FirebaseAuth – Authentication – React Native Firebase

Getting Started React Native Firebase : https://rnfirebase.io/

Authentication: https://rnfirebase.io/auth/usage

Auth Javascript: https://firebase.google.com/docs/reference/js/v8/firebase.auth.Auth#signinanonymously

1. FirebaseAuth App

```
> npx react-native@latest init FirebaseAuth
```

- navigation using `react-navigation` 6.x.x
- Firebase as backend for email auth using React Native Firebase
- Custom and reusable components
- Custom hook to **toggle password** field visibility on a TextInput
- handles server errors using Formik
- Login, Signup & Password Reset form built using Formik & yup
- show/hide Password Field's visibility 👁
- uses a custom Provider using Context API & Firebase's `onAuthStateChanged` handler to checks user's auth state with
- handles Forgot Password Reset using Firebase email method
- uses Expo Vector Icons
- uses KeyboardAwareScrollView package to handle keyboard appearance and automatically scrolls to focused TextInput
- uses `dotenv` and `expo-constants` packages to manage environment variables (so that they are not exposed on public repositories)
- all components are now functional components and use React Hooks

File Structure

FirebaseAuth : Link down Source

├── assets → □ All static assets, includes app logo

├── components → □ All re-suable UI components for form screens

│   └── Button.js → □ Custom Button component using Pressable, comes with two variants and handles opacity

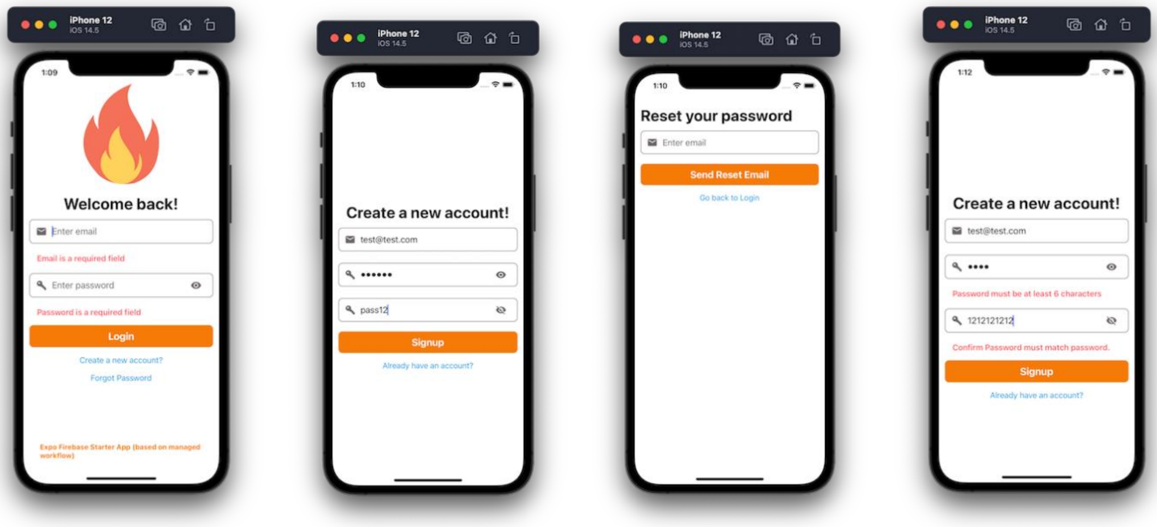
│   └── TextInput.js → □ Custom TextInput component that supports left and right cons

```
│    └── Icon.js → □ Icon component
│    └── FormErrorMessage.js → □ Component to display server errors from Firebase
│    └── LoadingIndicator.js → □ Loading indicator component
│    └── Logo.js → □ Logo component
│    └── View.js → □ Custom View component that supports safe area views
├── hooks → □ All custom hook components
│    └── useTogglePasswordVisibility.js → □ A custom hook that toggles password visibility on a TextInput component on a confirm password field
├── config → □ All configuration files
│    └── firebase.js → □ Configuration file to initialize firebase with firebaseConfig and auth
│    └── images.js → □ Require image assets, reusable values across the app
│    └── theme.js → □ Default set of colors, reusable values across the app
├── providers → □ All custom providers that use React Context API
│    └── AuthenticatedUserProvider.js → □ An Auth User Context component that shares Firebase user object when logged-in
├── navigation
│    └── AppStack.js → □ Protected routes such as Home screen
│    └── AuthStack.js → □ Routes such as Login screen, when the user is not authenticated
│    └── RootNavigator.js → □ Switch between Auth screens and App screens based on Firebase user logged-in state
├── screens
│    └── ForgotPassword.js → □ Forgot Password screen component
│    └── HomeScreen.js → □ Protected route/screen component
│    └── LoginScreen.js → □ Login screen component
│    └── SignupScreen.js → □ Signup screen component
```

├── App.js ⇥ □ Entry Point for Mobile apps, wrap all providers here

├── app.config.js ⇥ □ Expo config file

└── babel.config.js ⇥ □ Babel config (should be using `babel-preset-expo`)



Welcome back!

Enter email

Email is a required field

Enter password

Password is a required field

Login

Create a new account?

Forgot Password

Expo Firebase Starter App (based on managed workflow)

Create a new account!

test@test.com

••••••

pass12

Signup

Already have an account?

Reset your password

Enter email

Send Reset Email

Go back to Login

Create a new account!

test@test.com

••••

Password must be at least 6 characters

1212121212

Confirm Password must match password.

Signup

Already have an account?

1. Install via NPM

Pakage.json

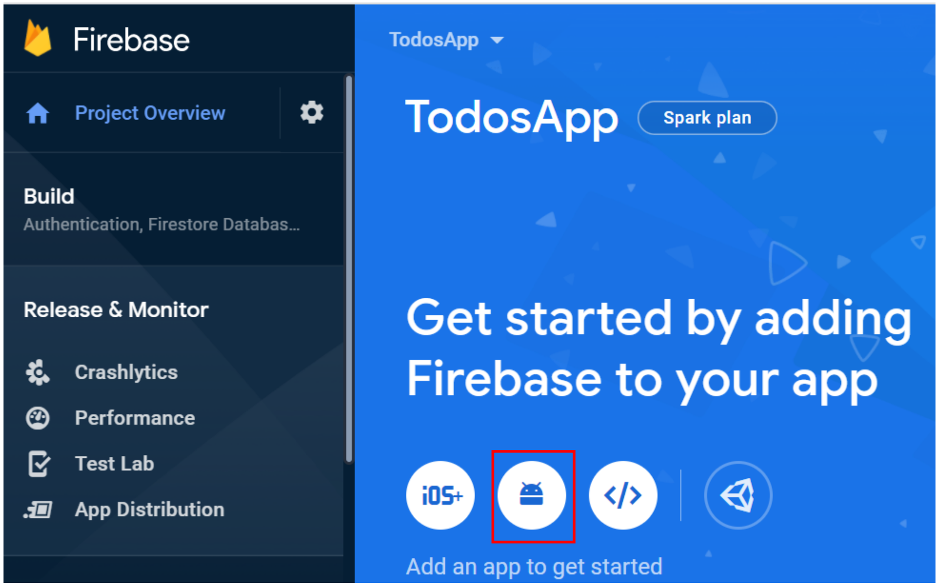
```
"dependencies": {
    ….
    "@expo/vector-icons": "12.0.5",

    "react-native-keyboard-aware-scroll-view":"0.9.5",
    "@react-native-firebase/app":"13.0.1",
    "@react-native-firebase/auth":"13.0.1",
    "formik":"2.2.9",
    "yup":"0.32.11",
    "@react-navigation/native":"6.0.6",
    "react-native-screens": "~3.8.0",
    "react-native-safe-area-context": "3.3.2",
    "@react-navigation/stack": "6.0.11",
    "react-native-gesture-handler": "~1.10.2"
    …..
  },
```

