MOBİL UYGULAMA GELİŞTİRME

Konu: İngilizce Türkce Flashcard Uygulaması

Dersi Veren : Öğr.Gör. AHMET ŞANSLI

Fatih MEMUR – G211210377

Süleyman Barış DALLI – B201210069

Githup :

I

İNGİLİZCE TÜRKÇE FLASHCARD UYGULAMASI

Uygulama, kullanıcıların kelime dağarcığını geliştirmek ve öğrenme sürecini desteklemek amacıyla tasarlanmıştır. Kullanıcılar, uygulama aracılığıyla kelime kartlarını inceleyebilir, doğru/yanlış ve favori olarak işaretleyebilir. Kendi gelişimlerini bu şekilde takip edebilirler.

Özellikler

Uygulamamız, kullanıcıların öğrenme sürecini optimize etmek için çeşitli özellikler sunmaktadır:

- **Kelime Kartları:** Kullanıcılar, her kartta İngilizce ve Türkçe karşılıklarıyla birlikte kelimeyi görebilir. Kartın ön ve arka yüzü olarak iki ayrı görünüm sunulur.

- **Doğru/Yanlış İşaretleme:** Kullanıcılar, her kelimenin öğrenme durumunu doğru veya yanlış olarak işaretleyebilir. Bu işaretlemeler, kullanıcının ilerleyişini takip etmesine ve zayıf noktalarını belirlemesine olanak tanır.

- **Favori Kelimeler:** Kullanıcılar, özellikle önemli veya öğrenmeye değer buldukları kelimeleri favori olarak işaretleyebilir ve daha sonra kolayca erişebilirler.

- **Telaffuz Kontrolü:** Kullanıcılar, herhangi bir kelimenin telaffuzunu dinleyebilir. Bu özellik, dil öğrenme sürecinde pratik yapmayı ve telaffuz becerilerini geliştirmeyi destekler.

- **Kelimelerin açıklaması:** Uygulamamız, her kelime için sadece bir tanım değil, aynı zamanda çağrışımlar yapmaya olanak sağlayan zengin açıklamalar sunar. Bu yaklaşım, kullanıcıların sadece kelimeleri ezberlemek yerine, onları daha geniş bir bağlam içinde anlamlandırmalarına yardımcı olur. Her kelime, sadece bir tanım değil, zenginleştirilmiş bir anlam dünyası sunar.

- **Örnek cümleler:** Her kelimenin yanında sunulan örnek cümleler, kelimenin gerçek hayattaki kullanımını gösterir. Bu örnekler, kelimenin sadece anlamını değil, aynı zamanda nasıl kullanıldığını ve hangi bağlamlarda yer aldığını gösterir. Kullanıcılarımız, örnek cümleler aracılığıyla daha sağlam bir kelime altyapısı oluşturabilir ve dil becerilerini daha iyi geliştirebilirler.

