SAKARYA
ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

MOBİL UYGULAMA GELİŞTİRME PROJE ÖDEVİ

Hüseyin Burhan BAŞARAN – G191210077 1. Öğretim A Grubu

Muhammet Enes VARDAR – G191210101 1. Öğretim A Grubu

Ahmet Mete DOKGÖZ – G191210053 1. Öğretim A Grubu

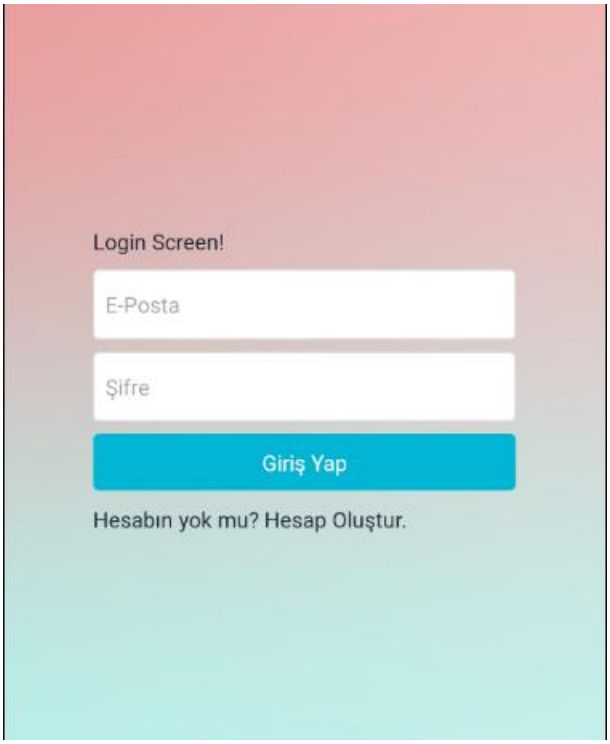
Mobil Uygulamamızın APK'sı

Mobil uygulamamızın boyutu fazla olduğundan dolayı APK'yı drive'a atıp drive linkini aşağı ekledik.

https://drive.google.com/file/d/1j-9JBLF7O8G6GSJJzH6eSxbcUNbYAa2t/view?usp=sharing

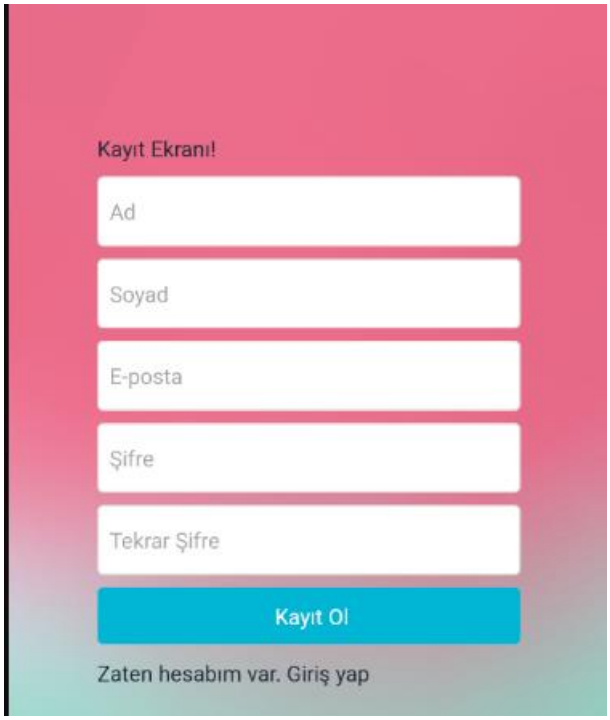
Mobil Uygulama

Uygulamamız dünya üzerindeki kullanılan para birimlerinin güncel olarak fiyatına ulaşabileceğimiz bir platformdur. Bununla birlikte kişilere uyarı gönderebilecek halde olup kişilerin dövizleri takip etmesini hatırlamasını da sağlamaktadır.



Uygulamamızın giriş ekranı. Bu ekranda eğer hesabınız varsa direk bilgilerinizi girerek giriş yapabileceğiniz bir sistem bulunmaktadır.

Hesabınızın olmadığı durumlarda ise hesap oluşturabileceğiniz bir buton bulunmaktadır ve hesap oluşturma sayfasına aktarılmaktasınız.