>npm install

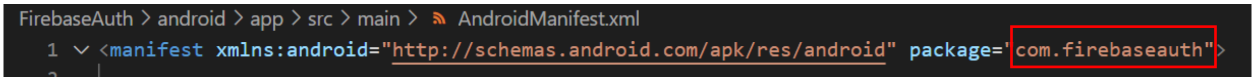
2. Android Setup

Firebase console: https://console.firebase.google.com/

Create Project: TodosApp -> Create app Android



`/android/app/src/main/AndroidManifest.xml`


FirebaseAuth > android > app > src > main > AndroidManifest.xml

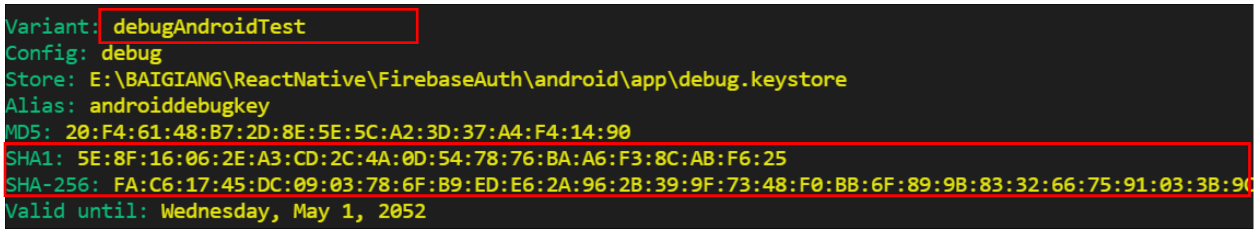
```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.firebaseauth">
```

Generate a certificate

> cd Android

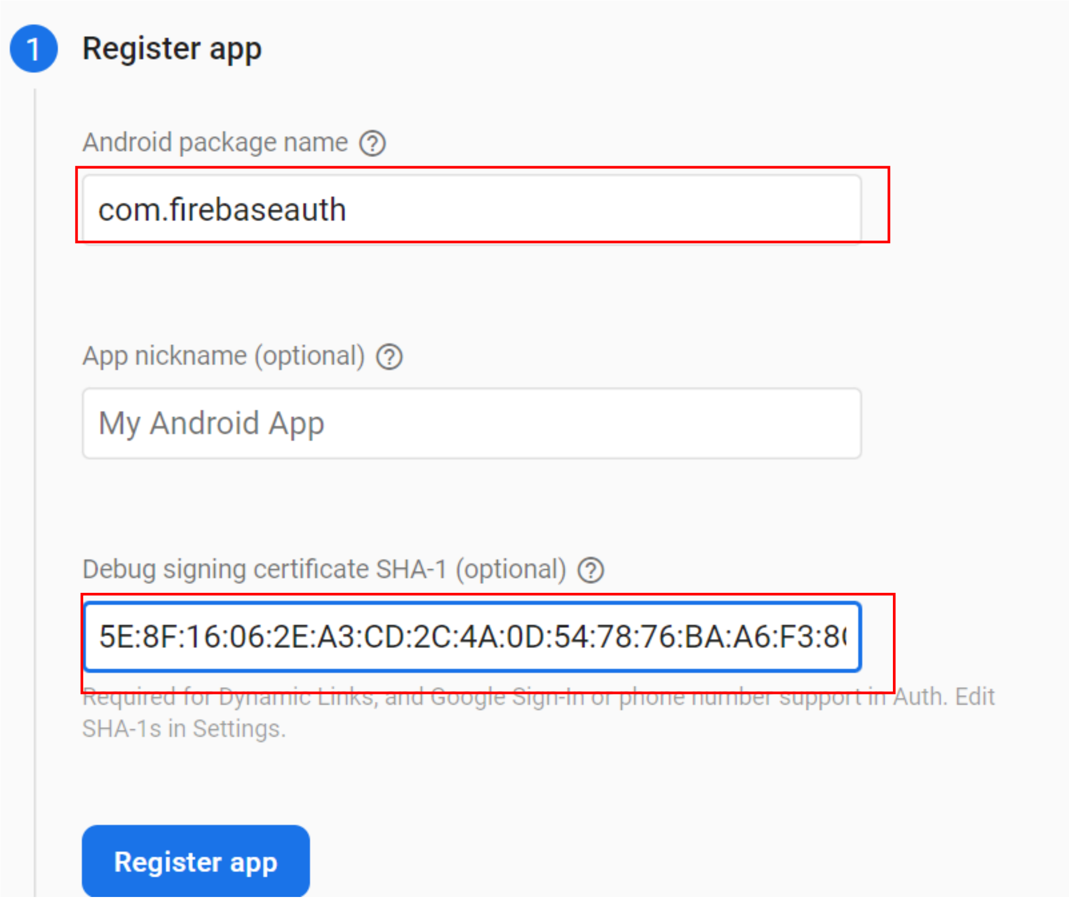
> ./gradlew signingReport

Copy both 'SHA1' and 'SHA-256' keys that belong to the 'debugAndroidTest' variant key option. Then, you can add those keys to the 'SHA certificate fingerprints' on your app in Firebase console.



```
Variant: debugAndroidTest
Config: debug
Store: E:\BAIGIANG\ReactNative\FirebaseAuth\android\app\debug.keystore
Alias: androiddebugkey
MD5: 20:F4:61:48:B7:2D:8E:5E:5C:A2:3D:37:A4:F4:14:90
SHA1: 5E:8F:16:06:2E:A3:CD:2C:4A:0D:54:78:76:BA:A6:F3:8C:AB:F6:25
SHA-256: FA:C6:17:45:DC:09:03:78:6F:B9:ED:E6:2A:96:2B:39:9F:73:48:F0:BB:6F:89:9B:83:32:66:75:91:03:3B:9C
Valid until: Wednesday, May 1, 2052
```

Create app Android



1 Register app

Android package name

com.firebaseauth

App nickname (optional)

My Android App

Debug signing certificate SHA-1 (optional)

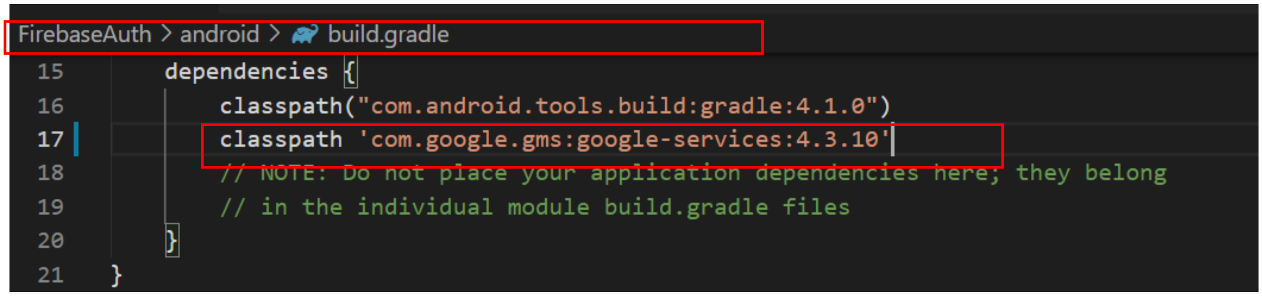
5E:8F:16:06:2E:A3:CD:2C:4A:0D:54:78:76:BA:A6:F3:8

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

Register app

Download the `google-services.json` file and place it inside of your project at the following location: `/android/app/google-services.json`

`Update ./android/build.gradle`

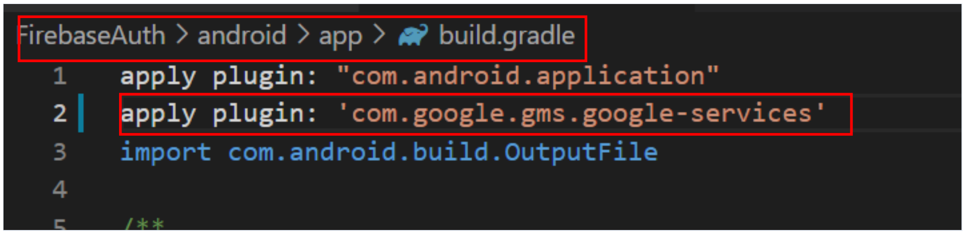


FirebaseAuth > android > build.gradle

```
    dependencies {
        classpath("com.android.tools.build:gradle:4.1.0")
        classpath 'com.google.gms:google-services:4.3.10'
        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
```

```
classpath 'com.google.gms:google-services:4.3.10'
```