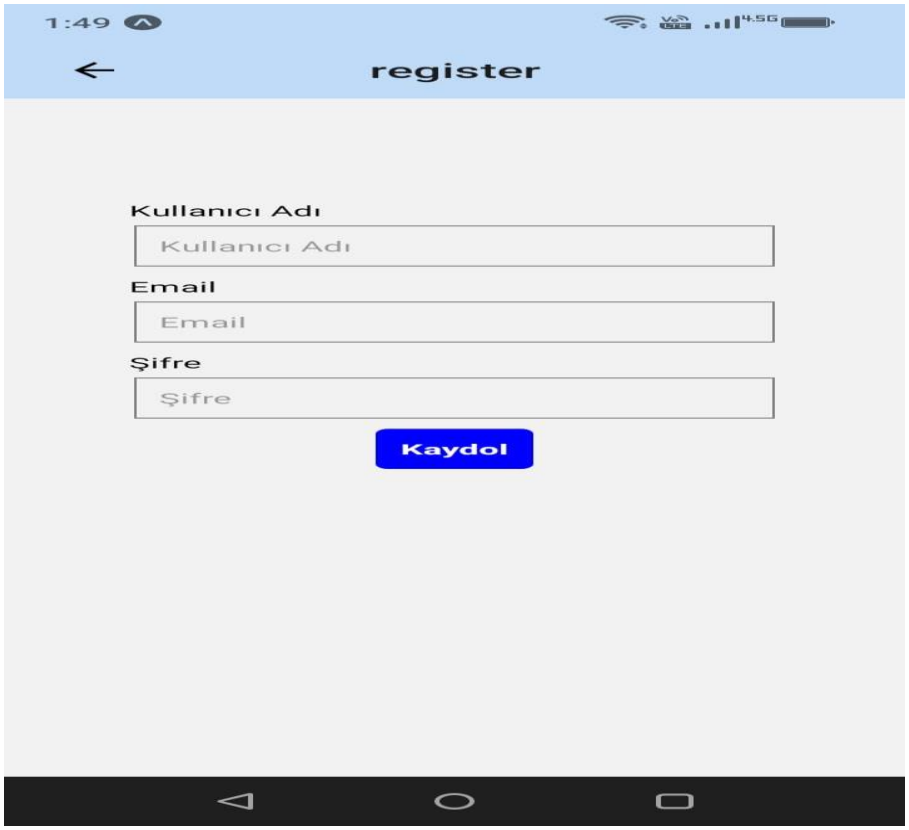
GENEL SAYFA YAPISI VE NAVİGATOR

```
return(
<NavigationContainer>
  <stack.Navigator>
    <stack.Screen
        name="login"
        component={login}
        options={{
          headerStyle: {backgroundColor: COLORS.TERTIARY},
          headerShown: true,
          headerShadowVisible: false,
          headerBackVisible: false,
          headerTitleAlign: 'center',

        }}
      />

  <stack.Screen
        name="register"
        component={register}
        options={{
          headerStyle: {backgroundColor: COLORS.TERTIARY},
          headerShown: true,
          headerShadowVisible: false,
          headerBackVisible: false,
          headerTitleAlign: 'center',
          headerLeft: () => (
            <GoBackButton/>
          )
        }}
      />
```

```
<stack.Screen
  name="Home"
  component={Main}
  options={{
    headerStyle:{backgroundColor:COLORS.TERTIARY},
    title:'AnaSayfa',
    headerShown:true,
    headerShadowVisible:false,
    headerBackVisible:false,
    headerTitleAlign:'center',
    headerLeft: () =>(
        <CustomButton navigate ='Menu' icon={<Entypo name="menu" size={24} color="black" />} />
    ),
    headerRight : () =>(
        <CustomButton navigate ='Profile' icon={<Feather name="user" size={24} color="black" />} />
    )
  }}
/>
```

Uygulama yüklendiğinde kullanıcı login sayfasına yönlendirilir eğer bir hesabı yoksa register sayfasında işlemini yaptıktan sonra tekrardan login olur ve anasayfaya yönlendirilir.

