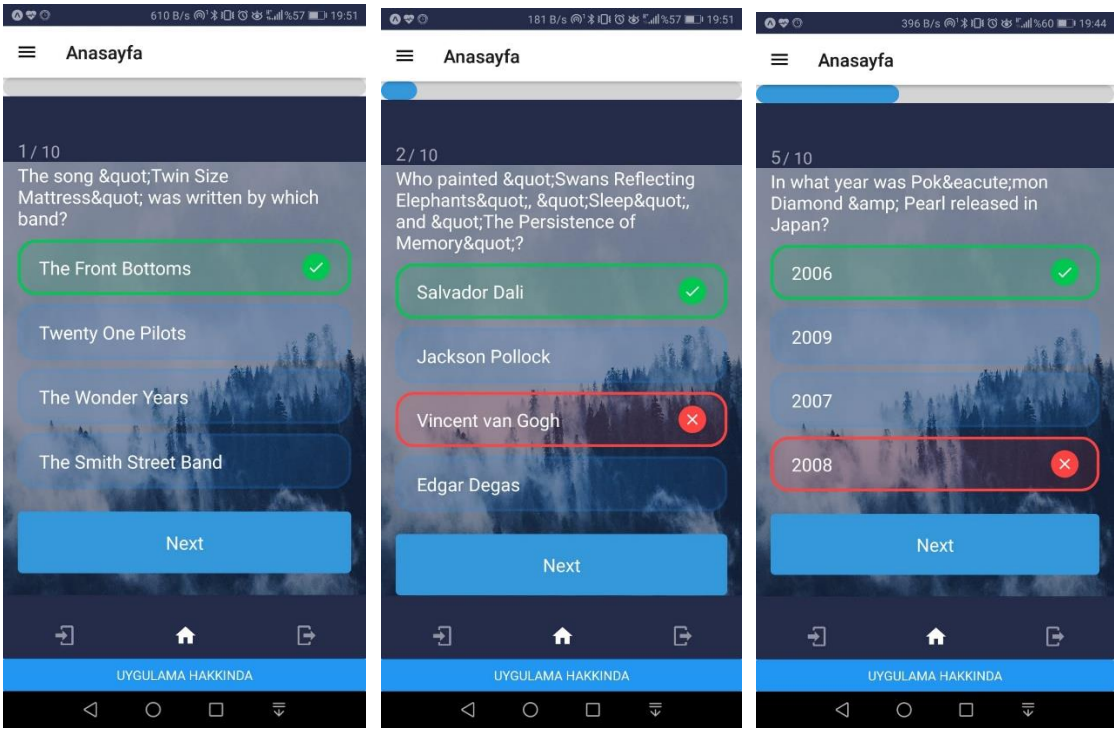
```
export type QuizPropsState = QuizProps & {answers : PropTypes.array};


export const catchTheQuestions = async () :Promise<QuizPropsState> => {
    const data = await (await fetch(questionsURL)).json();

    return data.results.map((quizprops: QuizProps) => ({
        ...quizprops,
        answers: ([quizprops.correct_answer, ...quizprops.incorrect_answers,])
    }))

    //console.log(data);  //kontrol için
}
```

*Şekil12*

JSON formatlı Api den soruları, cevapları, doğru ve yanlış cevapları ayrı ayrı çekmek için fetch() fonksiyonu json() fonksiyonu ile birlikte çağrılır. Şekil10'da apide bulunan question,

correct_answer, incorrect_answer sütunlarını uygulamada da kullanmak için QuizProps sınıfı oluşturmuştuk. Şekil12’ de apiden çekilen data verisini map() fonksiyonuyla düzenliyoruz ve ekranda doğru ve yanlış cevapları tek bir değişken kullanılarak yazdırmak için answers dizinini oluşturup elemanları olarak correct_answer ve incorrect_answer özelliklerini veriyoruz.