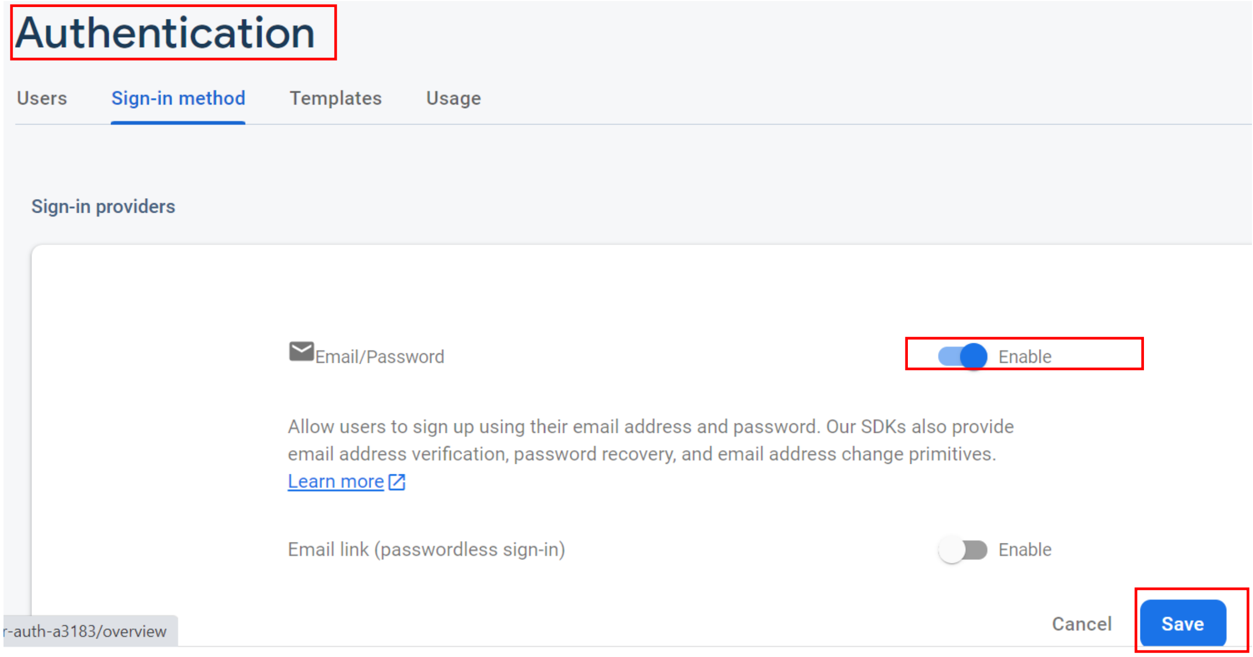
`Update ./android/app/build.gradle`



```
apply plugin: "com.android.application"
apply plugin: 'com.google.gms.google-services'
import com.android.build.OutputFile

/**
```

```
apply plugin: 'com.google.gms.google-services'
```

Create Sign-in providers with Email/Password

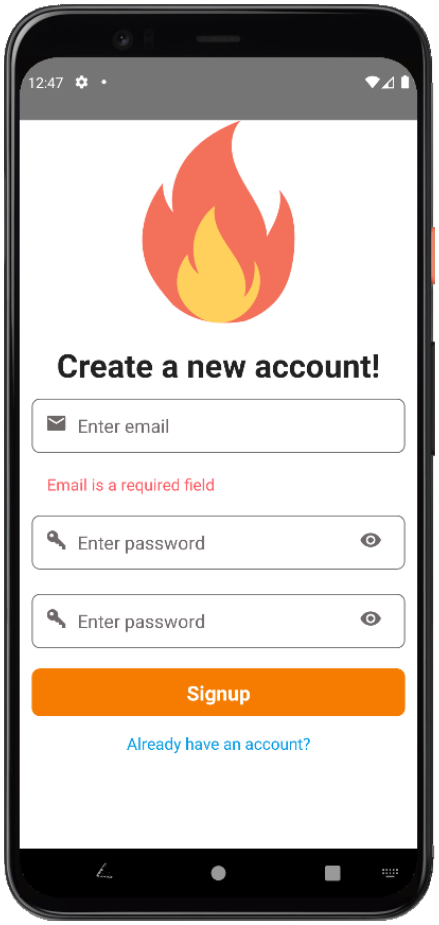


Email/Password ● Enable

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery, and email address change primitives. Learn more

Email link (passwordless sign-in) ○ Enable

5. **SignupScreen.js**

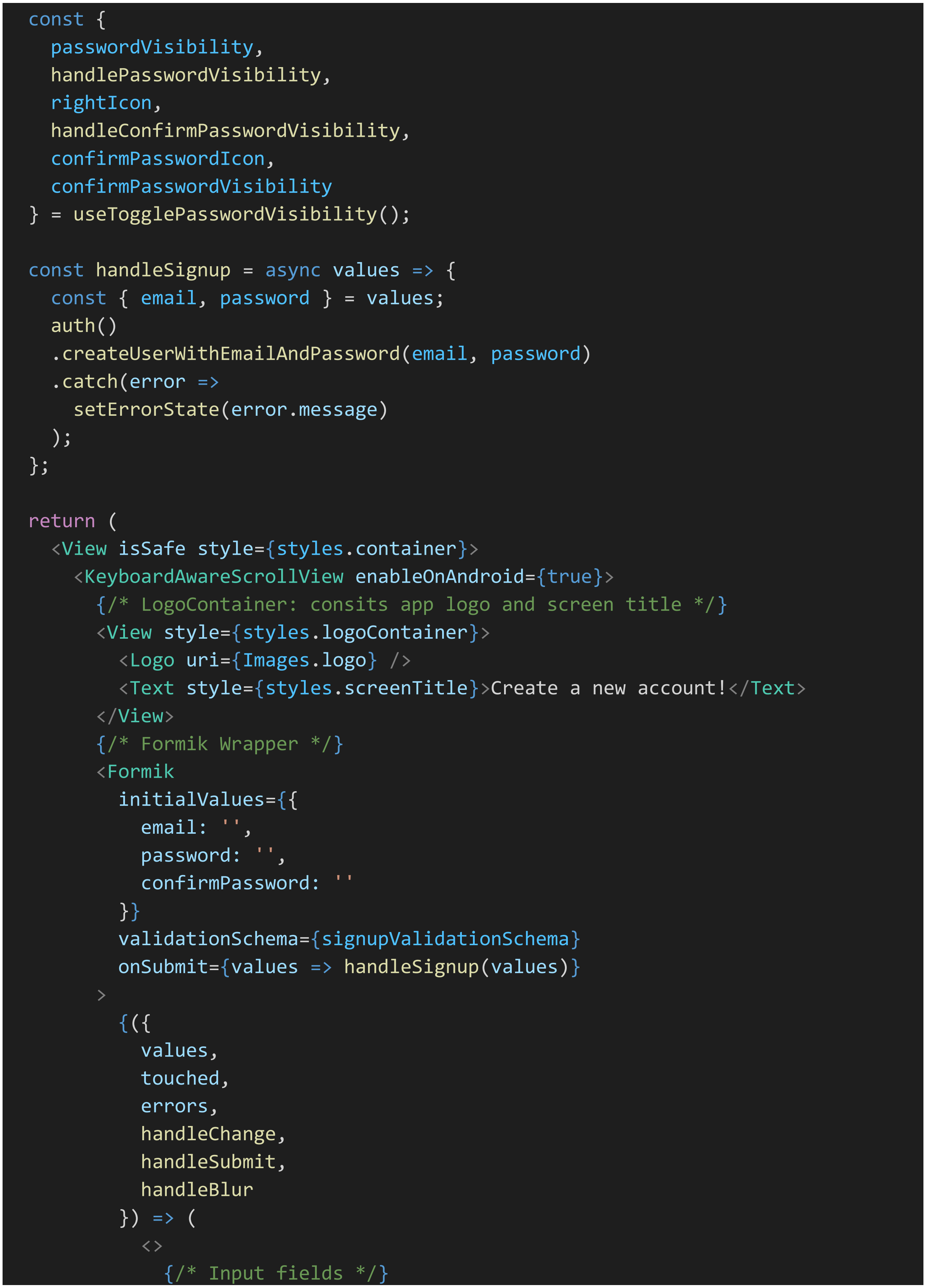


Create a new account!