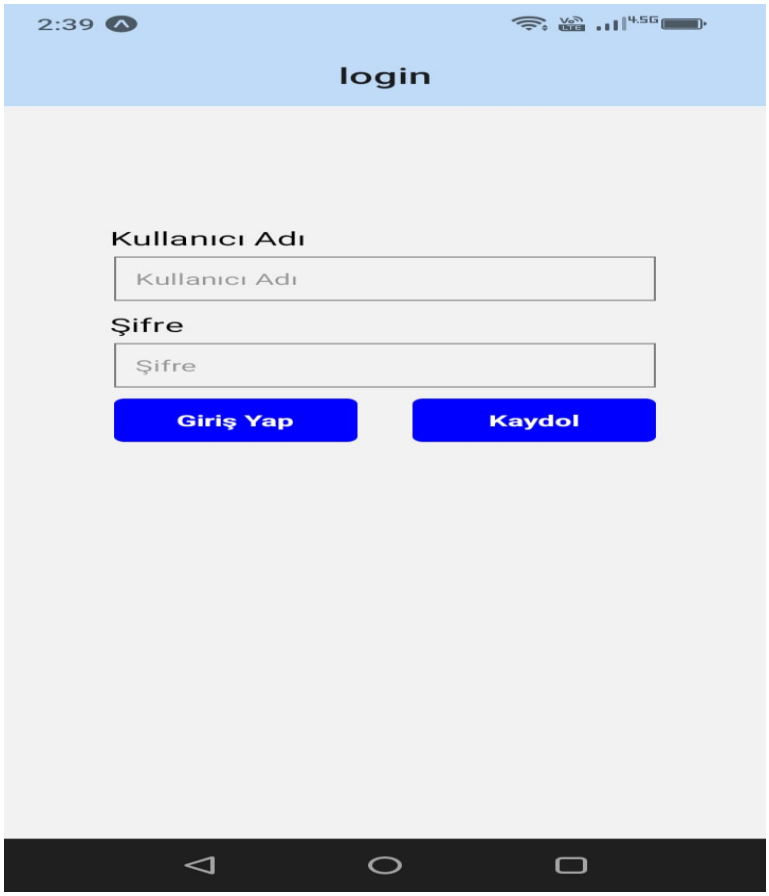
LOGİN VE REGİSTER



Kullanıcıdan alınan Kullanıcı Adı Email ve Sifre kontrolü yapılır sonrasında kontrollerin hatasız geçmesi durumunda kullanıcıyı login sayfasına yönlendirir.

```
db.transaction((tx) => {
    tx.executeSql(
        'CREATE TABLE IF NOT EXISTS users (userid INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT NOT NULL UNIQUE, email TEXT NOT NULL UNIQUE, password TEXT
        [],
        (tx, result) => {
          console.log('Tablo oluşturuldu.');
        },
        (tx, error) => {
          console.log('Tablo oluşturulurken hata oluştu:', error);
        }
      );
    tx.executeSql(
        'SELECT * FROM users WHERE email = ? OR userName = ?;',
        [email, username],
        (tx, result) => {
            if (result.rows.length > 0) {
                const existingUser = result.rows.item(0);
                if (existingUser.email === email) {
                    Alert.alert('Bu e-posta adresi zaten kullan□l□yor.');
                } else if (existingUser.userName === username) {
                    Alert.alert('Bu kullan□c□ ad□ zaten kullan□l□yor.');
                }
            } else {
                tx.executeSql(
                    'INSERT INTO users (userName, email, password) VALUES (?, ?, ?);',
                    [username, email, password],
                    (tx, result) => {
                        Alert.alert('Kay□t olma başar□l□');
                    },
                    (tx, error) => {
                        console.log('Kullan□c□ eklenirken hata oluştu:', error);
                    }
                );
```