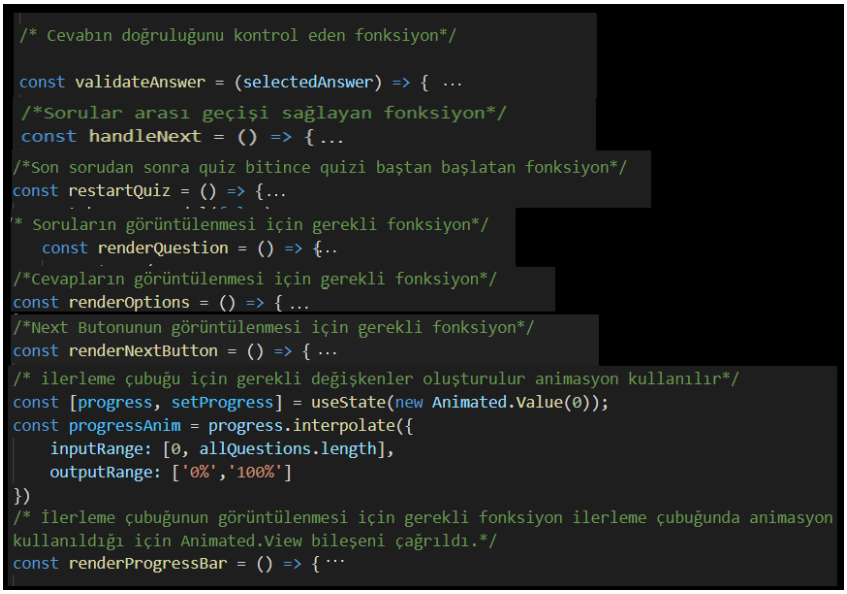
Answers dizinine özellikleri atarken başta bir takım sorunlarla karşılaşmamızın sonucunda doğru ve yanlış cevapların karıştırılması yapılamamış olup uygulamamızda soruların hepsinin doğru cevabı ilk seçenektir.




```
const HomeScreen = () => {

    const [isLoading, setLoading] = useState(false);            // api yüklendi mi
    const [allQuestions, setAllQuestions] = useState([]);       // soruları apiden alıp atama
    const [selectedAnswer,setSelectedAnswer] = useState([]);    // kullanıcının seçtiği yanıtı tutma
    const [score,setScore] = useState(0);                       // kullanıcının skorunu tutma
    const [currentNumber,setCurrentNumber] =useState(0);        // kullanıcının o anda bulunduğu soru indexini tutma
    const [quizOver,setQuizOver] = useState(true);              // quiz sonlandı mı kontrolü
    const [totalQuestions] = useState(0);
    const [currentOptionSelected, setCurrentOptionSelected] = useState(null);   // cevabın doğruluğunu kontrol ederken kullanmak için
    const [correctOption, setCorrectOption] = useState(null);                   // doğru cevabı atama
    const [isOptionsDisabled, setIsOptionsDisabled] = useState(false);          // seçenek seçilmemiş mi kontrolü
    const [showNextButton, setShowNextButton] = useState(false)                 // bir sonraki soruya geçişlerde butonun gözükmesini sağla
    const [showScoreModal, setShowScoreModal] = useState(false)                 // skoru görüntülemek için


    /* quiz uygulamasının başlamasını sağlar*/
    const start = async() => {
        setLoading(true);       // yükleniyor
        setQuizOver(false);     // quiz bitmedi
        const newQuestions = await catchTheQuestions(totalQuestions);   // soruları yakalayıp atama
        setAllQuestions(newQuestions);      // yeni oluşturulan soruları atama
        setScore(0);                        // başlangıçta skor 0
        setSelectedAnswer([]);              // seçilen cevabı dizi içinde tutar
        setLoading(false);                  // yüklemeyi bitirir.

    }

    useEffect(() => {
        start();            // start fonksiyonu çağrılarak quiz başlatılır.
    }, []);
```

*Şekil13*

Apiden gerekli kontroller yapılıp ilgili durum atamaları gerçekleştirildi. Ardından Quiz'in başlaması için start() fonksiyonu yazılıp useEffect içinde çağrıldı.




*Şekil14*

HomeScreen.js dosyası içinde Şekil14'teki fonksiyonlar yazılarak görüntüleme işlemleri gerçekleştirilmiştir.