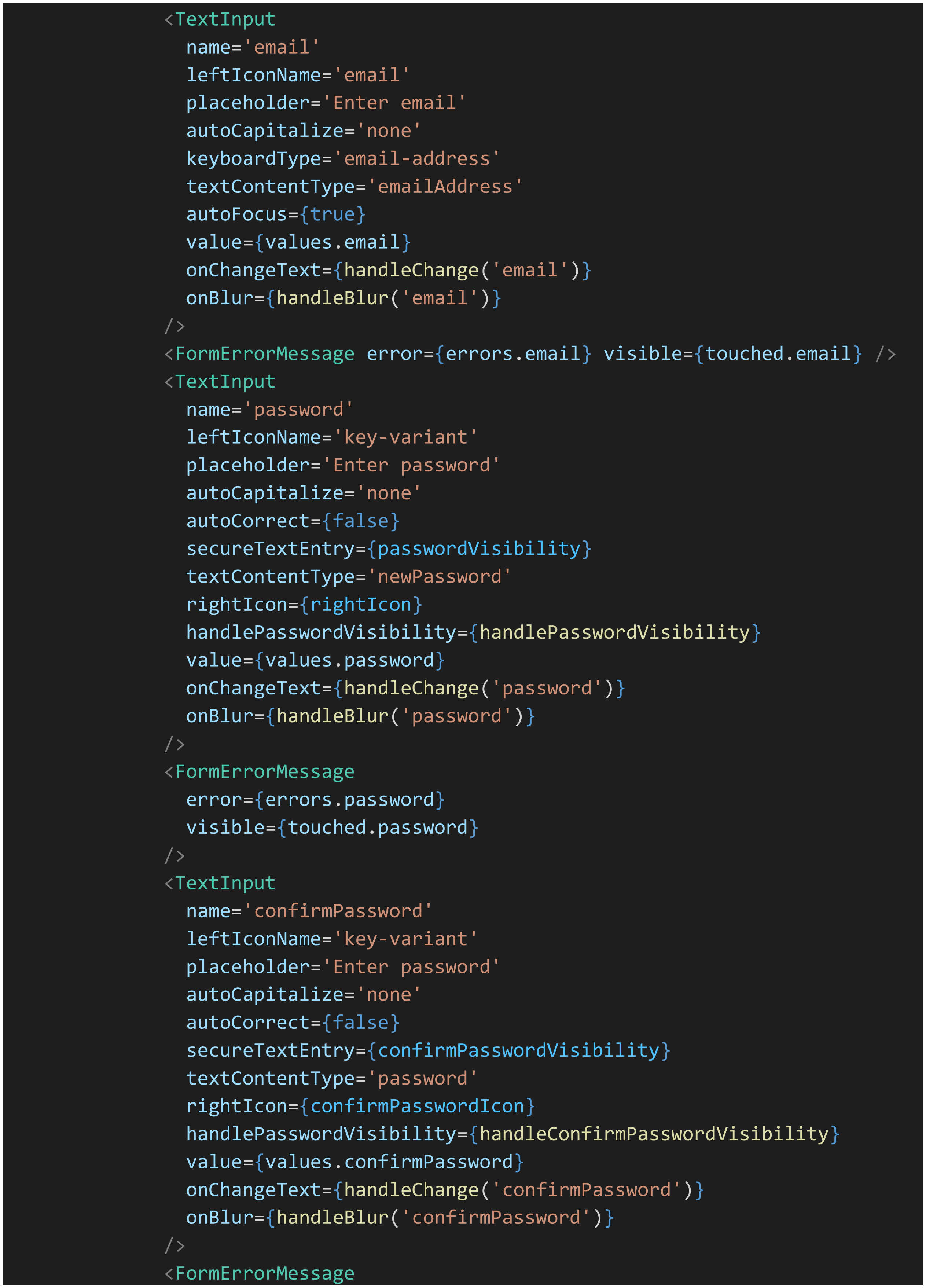
Email is a required field

./screens/SignupScreen.js

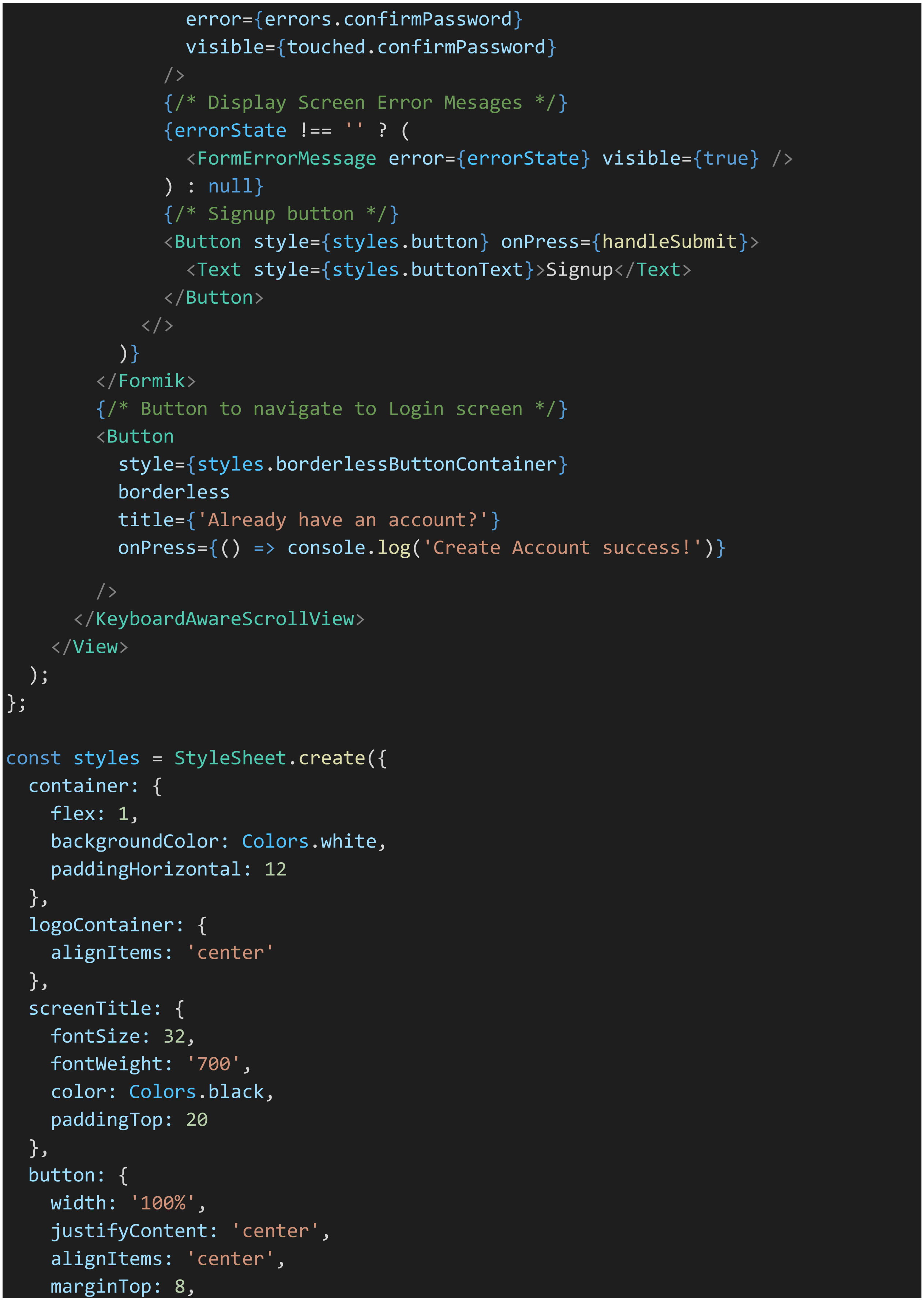
```
import React, { useState } from 'react';
import { Text, StyleSheet } from 'react-native';
import { Formik } from 'formik';
import auth from '@react-native-firebase/auth';
import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-view';

import { View, TextInput, Logo, Button, FormErrorMessage } from '../components';
import { Images, Colors} from '../config';
import { useTogglePasswordVisibility } from '../hooks';
import { signupValidationSchema } from '../utils';

export const SignupScreen = () => {
  const [errorState, setErrorState] = useState('');
```



```
const {
  passwordVisibility,
  handlePasswordVisibility,
  rightIcon,
  handleConfirmPasswordVisibility,
  confirmPasswordIcon,
  confirmPasswordVisibility
} = useTogglePasswordVisibility();

const handleSignup = async values => {
  const { email, password } = values;
  auth()
  .createUserWithEmailAndPassword(email, password)
  .catch(error =>
    setErrorState(error.message)
  );
};

return (
  <View isSafe style={styles.container}>
    <KeyboardAwareScrollView enableOnAndroid={true}>
      {/* LogoContainer: consits app logo and screen title */}
      <View style={styles.logoContainer}>
        <Logo uri={Images.logo} />
        <Text style={styles.screenTitle}>Create a new account!</Text>
      </View>
      {/* Formik Wrapper */}
      <Formik
        initialValues={{
          email: '',
          password: '',
          confirmPassword: ''
        }}
        validationSchema={signupValidationSchema}
        onSubmit={values => handleSignup(values)}
      >
        {({
          values,
          touched,
          errors,
          handleChange,
          handleSubmit,
          handleBlur
        }) => (
          <>
            {/* Input fields */}
```



```
              <TextInput
                name='email'
                leftIconName='email'
                placeholder='Enter email'
                autoCapitalize='none'
                keyboardType='email-address'
                textContentType='emailAddress'
                autoFocus={true}
                value={values.email}
                onChangeText={handleChange('email')}
                onBlur={handleBlur('email')}
              />
              <FormErrorMessage error={errors.email} visible={touched.email} />
              <TextInput
                name='password'
                leftIconName='key-variant'
                placeholder='Enter password'
                autoCapitalize='none'
                autoCorrect={false}
                secureTextEntry={passwordVisibility}
                textContentType='newPassword'
                rightIcon={rightIcon}
                handlePasswordVisibility={handlePasswordVisibility}
                value={values.password}