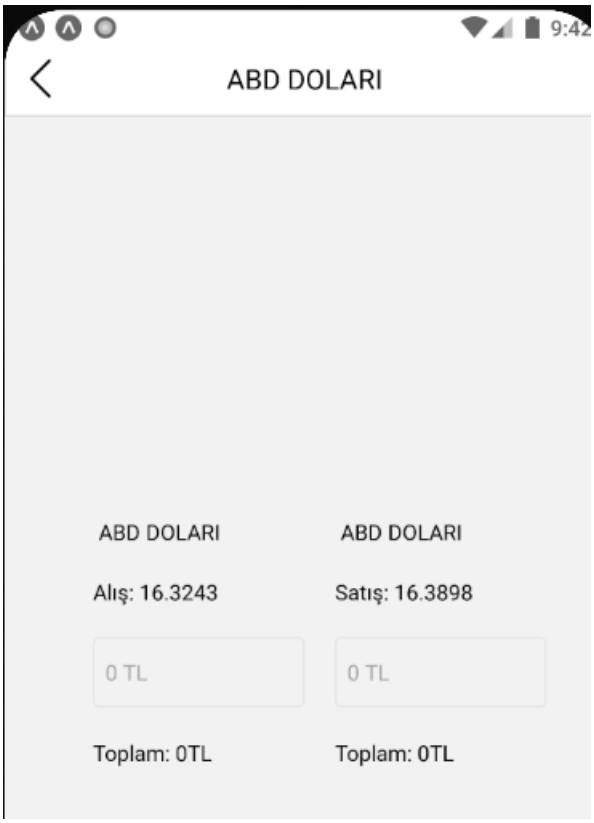


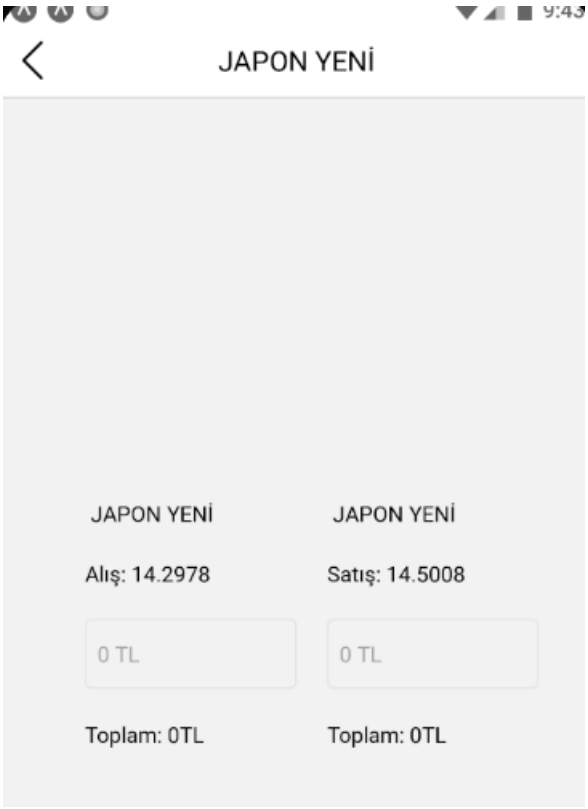
Kayıt olma ekranında isim, soyisim, email adresi ve şifrenizi yazmanızı istemektedir. Bu bilgileri girdikten sonra kayıt işleminiz başarıyla gerçekleşecektir.

Girdiğiniz bilgiler firebase'de tutularak yeniden giriş yapmak istediğinizde oradan ulaşılan bilgileri kullanarak doğrulama yapacaktır.

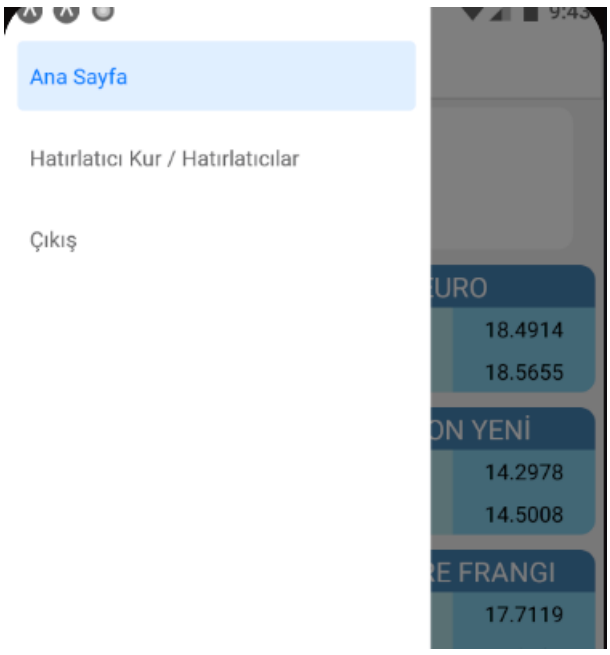
Uygulamamızın ana sayfa tasarımı. Ana sayfa üzerinde paraların alış ve satışlarını gösteren bir arayüz bulunmaktadır. Buradan kişiler ilgilendiği dövizlerin alış ve satış fiyatlarını görebilecektir.

Kullanıcıların eğer almayı düşündüğü bir döviz bulunmakta ise o para biriminin üzerine tıklayarak almak istediği tutarın maliyetini hesaplayabileceği bir sayfaya iletilmektedir. Bu iletim stack navigatör ile yapılmaktadır.