*Şekil15*

Home Ekranındayken Logout tab seçeneğiyle Logout ekranına gelindiğinde kullanıcıya "Hesabınızdan çıkmak istediğinize emin misiniz?" sorusu yöneltilir. Evet seçeneği seçilirse çıkış yapılarak login ekranına, hayır seçeneği seçilirse Home ekranına yani quiz sorularına yönlendirme yapılır. Login ekranına yönlendirme yapılabilmesi için yazılan gerekli kod Şekil15'te, Sorulara geri dönmek için Home ekranına yönlendirme yapan kod Şekil17'de verilmiştir.


*Şekil16*

```
const goHomeScreen = () => {
    navigation.navigate("Home")
}
```

*Şekil17*

## 3-Nasıl Oynanır?

İkinci menü seçeneği “Nasıl Oynanır?”, burada kullanıcıya QuizGame tanıtılarak oyunu nasıl sonuçlandıracağı ve süreçte karşılaşabileceği durumlar uygulamamızın ekran görüntüleriyle anlatılmıştır.

Ekran Görüntülerini ScrollView componenti içinde gösterdik. ScrollView kaydırılabilir içerikler kullanabileceğimiz en basit componenttir. Görüntülenecek verinin çok fazla olmadığı durumlarda ScrollView kullanmak oldukça kullanışlıdır. Hem yatay hem de dikey olarak içerikleri kaydırmamızı sağlar. Biz Nasıl Oynanır ekranında ekran görüntülerini dikey kaydırma ile görüntüledik:

```
const NasilOynanir = () => {

    const {width,height} = Dimensions.get('window')

        return (

 <SafeAreaView>
                <ScrollView>
                <Text>Home sembolünün üzerine tıklayın.</Text>
                <Text>Doğru olduğunu düşündüğünüz seçeneğin üzerine tıklayın.</Text>
                <Image source={require('../assets/1.jpeg')} style={{width:width,height:height,resizeMode:'contain'}} />
                <Text>Seçtiğiniz seçenek doğru ise yeşil renk alacaktır.</Text>
                <Image source={require('../assets/2.jpeg')} style={{width:width,height:height,resizeMode:'contain'}} />
                <Text>Seçtiğiniz seçenek yanlış ise kırmızı renk alacaktır.</Text>
                <Image source={require('../assets/3.jpeg')} style={{width:width,height:height,resizeMode:'contain'}} />
                <Text>Her test 10 sorudan oluşur.</Text>
                <Text>İlerleme seviyenizi yukarıdaki bardan görebilirsiniz.</Text>
                <Image source={require('../assets/4.jpeg')} style={{width:width,height:height,resizeMode:'contain'}} />
                <Text>Test sonucunda kaç doğru cevap verdiğinizi görürsünüz.</Text>
                <Image source={require('../assets/5.jpeg')} style={{width:width,height:height,resizeMode:'contain'}} />
                </ScrollView>
        </SafeAreaView>


        )

}
export default NasilOynanir
```

*Şekil18*

## 4-Push Notifications

Push notification, akıllı telefonlarda bildirim çubuğuna local ya da remote olarak bildirim göndermemizi sağlayan sistemlerin genel adıdır.

Push notification için android tarafında Expo ve Firebase CloudMessaging, IOS tarafında Apple Push Notification Service kullanılabilir. Bizim geliştirdiğimiz projede android için geliştirdiğimizden Expo Notification kullandık. Projemizin apk'sını oluşturduğumuzda push notification'ın çalışmadığını gözlemledik. Araştırmalarımız sonucu Firebase CloudMessaging ile ilgili konfigürasyonlar yapmamız gerektiğini öğrendik.

```
export default function App() {

  const [expoPushToken, setExpoPushToken] = useState('');     // expo'dan notification için token atanır
  const [notification, setNotification] = useState(false);
  const notificationListener = useRef();
  const responseListener = useRef();

  useEffect(() => {
  registerForPushNotificationsAsync().then(token => setExpoPushToken(token));

  // This listener is fired whenever a notification is received while the app is foregrounded
  notificationListener.current = Notifications.addNotificationReceivedListener(notification => {
    setNotification(notification);
  });

  // This listener is fired whenever a user taps on or interacts with a notification (works when app is foregr
  responseListener.current = Notifications.addNotificationResponseReceivedListener(response => {
    console.log(response);
  });

  return () => {
    Notifications.removeNotificationSubscription(notificationListener.current);
    Notifications.removeNotificationSubscription(responseListener.current);
  };
}, []);
```