                onChangeText={handleChange('password')}
                onBlur={handleBlur('password')}
              />
              <FormErrorMessage
                error={errors.password}
                visible={touched.password}
              />
              <TextInput
                name='confirmPassword'
                leftIconName='key-variant'
                placeholder='Enter password'
                autoCapitalize='none'
                autoCorrect={false}
                secureTextEntry={confirmPasswordVisibility}
                textContentType='password'
                rightIcon={confirmPasswordIcon}
                handlePasswordVisibility={handleConfirmPasswordVisibility}
                value={values.confirmPassword}
                onChangeText={handleChange('confirmPassword')}
                onBlur={handleBlur('confirmPassword')}
              />
              <FormErrorMessage
```



```
                error={errors.confirmPassword}
                visible={touched.confirmPassword}
              />
              {/* Display Screen Error Mesages */}
              {errorState !== '' ? (
                <FormErrorMessage error={errorState} visible={true} />
              ) : null}
              {/* Signup button */}
              <Button style={styles.button} onPress={handleSubmit}>
                <Text style={styles.buttonText}>Signup</Text>
              </Button>
            </>
          )}
        </Formik>
        {/* Button to navigate to Login screen */}
        <Button
          style={styles.borderlessButtonContainer}
          borderless
          title={'Already have an account?'}
          onPress={() => console.log('Create Account success!')}

        />
      </KeyboardAwareScrollView>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: Colors.white,
    paddingHorizontal: 12
  },
  logoContainer: {
    alignItems: 'center'
  },
  screenTitle: {
    fontSize: 32,
    fontWeight: '700',
    color: Colors.black,
    paddingTop: 20
  },
  button: {
    width: '100%',
    justifyContent: 'center',
    alignItems: 'center',
    marginTop: 8,
    backgroundColor: Colors.orange,
    padding: 10,
    borderRadius: 8
  },
  buttonText: {
    fontSize: 20,
    color: Colors.white,
    fontWeight: '700'
  },
  borderlessButtonContainer: {
    marginTop: 16,
    alignItems: 'center',
    justifyContent: 'center'
  }
});
```

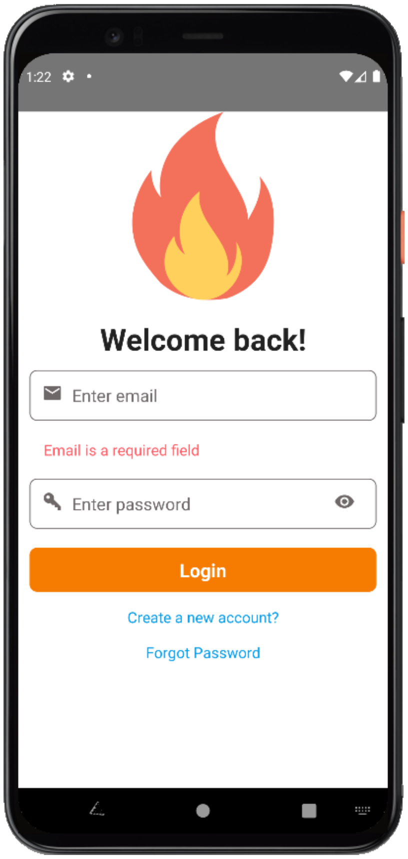
Create ./screens/index.js

```
import { SignupScreen } from './SignupScreen';
export { SignupScreen};
```

Test **SignupScreen.js => Cập nhật App.js**

```
import React from 'react';
import { SafeAreaProvider } from 'react-native-safe-area-context';
import { SignupScreen} from './screens';

const App = () => {
  return (
      <SafeAreaProvider>
        <SignupScreen />
      </SafeAreaProvider>
  );
};

export default App;
```

./screens/LoginScreen.js



1:22

Welcome back!