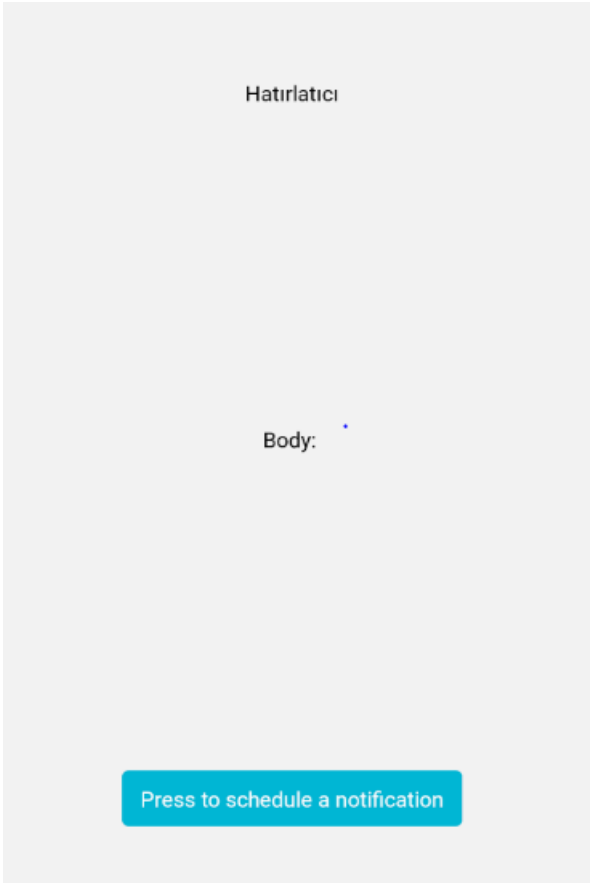




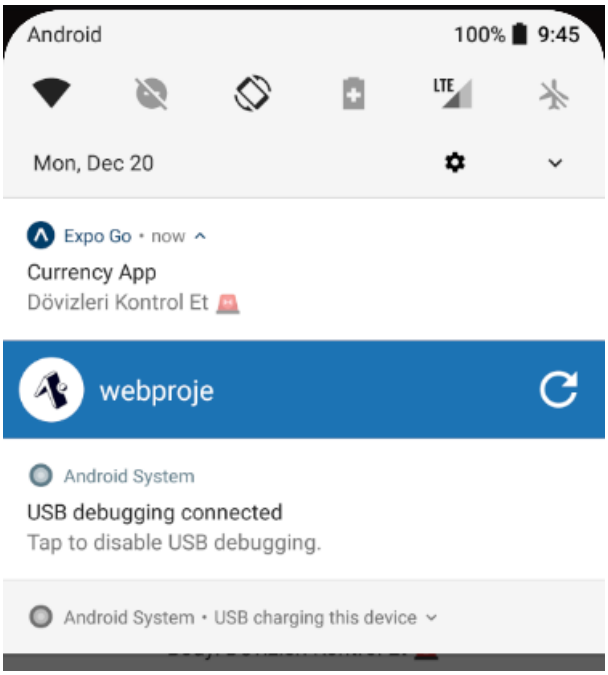
Alış veya satış yapmak istediğiniz dövizlerin istediğiniz miktarında ne kadar para ile alım veya satış yapacağınızın gösterildiği sayfa.



Uygulamamızda ana sayfaya, hatırlatıcı sayfasına ve çıkış sayfasına erişebileciğiz bir drawer navigation menü bulunmaktadır.

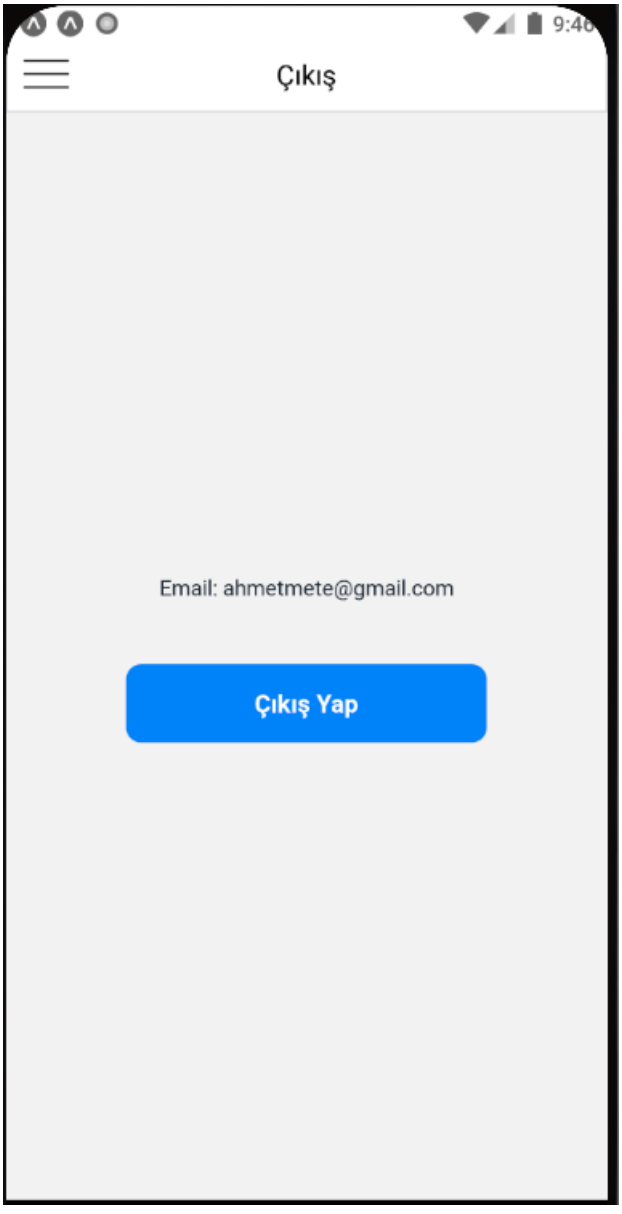


Hatırlatıcı sayfasında kullanıcıların dövizleri takip etmesini unutmamasını sağlamak amacıyla oluşturulan hatırlatıcıyı göstermekte. Hatırlatıcının bulunmadığı durumlarda ise yeni hatırlatıcı oluşturmaya imkan sağlamaktadır.



Uygulamamızda ana sayfa ve hazırlayanlar olmak üzere 2 sayfa arasında geçiş yapılabilecek bir tab navigatör bulunmaktadır.

Tab drawer'a tıklayarak hazırlayanlar sayfasına geçtiğimiz durumda bu sayfa ile karşılaşmaktayız.