*Şekil19 expo notification için gerekli kodları app.js içine yazdık*

```
<Button
        title="Uygulama Hakkında"
        onPress={async () => {
          await sendPushNotification(expoPushToken);
        }}
      />
```

*Şekil20: Bildirim gönderimi için buton oluşturduk*

```
// Bildirim Gönderme Fonksiyonu gönderilecek bildirim burda ayarlanıyor.
async function sendPushNotification(expoPushToken) {
  const message = {
    to: expoPushToken,
    sound: 'default',
    title: 'Hoşgeldiniz!',
    body: 'QuizGame Uygulamasına Hoşgeldiniz! Uygulama hakkında bilgi edinmek
    data: { someData: 'goes here' },

  };
```

*Şekil21*

```
await fetch(

  'https://exp.host/--/api/v2/push/send', {
  method: 'POST',
  headers: {
    Accept: 'application/json',
    'Accept-encoding': 'gzip, deflate',
    'Content-Type': 'application/json',
  },
  body: JSON.stringify(message),


} );
```

*Şekil22*

Push Notification'ın gerçekleşmesi için gereken en önemli kod Şekil22'de verilmiştir. Bu kodda expo apisinden post metoduyla yukarda oluşturulan mesaj gönderilir.

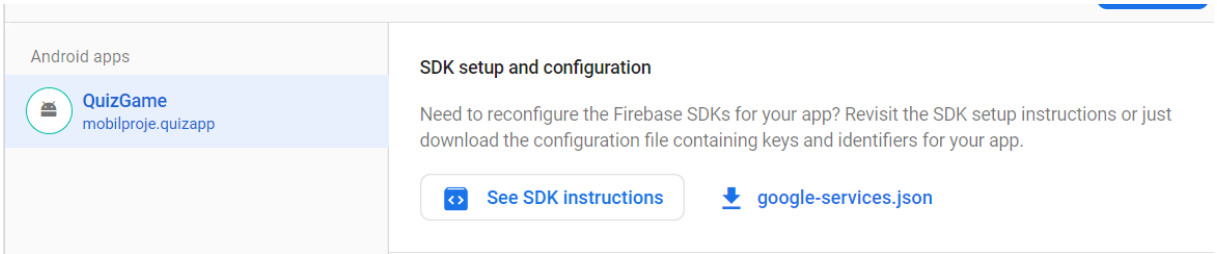
```
async function registerForPushNotificationsAsync() {
  let token;
  if (Constants.isDevice) {
    const { status: existingStatus } = await Notifications.getPermissionsAsync();
    let finalStatus = existingStatus;
    if (existingStatus !== 'granted') {
      const { status } = await Notifications.requestPermissionsAsync();
      finalStatus = status;
    }
    if (finalStatus !== 'granted') {
      alert('Failed to get push token for push notification!');
      return;
    }
    token = (await Notifications.getExpoPushTokenAsync()).data;
    console.log(token);
  } else {
    alert('Must use physical device for Push Notifications');
  }

  if (Platform.OS === 'android') {
    Notifications.setNotificationChannelAsync('default', {
      name: 'default',
      importance: Notifications.AndroidImportance.MAX,
      vibrationPattern: [0, 250, 250, 250],
      lightColor: '#FF231F7C',
    });
  }

  return token;
}
```

*Şekil23*

Push notification'ın çalışması için expodan bir token alınması gerekiyor. Bu token alınmadığı zaman push notification asla gerçekleştirilemiyor.

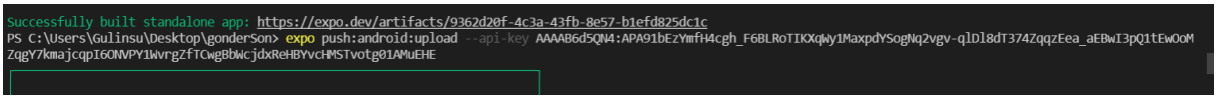

*Şekil24*

Push notification'ın oluşturulan apk'da da çalışabilmesi için firebase'den uygulamamız için bir android uygulama oluşturmamız gerekiyor.