Enter email

Email is a required field

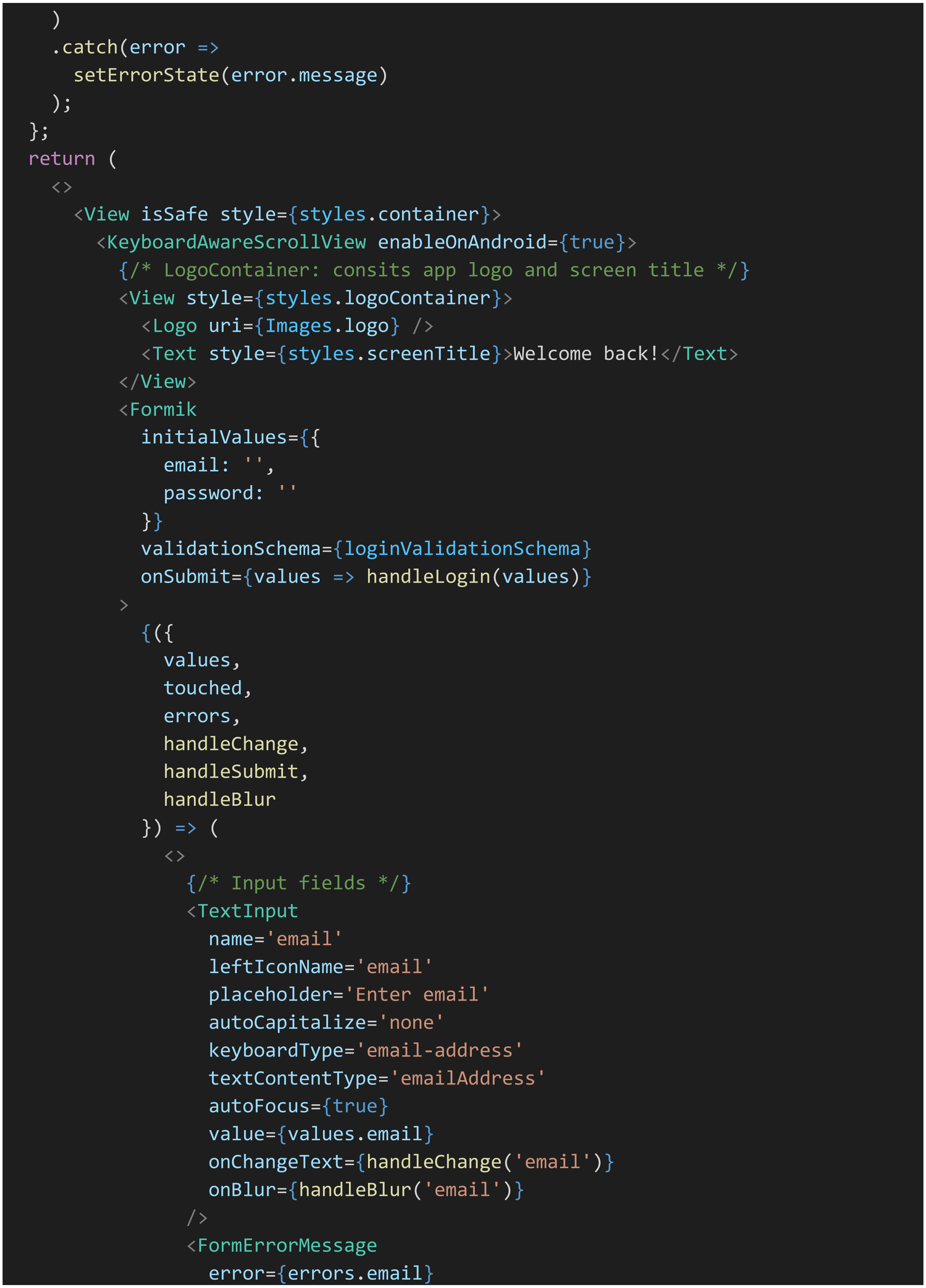
Enter password

Login

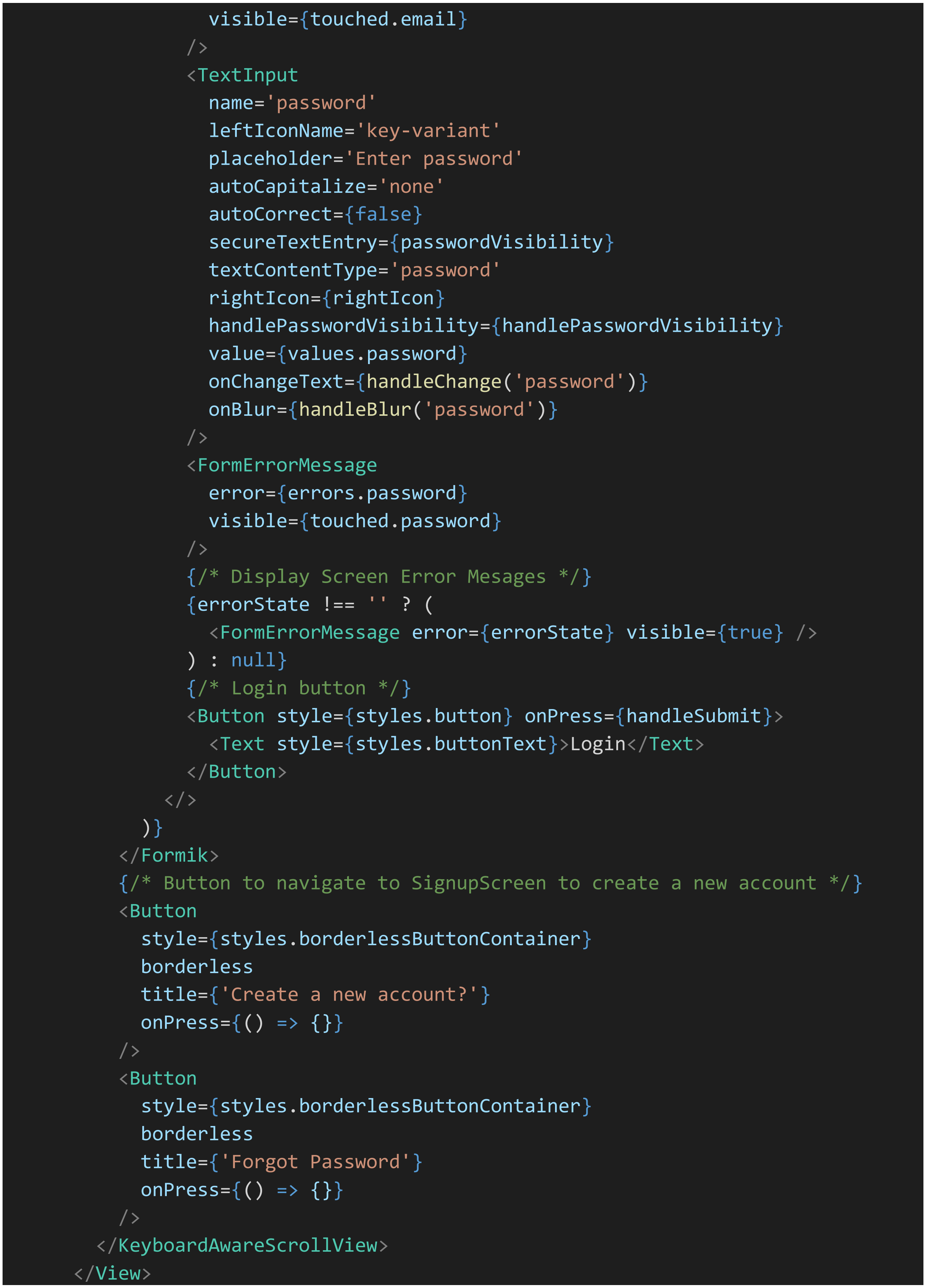
Create a new account?