Son olarak programda kullanıcı çıkış yapmak için drawer menüden çıkış seçeneğini seçerek çıkış sayfasına yönlendirilir. Bu yönlendirmenin ardından çıkış yap butonuna basarak uygulamadan çıkış yapmaktadır. Kullanıcı çıkış işlemi gerçekleştikten sonra giriş sayfasına yönlendirilir.

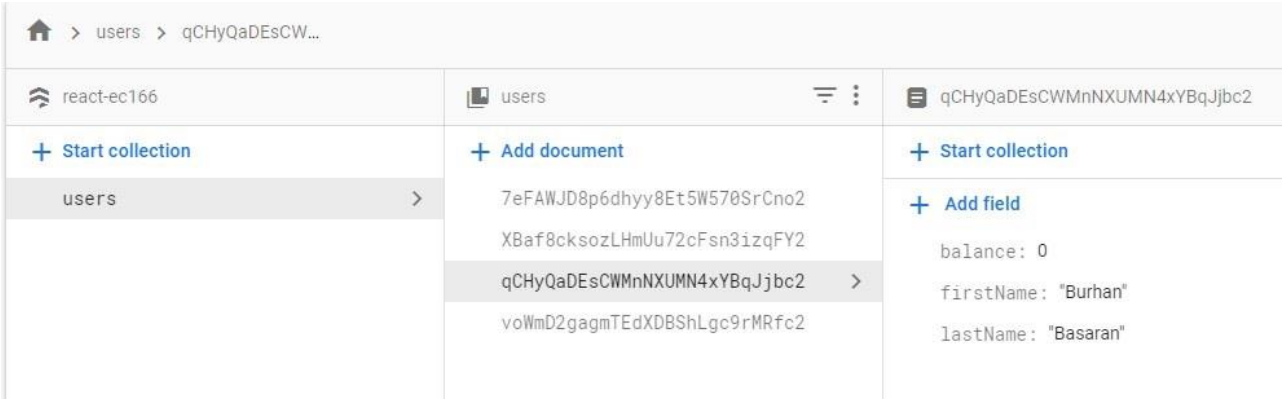
Uygulamanın firebase ile bağlantılarının yapıldığı ve firestore'un kullanıldığı kodlar.

```
// Import the functions you need from the SDKs you need
import * as firebase from "firebase";
import "firebase/firestore";

// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {


  apiKey: "AIzaSyCaiYK09SHcEs1T9e6AQ3_yJlIJPJfmE44",
  authDomain: "react-ec166.firebaseapp.com",
  projectId: "react-ec166",
  storageBucket: "react-ec166.appspot.com",
  messagingSenderId: "475599971513",
  appId: "1:475599971513:web:63aa5fa1e966b5e6326768"

};

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

const db = firebase.database();
const dbfirestore = firebase.firestore();
const auth = firebase.auth();

export { auth , dbfirestore};
export default db;
```

```
function RegisterScreen (props) {
    const [name, setName] = useState('')
    const [lastName, setLastName] = useState('')
    const [email, setEmail] = useState('')
    const [password, setPassword] = useState('')
    const [password2, setPassword2] = useState('')

    const navigation = useNavigation()

    useEffect(() => {
        const unsubscribe = auth.onAuthStateChanged(user => {
        if (user) {
            navigation.replace("HomeApp")
        }
        })

        return unsubscribe
    }, [])


    const handleSignUp = () => {
        if(password==password2){
          auth
          .createUserWithEmailAndPassword(email, password)
          .then(userCredentials => {
            const user = userCredentials.user;
            console.log('Registered with:', user.email);



            dbfirestore.collection("users")
                .doc(user.uid)
                .set({
                    firstName: name,
                    lastName: lastName,
                    balance: 0,
          });
          })
          .catch(error => alert(error.message))
```

Uygulamada kullanıcı kayıt işleminin yapılmasını sağlayan kodlar. Burada yine Firebase authentication'da e-mail ve şifre ile kayıt işlemi gerçekleşir ve ardından firestore database'ine kullanıcının adı , soyadı ve bakiyesi users tablosunda kullanıcı id altına eklenir.

Identifier	Providers	Created ↓	Signed In	User UID
burhanbasaran@gmail.com	✉	Dec 21, 2021		Fbh2gxxLhiWVWO0b22gDFG80Ar...
enesvardar@gmail.com	✉	Dec 20, 2021	Dec 20, 2021	voWmD2gagmTEdXDBShLgc9rMR...
ahmetdokgoz@gmail.com	✉	Dec 20, 2021	Dec 20, 2021	7eFAWJD8p6dhyy8Et5W570SrCno2



```
function LoginScreen (props) {



    const [email, setEmail] = useState('')
    const [password, setPassword] = useState('')

    const navigation = useNavigation()

    useEffect(() => {
      const unsubscribe = auth.onAuthStateChanged(user => {
        if (user) {
            navigation.replace("HomeApp")
        }
      })

      return unsubscribe
    }, [])

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

Uygulamanın giriş ekranında kullanıcı girişinin yapılmasını sağlayan kodlar. Firebase authentication'da e-mail ve şifre ile giriş işlemi gerçekleşir. Giriş işlemi başarılı olursa ana sayfa ekranına yönlendirilir.

```
function SignOutScreen (props) {
  const navigation = useNavigation()

  const handleSignOut = () => {
    auth
      .signOut()
      .then(() => {
        navigation.replace("Login")
      })
      .catch(error => alert(error.message))
  }
```

Uygulamada çıkış yapma işlevini gerçekleştiren kodlar. Kullanıcı giriş sayfasına yönlendirilir.

Uygulamada push notification'ın uygulanmasını sağlayan kodlar.