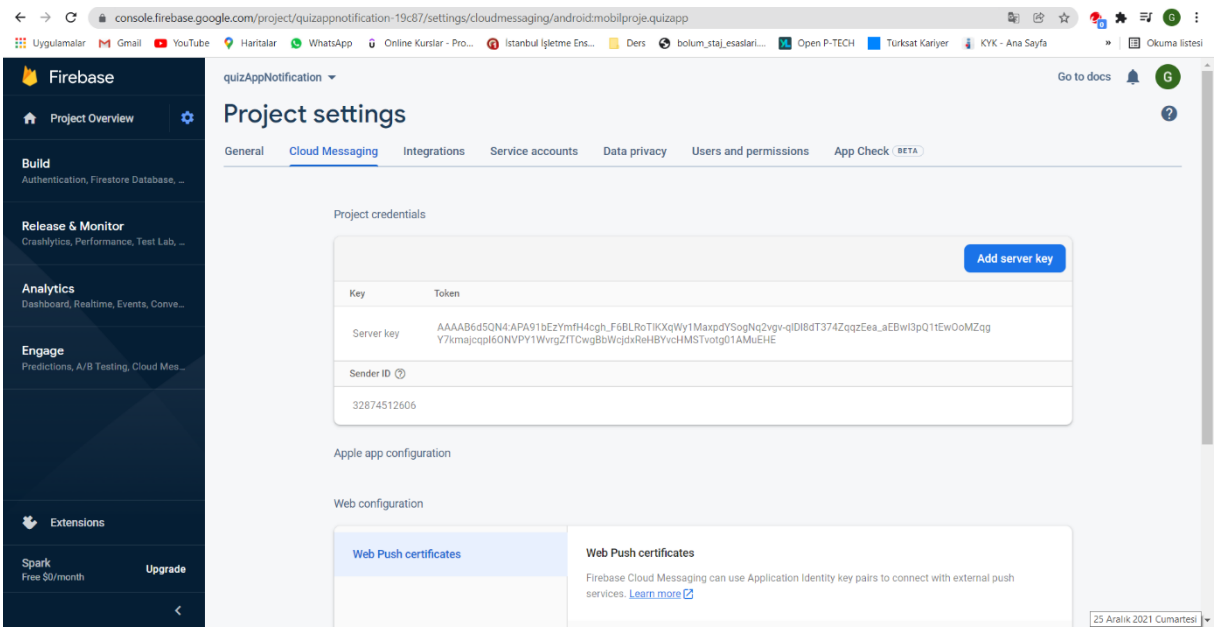

*Şekil25*

Uygulamayı oluşturduktan sonra google-services.json isimli dosyayı indirip projemize eklememiz gerekiyor. Şekil24'te ekleme işlemlerini gerçekleştirip app.json dosyasında googleServiceFile olarak belirtiyoruz.