Forgot Password

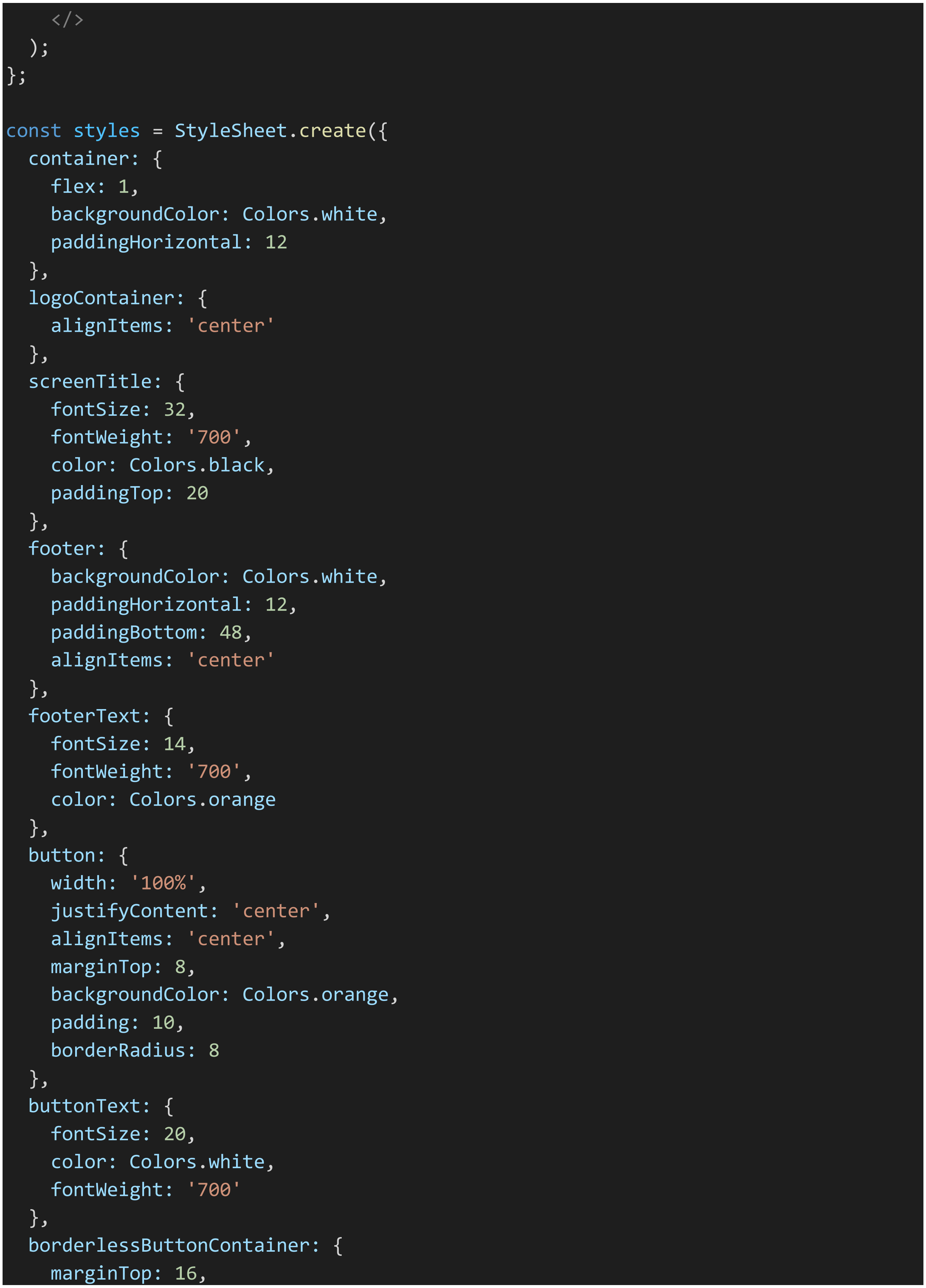
```
import React, { useState } from 'react';
import { Text, StyleSheet } from 'react-native';
import { Formik } from 'formik';
import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-
view';
import auth from '@react-native-firebase/auth';
import { View, TextInput, Logo, Button, FormErrorMessage } from '../components';
import { Images, Colors} from '../config';
import { useTogglePasswordVisibility } from '../hooks';
import { loginValidationSchema } from '../utils';

export const LoginScreen = () => {
  const [errorState, setErrorState] = useState('');
  const { passwordVisibility, handlePasswordVisibility, rightIcon } =
    useTogglePasswordVisibility();

  const handleLogin = (values) => {
    const { email, password } = values;
    auth().signInWithEmailAndPassword(email, password)
    .then(
      ({user})=> console.log(user)
```



```
    )
    .catch(error =>
      setErrorState(error.message)
    );
  };
  return (
    <>
      <View isSafe style={styles.container}>
        <KeyboardAwareScrollView enableOnAndroid={true}>
          {/* LogoContainer: consits app logo and screen title */}
          <View style={styles.logoContainer}>
            <Logo uri={Images.logo} />
            <Text style={styles.screenTitle}>Welcome back!</Text>
          </View>
          <Formik
            initialValues={{
              email: '',
              password: ''
            }}
            validationSchema={loginValidationSchema}
            onSubmit={values => handleLogin(values)}
          >
            {({
              values,
              touched,
              errors,
              handleChange,
              handleSubmit,
              handleBlur
            }) => (
              <>
                {/* Input fields */}
                <TextInput
                  name='email'
                  leftIconName='email'
                  placeholder='Enter email'
                  autoCapitalize='none'
                  keyboardType='email-address'
                  textContentType='emailAddress'
                  autoFocus={true}
                  value={values.email}
                  onChangeText={handleChange('email')}
                  onBlur={handleBlur('email')}
                />
                <FormErrorMessage
                  error={errors.email}
```



```
                visible={touched.email}
              />
              <TextInput
                name='password'
                leftIconName='key-variant'
                placeholder='Enter password'
                autoCapitalize='none'
                autoCorrect={false}
                secureTextEntry={passwordVisibility}
                textContentType='password'
                rightIcon={rightIcon}
                handlePasswordVisibility={handlePasswordVisibility}
                value={values.password}
                onChangeText={handleChange('password')}
                onBlur={handleBlur('password')}
              />
              <FormErrorMessage
                error={errors.password}
                visible={touched.password}
              />
              {/* Display Screen Error Mesages */}
              {errorState !== '' ? (
                <FormErrorMessage error={errorState} visible={true} />
              ) : null}
              {/* Login button */}
              <Button style={styles.button} onPress={handleSubmit}>
                <Text style={styles.buttonText}>Login</Text>
              </Button>
            </>
          )}
        </Formik>
        {/* Button to navigate to SignupScreen to create a new account */}
        <Button
          style={styles.borderlessButtonContainer}
          borderless
          title={'Create a new account?'}
          onPress={() => {}}
        />
        <Button
          style={styles.borderlessButtonContainer}
          borderless
          title={'Forgot Password'}
          onPress={() => {}}
        />
      </KeyboardAwareScrollView>
    </View>
```



```
    </>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: Colors.white,
    paddingHorizontal: 12
  },
  logoContainer: {
    alignItems: 'center'
  },
  screenTitle: {
    fontSize: 32,
    fontWeight: '700',
    color: Colors.black,
    paddingTop: 20
  },
  footer: {
    backgroundColor: Colors.white,
    paddingHorizontal: 12,
    paddingBottom: 48,
    alignItems: 'center'
  },
  footerText: {
    fontSize: 14,
    fontWeight: '700',
    color: Colors.orange
  },
  button: {
    width: '100%',
    justifyContent: 'center',
    alignItems: 'center',
    marginTop: 8,
    backgroundColor: Colors.orange,
    padding: 10,
    borderRadius: 8
  },
  buttonText: {
    fontSize: 20,
    color: Colors.white,
    fontWeight: '700'
  },
  borderlessButtonContainer: {
    marginTop: 16,
    alignItems: 'center',
    justifyContent: 'center'
  }
});
```