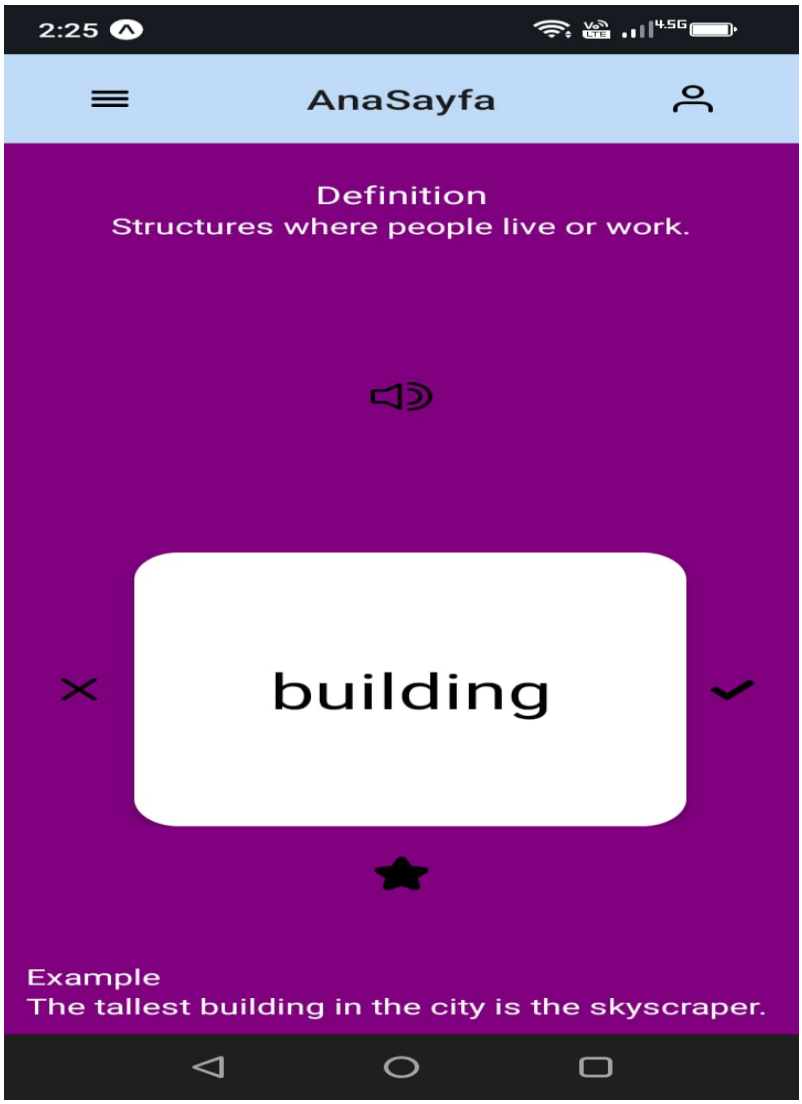
```
return (
    <View style={styles.container}>
    <Text style={styles.label}>Kullanıcı Adı</Text>
    <TextInput
        style={styles.input}
        onChangeText={(text) => setUsername(text)}
        value={username}
        placeholder="Kullanıcı Adı"
    />
    <Text style={styles.label}>Email</Text>
    <TextInput
        style={styles.input}
        onChangeText={(text) => setEmail(text)}
        value={email}
        placeholder="Email"
        keyboardType="email-address"
    />
    <Text style={styles.label}>Şifre</Text>
    <TextInput
        style={styles.input}
        onChangeText={(text) => setPassword(text)}
        value={password}
        secureTextEntry={true}
        placeholder="Şifre"
    />
    <TouchableOpacity style={styles.button} onPress={handleSignUp}>
        <Text style={styles.buttonText}>Kaydol</Text>
    </TouchableOpacity>
    </View>
);
};
```




```
const LoginScreen = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const navigation = useNavigation();

  const handleLogin = () => {
    if(username==''){
        Alert.alert('kullanıcı adı boş bırakılamaz.')
    }
    else if(password==''){
        Alert.alert('şifre boş bırakılamaz.')
    }
    else{
    db.transaction((tx) => {
      tx.executeSql(
        'SELECT * FROM users WHERE userName = ? AND password = ?;',
        [username, password],
        (tx, result) => {
          if (result.rows.length > 0) {
            navigation.navigate('Home');
            console.log('Giriş başarılı. Kullanıcı:');
          } else {
            Alert.alert('Kullanıcı adı veya şifre hatalı.');
            console.log('Kullanıcı bulunamadı veya hatalı giriş.');
          }
        },
        (tx, error) => {
          console.log('Kullanıcı girişi sırasında hata:', error);
        }
      );
    });
  }
  };
```

```
return (
  <View style={styles.container}>
    <Text style={styles.label}>Kullan□c□ Ad□</Text>
    <TextInput
      style={styles.input}
      onChangeText={(text) => setUsername(text)}
      value={username}
      placeholder="Kullan□c□ Ad□"
    />
    <Text style={styles.label}>$ifre</Text>
    <TextInput
      style={styles.input}
      onChangeText={(text) => setPassword(text)}
      value={password}
      secureTextEntry={true}
      placeholder="$ifre"
    />
    <View style={styles.buttonContainer}>
      <TouchableOpacity style={styles.button} onPress={handleLogin}>
        <Text style={styles.buttonText}>Giriş Yap</Text>
      </TouchableOpacity>
      <TouchableOpacity style={styles.button} onPress={handleSignUp}>
        <Text style={styles.buttonText}>Kaydol</Text>
      </TouchableOpacity>
    </View>
  </View>
);
```

ANASAYFA



Kullanıcı ;

- Doğru butonuna tıklarsa, o kelimeyi doğru kategorisine ekler.
- Yanlış butonuna tıklarsa, kelimeyi yanlışlar listesine ekler.
- Favori butonuna tıklarsa, kelimeyi favorilerine ekler.
- Doğru favori ve yanlış butonlara uzun süre tıklanıldığı taktirde ilgili liste ekrana getirilir