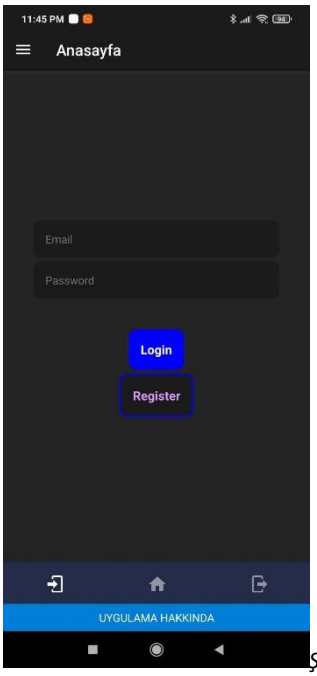



*Şekil26*

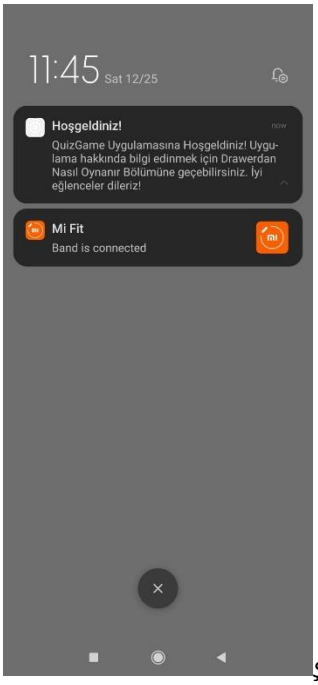
Şekil26'daki expo push:android:upload kodunu eklemezsek şekil24'te yaptığımız ayarlamalar yetersiz kalıyor. Bu kodu da çalıştırıyoruz ve push notification android için kullanılabilir hale geliyor.




*Şekil27: Şekil26'daki kodda verilen api key Firebase projemizdeki Cloud Messaging altındaki Server key'dir.*




*Şekil28*

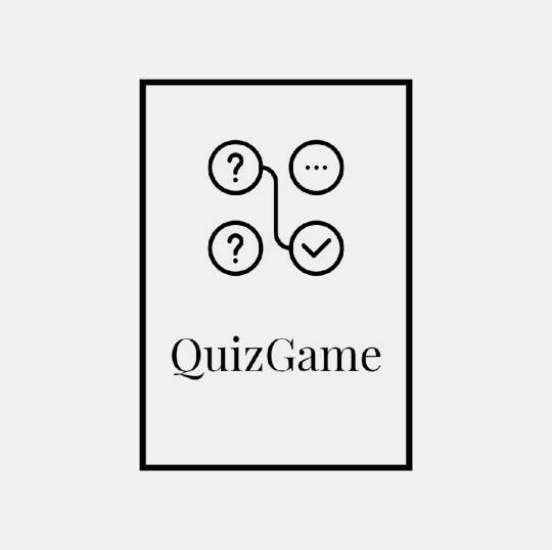



*Şekil29*

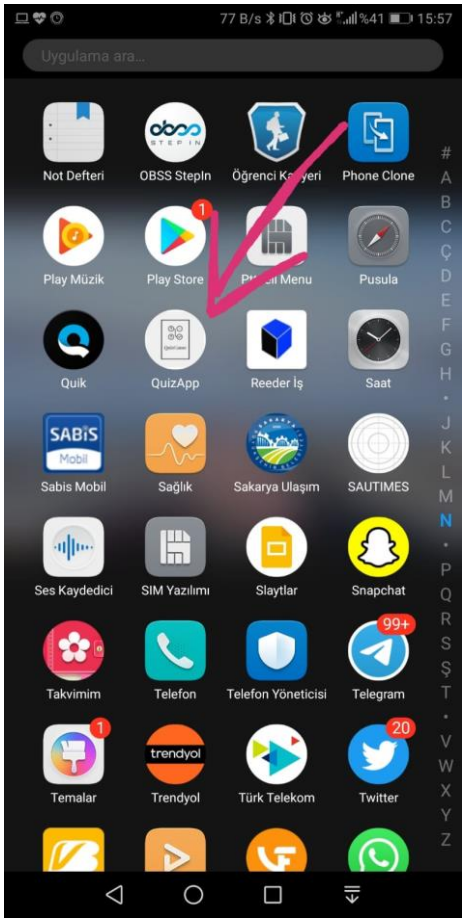
Notification ayarları tamamen tamamlandıktan sonra uygulama ekranındaki “UYGULAMA HAKKINDA” Butonuna basıldığında Şekil29’da görüldüğü gibi bir bildirim gönderimi gerçekleşmektedir.

## 5-Uygulamanın Son Görünümü

Uygulamamıza uygun aşağıdaki ikonu tasarladık ve uygulamamızın telefona indirildikten sonraki görüntüsü ektedir.




*Şekil30*




*Şekil31*

Uygulamanın apk’sının bulunduğu Google Drive linki ektedir:

https://drive.google.com/file/d/1wSPN1xRicLE-TvKMF-Kz-QWOzSXkFxgy/view?usp=sharing