```
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') {
      console.log('Failed to get push token for push notification!');
      return;
    }
    token = (await Notifications.getExpoPushTokenAsync()).data;
    console.log(token);
  } else {
    console.log('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',
    });
  }
  let uid = auth.currentUser.uid;
  db.ref("users").child(uid).update({
    expoPushToken: token
  });
  return token;
```

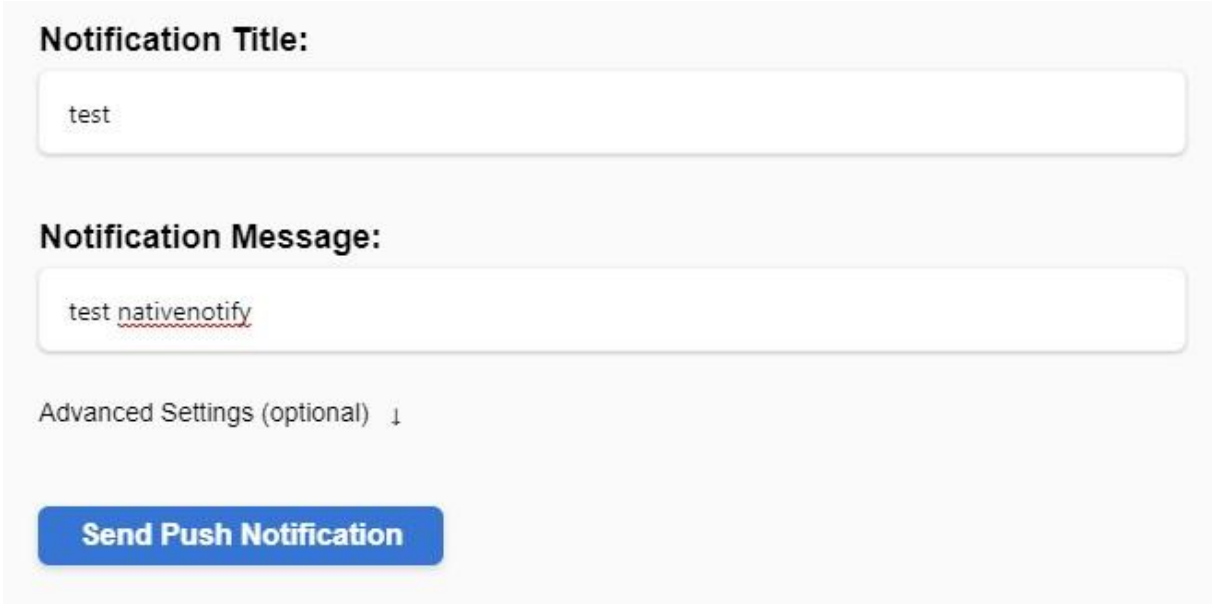
Her kullanıcı için benzersiz bir token oluşturulur. Bu token firebase realtime database'e users tablosunda kullanıcı idsi altına eklenir.

```
Notifications.setNotificationHandler({
    handleNotification: async () => ({
      shouldShowAlert: true,
      shouldPlaySound: false,
      shouldSetBadge: false,
    }),
  });

function NotificationsScreen (props) {

    const [expoPushToken, setExpoPushToken] = useState('');
    const [notification, setNotification] = useState(false);
    const notificationListener = useRef();
    const responseListener = useRef();

    useEffect(() => {

      registerForPushNotificationsAsync().then(token => {setExpoPushToken(token),
        axios.post(`https://app.nativenotify.com/api/expo/key`, { appId: 816, appToken: 'DVb32JJblpWfH3jiIfV1hY', expoToken: token })
     });

      notificationListener.current = Notifications.addNotificationReceivedListener(notification => {
        setNotification(notification);
      });

      responseListener.current = Notifications.addNotificationResponseReceivedListener(response => {
        console.log(response);
      });

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

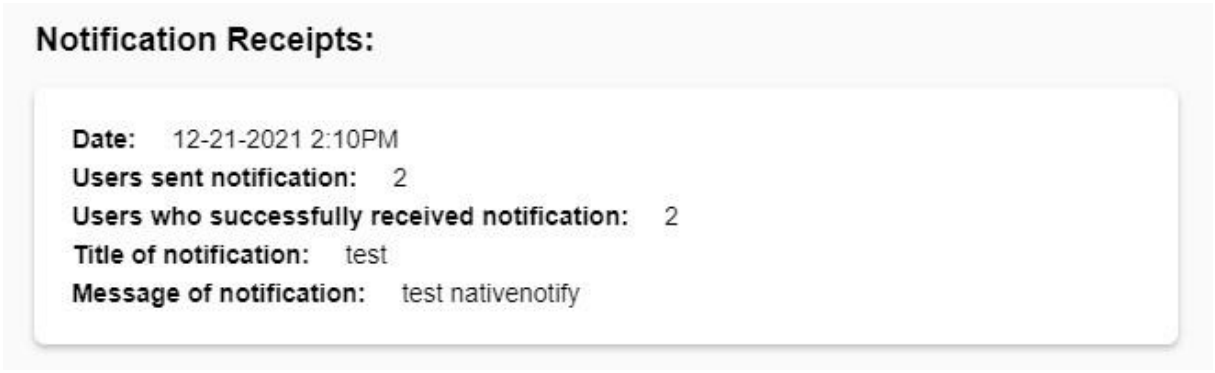
UseEffect ile token oluşturan fonksiyon çağırılır ve bu token nativenotify kütüphanesine eklenir. Bu kütüphane kullanıcı tokenlerini kaydeder ve bütün kullanıcılara toplu bildirim gönderebilmemizi sağlar.