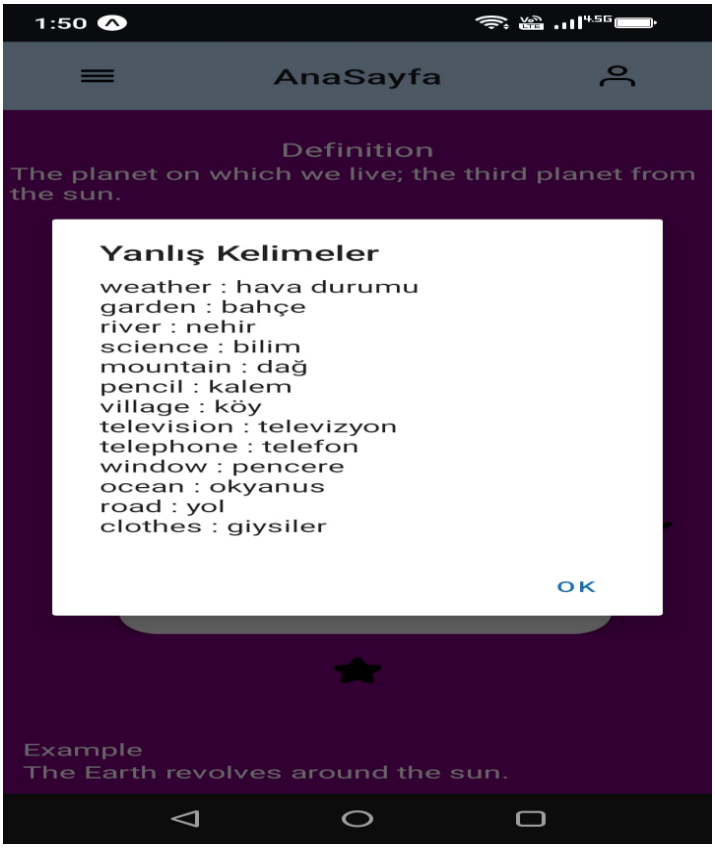
Kelimenin yanında yer alan hoparlör simgesine tıkladığında:

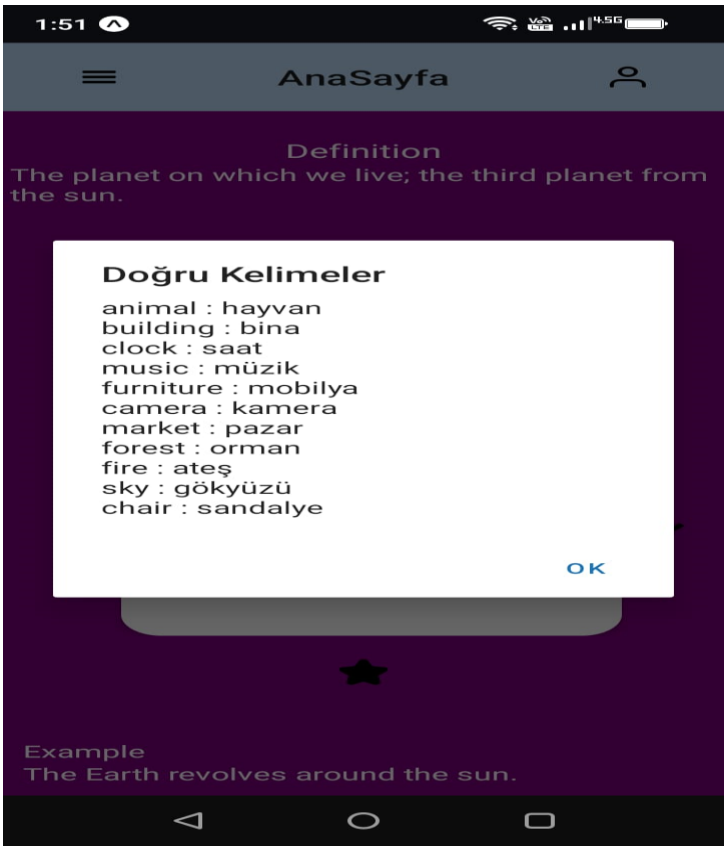
- Tek bir tıklama ile kelimenin telaffuzunu dinleyebilir.
- Uzun tıklama ile kelimenin açıklamasını dinleyebilir.