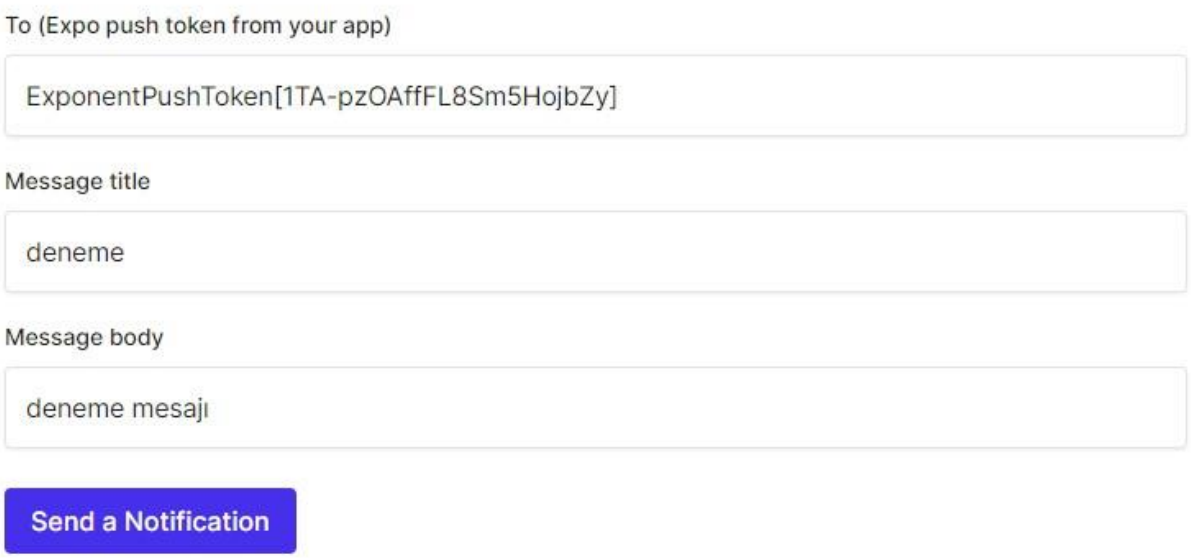
```
async function schedulePushNotification() {
  await Notifications.scheduleNotificationAsync({
    content: {
      title: "Currency App",
      body: 'Dövizleri Kontrol Et 🚨',
      data: { data: auth.currentUser.uid },
    },
    trigger: { seconds: 2 },
  });
}
```

Örnek Push notification oluşturur.

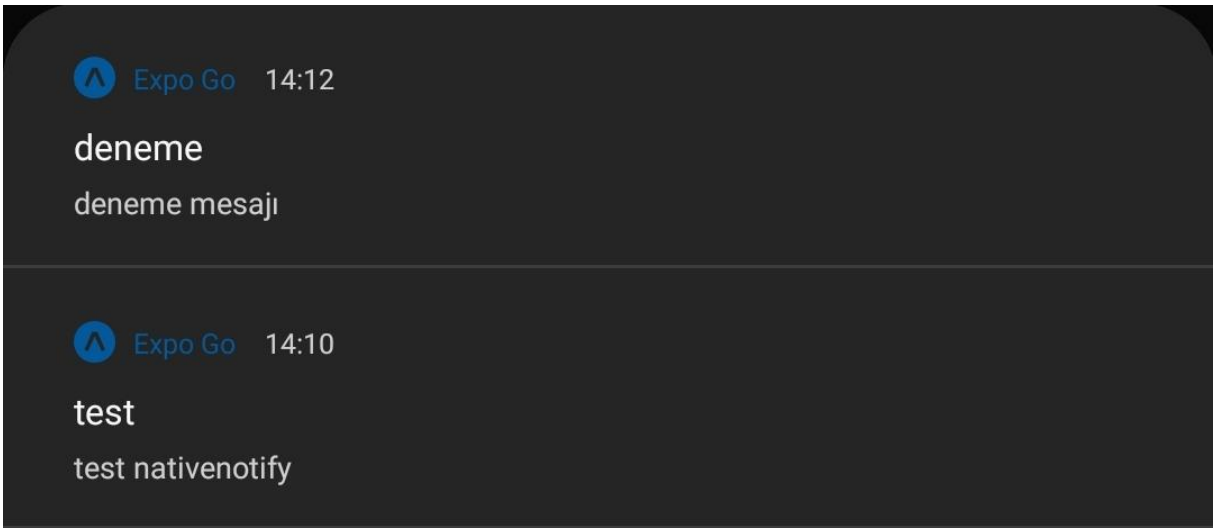


Nativenotify ile bütün kullanıcılara token girmeden bildirim gönderilebilir.