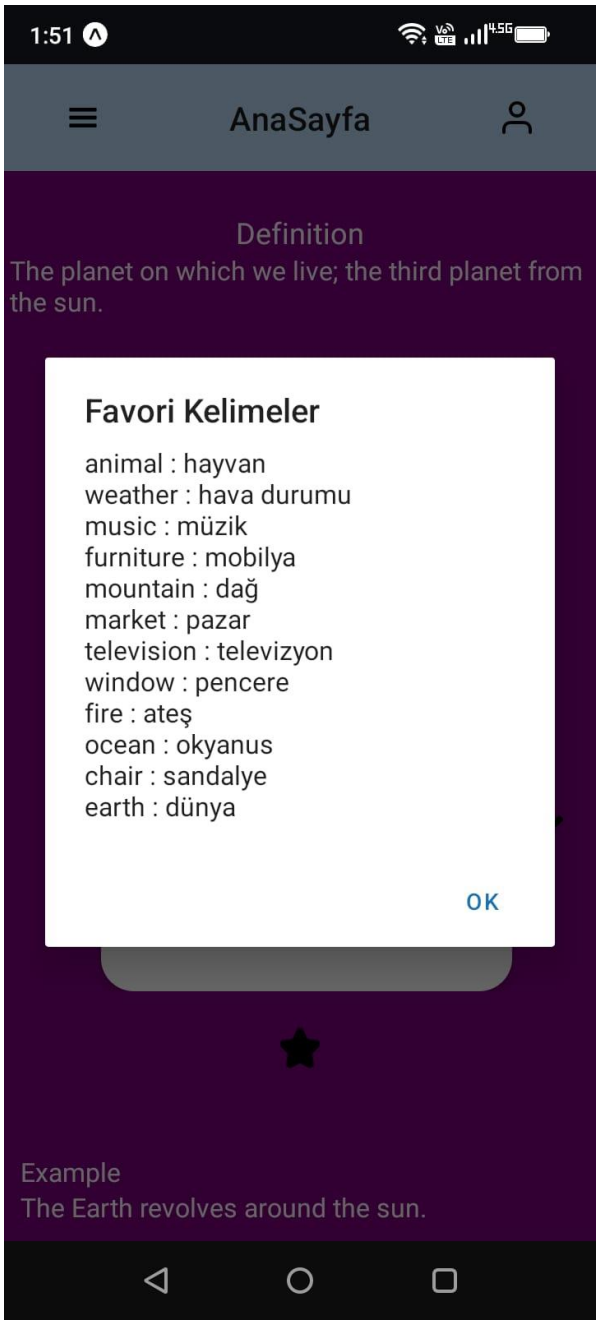
```
<View style={styles.background}>


    <><View style={styles.header}>
        <Text style={styles.headerText}>Definition</Text>
    </View><View style={styles.definition}>
            <Text style={styles.definitionText}>{flipCardData[currentIndex].description}</Text>
        </View><>
            <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
                <TouchableOpacity
                    style={styles.buttoni}
                    onPress={() => speakText(flipCardData[currentIndex].face) }
                    onLongPress={() => speakText(flipCardData[currentIndex].description)}

                >
                    <AntDesign name="sound" size={28} color="black" />
                </TouchableOpacity>
            </View>
            <View>
```

```
<View style={styles.container}>

    <TouchableOpacity style={styles.button} onPress={() => onPressHandlerYanlis('Yanlışlarım listesine eklendi lütfen daha dikkatli ol')} onLongPress={onLongPress
        <Feather name='x' size={28} color='black' />
    </TouchableOpacity>
    {flipCardData.map((card, index) => (
        index === currentIndex && (
            <FlipCard key={card.id} style={styles.cardContainer}>
                <View style={styles.face}>
                    <Text style={styles.text}>{card.face}</Text>
                </View>
                <View style={styles.back}>
                    <Text style={styles.text}>{card.back}</Text>
                </View>
                <TouchableOpacity style={styles.button} onPress={() => onPressHandlerDogru('Bravo bir kelime daha öğrendin')} onLongPress={onLongPressHandlerDogru
                    <FontAwesome5 name='check' size={28} color='black' />

                </TouchableOpacity>
            </FlipCard>
```

```
                )
            ))}
            <TouchableOpacity style={styles.button} onPress={() => onPressHandlerDogru('Bravo bir kelime daha öğrendin')} onLongPress={onLongPressHandlerDogru}>
                <FontAwesome5 name='check' size={20} color='black' />

            </TouchableOpacity>

        </View>

    </View>
    <TouchableOpacity style={styles.button} onPress={() => onPressHandlerFavori('Favoriler listesine eklendi.')} onLongPress={onLongPressHandlerFavori}>
        <AntDesign name='star' size={28} color='black' />
    </TouchableOpacity>


</View>
<View style={styles.exampleSentences}>
    <Text style={styles.exampleSentencesText}>Example</Text>
    <Text style={styles.exampleSentencesText}>{flipCardData[currentIndex].example}</Text>

</View>
<Toast ref={(ref) => Toast.setRef(ref)} /></></>
```


1:50
AnaSayfa
Definition
The planet on which we live; the third planet from the sun.
Yanlış Kelimeler
weather : hava durumu
garden : bahçe
river : nehir
science : bilim
mountain : dağ
pencil : kalem
village : köy
television : televizyon
telephone : telefon
window : pencere
ocean : okyanus
road : yol
clothes : giysiler
OK
Example
The Earth revolves around the sun.