Expo push notification ile kullanıcılara tek tek token ile bildirim gönderilir.



```
return(
  <View style={{ flex: 1, flexDirection: "column" ,justifyContent: "space-around", margin: 5}}>
    <View style={{flex: 1, flexDirection: "column", alignItems: "center", justifyContent:"center", backgroundColor:"steelblue",borderTopStartRad
      <TouchableOpacity onPress={() => navigation.navigate('Currency', {name:isim,alisFiyatı:alis,satisFiyatı:satis})}>
        <RVText style={{color:"white"}} content={isim}/>

      </TouchableOpacity>

      </View>
  <View style={{ flex: 1, flexDirection: "row",justifyContent: "space-around"}}>

      <View style={{flex: 1, flexDirection: "row", alignItems: "center", justifyContent:"center", backgroundColor:"powderblue"}}>
        <Text >
         Alış
        </Text>
      </View>

      <View style={{flex: 1, flexDirection: "row", alignItems: "center", justifyContent:"center", backgroundColor:"skyblue"}}>
        <Text>
          {alis}
        </Text>
      </View>


  </View>
  <View style={{ flex: 1, flexDirection: "row",justifyContent: "space-around"}}>

      <View style={{flex: 1, flexDirection: "row", alignItems: "center", justifyContent:"center", backgroundColor:"powderblue",borderBottomStart
        <Text>
        Satış
        </Text>
      </View>

      <View style={{flex: 1, flexDirection: "row", alignItems: "center", justifyContent:"center", backgroundColor:"skyblue",borderBottomEndRadiu
        <Text>
          {satis}
        </Text>
      </View>
```

Ana sayfa üzerindeki dövizleri gösteren yapıların kodları.

```
const [al, setAl] = useState('0')
const [sat, setSat] = useState('0')


return (
    <SafeAreaView style={{ flex: 1 }}>
    <CustomHeader title={route.params.name} navigation={navigation}/>
    <Center flex={1} px="3" >
          <Stack
            mt={3}
            direction={{base: "row",md: "row" }}
            w={{ base: "75%", md: "100%", }}
            space={5}>


            <Stack
            mt={3}
            direction={{base: "column",md: "row" }}
            w={{ base: "50%", md: "25%", }}
            space={5}>
            <Text> {route.params.name}</Text>

          <Text>Alış: {route.params.alisFiyatı}</Text>
          <Input size="md" placeholder="0 TL" onChangeText={text => setAl(text)} />
          <Text>Toplam: {al*route.params.alisFiyatı}TL</Text>
        </Stack>
        <Stack
            mt={3}
            direction={{base: "column",md: "row" }}
            w={{ base: "50%", md: "25%", }}
            space={5}>
            <Text> {route.params.name}</Text>
          <Text>Satış: {route.params.satisFiyatı}</Text>
          <Input size="md" placeholder="0 TL" onChangeText={text => setSat(text)} />

          <Text>Toplam: {sat*route.params.satisFiyatı}TL</Text>
        </Stack>
        </Stack>
      </Center>
```

Kullanıcının dövizin üstüne tıkladığında döviz miktarını hesaplayabileceği kodlar.

```
export function HomeScreen({ navigation, route }) {
        const [name, setName] = useState('')
        const [lastName, setLastName] = useState('')
        const [balance, setBalance] = useState('')

        const Name= async()=>{
          let doc  = await dbfirestore.collection('users').doc(auth.currentUser.uid).get()
          let dataObj = doc.data();
          setName(dataObj.firstName);
          setLastName(dataObj.lastName);
          setBalance(dataObj.balance);
        }

        useEffect(() => { Name(); },[]);
        return (

            <SafeAreaView style={{ flex: 1 }}>
                <CustomHeader title="Ana Sayfa" isHome={true} navigation={navigation} />


                <View style={{flex: 1, justifyContent: 'center', alignItems: 'center'}}>


                <View style={{flex: 1, backgroundColor: 'white' ,flexDirection: "column" ,justifyContent: "center",alignItems:"center", margin: 5 ,wid

                <Text>Hosgeldin {name} {lastName} </Text>
                <Text>Bakiye {balance} </Text>

                <Image source={require('../../assets/favicon.png')} style={{width:50,height:50,resizeMode:'contain'}} />

                </View>


                 <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                   <EurLayout navigation={navigation} name={'USD'}/>
                   <EurLayout navigation={navigation} name={'EUR'}/>
                 </View>
                 <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                   <EurLayout navigation={navigation} name={'AUD'}/>
```

```
                 <Text>Hosgeldin {name} {lastName} </Text>
                 <Text>Bakiye {balance} </Text>

                 <Image source={require('../../assets/favicon.png')} style={{width:50,height:50,resizeMode:'contain'}} />

                 </View>


                  <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                    <EurLayout navigation={navigation} name={'USD'}/>
                    <EurLayout navigation={navigation} name={'EUR'}/>
                  </View>
                  <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                    <EurLayout navigation={navigation} name={'AUD'}/>
                    <EurLayout navigation={navigation} name={'JPY'}/>
                  </View>
                  <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                    <EurLayout navigation={navigation} name={'CAD'}/>
                    <EurLayout navigation={navigation} name={'CHF'}/>
                  </View>
                  <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                    <EurLayout navigation={navigation} name={'GBP'}/>
                    <EurLayout navigation={navigation} name={'NOK'}/>
                  </View>
                  <View style={{ flex: 1, flexDirection: "row" ,justifyContent: "space-around"}}>
                    <EurLayout navigation={navigation} name={'SAR'}/>
                    <EurLayout navigation={navigation} name={'SEK'}/>
                  </View>
                  <View style={{flex: 1, backgroundColor: 'white'}} />


                 </View>

            </SafeAreaView>
        );
}
```