1:51
AnaSayfa
Definition
The planet on which we live; the third planet from the sun.
Doğru Kelimeler
animal : hayvan
building : bina
clock : saat
music : müzik
furniture : mobilya
camera : kamera
market : pazar
forest : orman
fire : ateş
sky : gökyüzü
chair : sandalye
OK
Example
The Earth revolves around the sun.


1:51
AnaSayfa
Definition
The planet on which we live; the third planet from the sun.
Favori Kelimeler
animal : hayvan
weather : hava durumu
music : müzik
furniture : mobilya
mountain : dağ
market : pazar
television : televizyon
window : pencere
fire : ateş
ocean : okyanus
chair : sandalye
earth : dünya
OK
Example
The Earth revolves around the sun.

```
const speakText = async (text) => {
    try {
        await Speech.speak(text, { language: 'en' }); // Türkçe örneği
    } catch (error) {
        console.error('Seslendirme hatası:', error);
    }
};

const flipCardData = wordsData;
const [currentIndex, setCurrentIndex] = useState(0);
const [wrongWords, setWrongWords] = useState([]);
const [favoriteWords, setFavoriteWords] = useState([]);
const [correctWords, setCorrectWords] = useState([]);

const handleNextWord = () => {
    if (currentIndex < flipCardData.length - 1) {
        setCurrentIndex(currentIndex + 1);
    }
};

const onPressHandlerDogru = (message) => {
    setCorrectWords((prevWords) => [
        ...prevWords,
        { face: flipCardData[currentIndex].face, back: flipCardData[currentIndex].back },
    ]);
    Toast.show({
        type: 'success',
        text1: 'Bildirim',
        text2: message,
    });
    handleNextWord();
};
```

```
const onPressHandlerYanlis = (message) => {
    setWrongWords((prevWords) => [
        ...prevWords,
        { face: flipCardData[currentIndex].face, back: flipCardData[curr
    ]);
    Toast.show({
        type: 'error',
        text1: 'Bildirim',
        text2: message,
    });
    handleNextWord();
};
const onPressHandlerFavori = (message) => {
    const currentCard = flipCardData[currentIndex];

    // Favori kelimeler listesinde o kelimenin olup olmadığını kontrol e
    const isAlreadyFavorite = favoriteWords.some(word => word.face === c

    // Eğer favori kelimeler listesinde yoksa ekle
    if (!isAlreadyFavorite) {
        setFavoriteWords((prevWords) => [
            ...prevWords,
            { face: currentCard.face, back: currentCard.back },
        ]);
        Toast.show({
            type: 'info',
            text1: 'Bildirim',
            text2: message,
        });
```

```
const onLongPressHandlerYanlis = () => {
    // const currentCard = flipCardData[currentIndex];
    //rsetWrongWords(prevWords => [...prevWords, { face: currentCard.face, back: currentCard.back }]);

    let message = "";
    wrongWords.forEach((word, index) => {
        message += `${word.face} : ${word.back}\n`;
    });
    Alert.alert('Yanlış Kelimeler', message);
};

const onLongPressHandlerFavori = () => {
    //const currentCard = flipCardData[currentIndex];
    // setFavoriteWords(prevWords => [...prevWords, { face: currentCard.face, back: currentCard.back }]);

    let message = "";
    favoriteWords.forEach((word, index) => {
        message += `${word.face} : ${word.back}\n`;
    });
    Alert.alert('Favori Kelimeler', message);
};

const onLongPressHandlerDogru = () => {
    // const currentCard = flipCardData[currentIndex];
    //setCorrectWords(prevWords => [...prevWords, { face: currentCard.face, back: currentCard.back }]);

    let message = "";
    correctWords.forEach((word, index) => {
        message += `${word.face} : ${word.back}\n`;
    });
    Alert.alert('Doğru Kelimeler', message);
};
```