Uygulamanın ana sayfasının kodları.

```
render() {
    return (
        <SafeAreaView style={{ flex: 1 }}>
        <CustomHeader title="Hazırlayanlar" isHome={true} navigation={this.props.navigation}/>
        <View style={{flex:2}}></View>
        <View style = {{textAlign: "center", flexDirection: "column"}}>
            <Text style ={{fontSize: 35, fontWeight:'bold'}}>
                Hazırlayanlar
            </Text>
        </View>
        <View style={{flex:1}}></View>
        <View style = {{textAlign: "center", flexDirection: "column"}}>
            <Text style ={{fontSize: 30}}>
                Burhan BAŞARAN
            </Text>
        </View>
        <View style={{flex:1}}></View>
        <View >
            <Text style ={{fontSize: 30}}>
                Muhammet Enes VARDAR
            </Text>
        </View>
        <View style={{flex:1}}></View>
        <View >
            <Text style ={{fontSize: 30}}>
                Ahmet Mete DOKGÖZ
            </Text>
        </View>
        <View style={{flex:3}}></View>
        </SafeAreaView>
    );
}
```

Hazırlayanlar sayfasının kodları.

```
export class CustomHeader extends Component {
    render() {
        let {navigation, isHome, title} = this.props
        return (
            <View style={{flexDirection: 'row', height: 70, backgroundColor:'#ffffff',borderWidth: 1,borderColor:'#d8d8d8'}}>
            <View style={{flex: 1, justifyContent: 'center',marginTop:20}}>
                {
                isHome ?
                <TouchableOpacity onPress={() => navigation.openDrawer()}>
                    <Image style={{width: 30, height: 30, marginLeft: 10}}
                    source={IMAGE.ICON_MENU}
                    resizeMode="contain"/>
                </TouchableOpacity>
                :
                <TouchableOpacity
                style={{flexDirection: 'row', alignItems: 'center'}}
                onPress={() => navigation.goBack()}
                >
                    <Image style={{width: 25, height: 25, marginLeft: 10,marginRight:10}}
                    source={IMAGE.ICON_BACK}
                    resizeMode="contain"
                    />
                </TouchableOpacity>
                }
            </View>

                <View style={{flex: 1.5, justifyContent: 'center',marginTop:20}}>
                <Text style={{textAlign: 'center',fontSize:18}}>{title}</Text>
                </View>
                <View style={{flex: 1}}></View>
            </View>
  )
    }
}
```

Sayfaların üstünde bulunan başlık kısmının kodları.

```
export const getDoviz = async woeid => {
  const response = await fetch(
    `https://dovizkurlari-l6vtviaacq-uc.a.run.app/api/doviz/${woeid}`,
  );
  const { BanknoteBuying, BanknoteSelling, isim } = await response.json();

  return {
    alis: BanknoteBuying,
    satis: BanknoteSelling,
    isim:isim,
  };
};
```

Uygulamada API’ın kullanım şekli.

```
const StackHome = createStackNavigator()
function HomeStack() {
  return (
    <StackHome.Navigator initialRouteName="Home" screenOptions={{ headerShown: false }}>
      <StackHome.Screen name="Home" component={HomeScreen} />
      <StackHome.Screen name="HomeDetail" component={HomeScreenDetail} />
      <StackHome.Screen name="Currency" component={CurrencyScreen} />
    </StackHome.Navigator>
  )
}


const StackSetting = createStackNavigator()
function SettingStack() {
  return (
    <StackSetting.Navigator initialRouteName="Setting" screenOptions={{ headerShown: false }}>
      <StackSetting.Screen name="Setting" component={SettingsScreen} />
      <StackSetting.Screen name="SettingDetail" component={SettingsScreenDetail} />
    </StackSetting.Navigator>
  )
}
```

Uygulamada kullanılan Stack Navigatorların kodları.

```
const Tab = createBottomTabNavigator();
function TabNavigator() {
  return (
    <Tab.Navigator

    screenOptions={{ headerShown: false }}
     initialRouteName="Feed"
     activeColor="#e91e63"
     labelStyle={{ fontSize: 12 }}
     style={{ backgroundColor: 'tomato'}}
   >
     <Tab.Screen
       name="HomeStack"
       component={HomeStack}
       options={{
         tabBarLabel: 'Ana Sayfa',
         tabBarIcon: ({ color }) => (
           <MaterialCommunityIcons name="home" color={color} size={26} />
         ),
       }}
     />
     <Tab.Screen
       name="Settings"
       component={SettingStack}
       options={{
         tabBarLabel: 'Hazırlayanlar',
         tabBarIcon: ({ color }) => (
           <MaterialCommunityIcons name="cog" color={color} size={26} />
         ),
       }}
     />
   </Tab.Navigator>
  )
}
```

Uygulamada bulunnan Tab Screen’lerin kodları.

```
const Drawer = createDrawerNavigator();
function DrawerNavigator() {

  return (
    <Drawer.Navigator initialRouteName="MenuTab"  screenOptions={{ headerShown: false }}>

        <Drawer.Screen name="Ana Sayfa" component={TabNavigator} />
        <Drawer.Screen name="Hatırlatıcı Kur / Hatırlatıcılar" component={NotificationsScreen} />
        <Drawer.Screen name="Çıkış" component={SignOutScreen} />
    </Drawer.Navigator>
  )
}


const StackApp = createStackNavigator()
export default function App() {
  return (
    <SafeAreaView style={{flex: 1}}>
    <NativeBaseProvider>

    <NavigationContainer>
        <StackApp.Navigator initialRouteName="Login" screenOptions={{ headerShown: false }}>
          <StackApp.Screen name="HomeApp" component={DrawerNavigator}/>
          <StackApp.Screen name="Login" component={LoginScreen}/>
          <StackApp.Screen name="Register" component={RegisterScreen}/>
        </StackApp.Navigator>
    </NavigationContainer>
    </NativeBaseProvider>
  </SafeAreaView>

  );
}
```

Uygulamada bulunan Drawer Screen’in kodları.