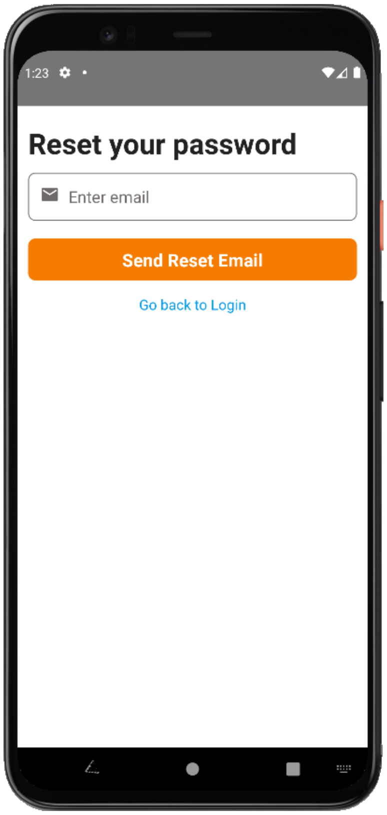
Update ./screens/index.js

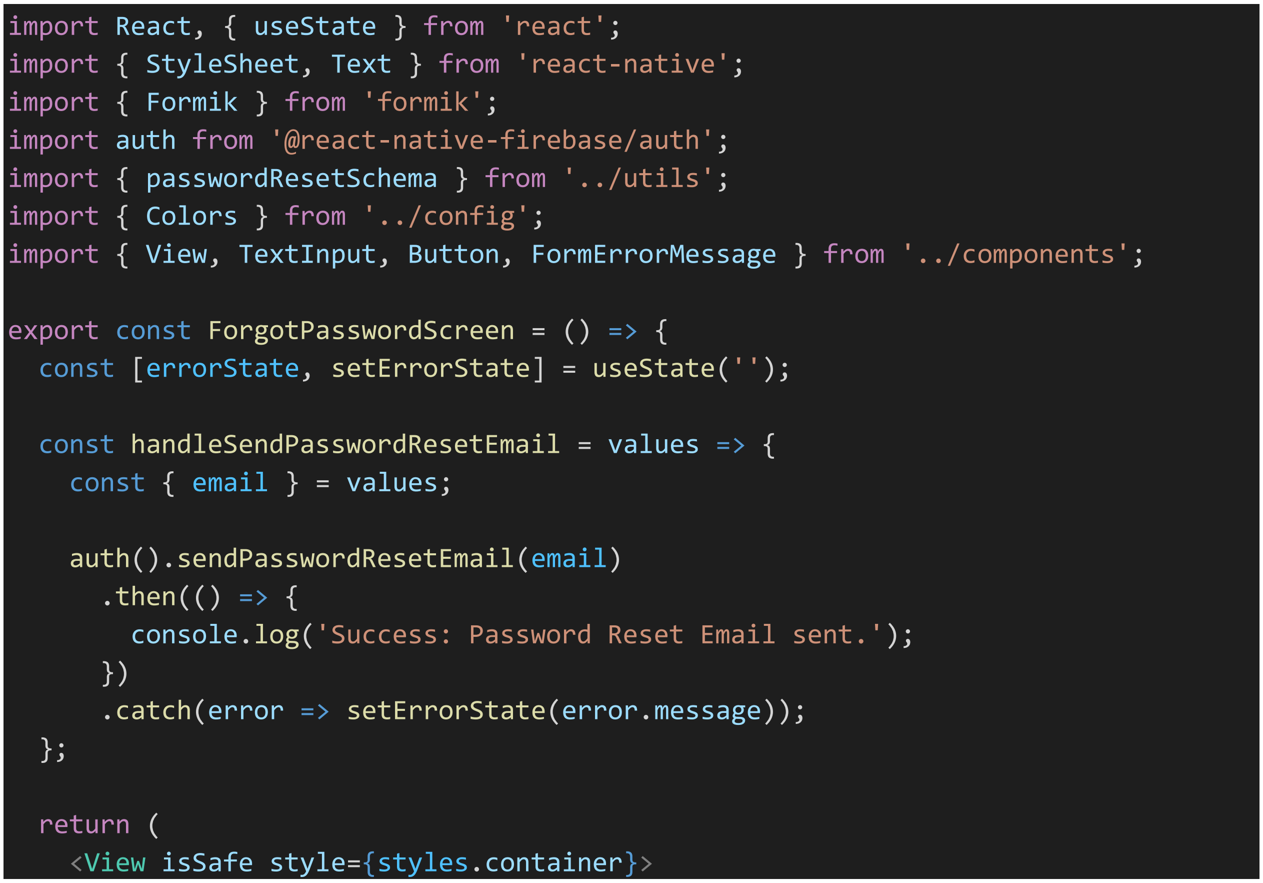
```
import { LoginScreen } from './LoginScreen';
import { SignupScreen } from './SignupScreen';

export {LoginScreen, SignupScreen};
```

Test LoginScreen.js => Update App.js

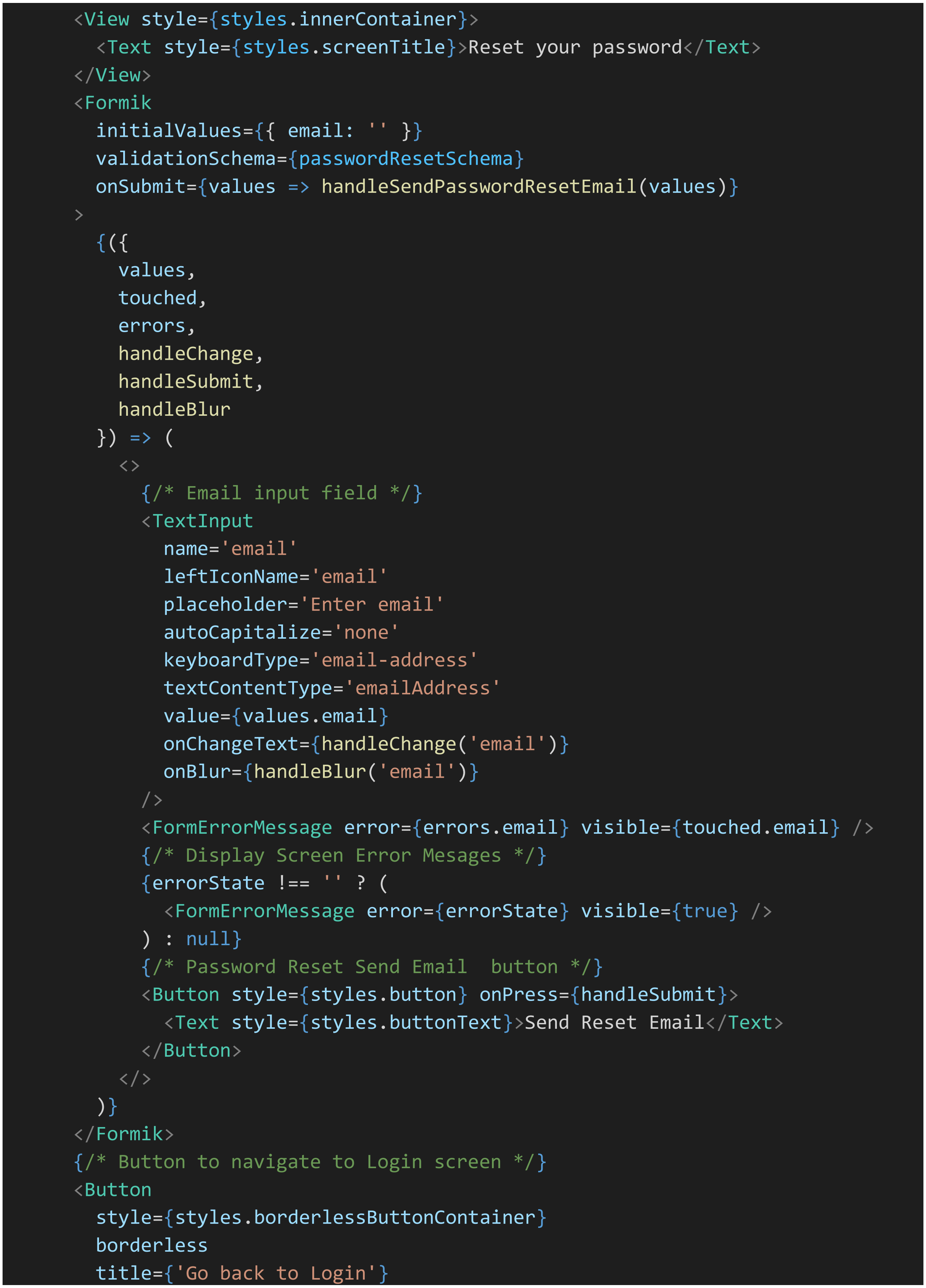
```
import React from 'react';
import { SafeAreaProvider } from 'react-native-safe-area-context';
import {LoginScreen, SignupScreen} from './screens';
const App = () => {
  return (

      <SafeAreaProvider>
        <LoginScreen />
      </SafeAreaProvider>
  );
};

export default App;
```

Create ./screens/ForgotPasswordScreen.js





```
import React, { useState } from 'react';
import { StyleSheet, Text } from 'react-native';
import { Formik } from 'formik';
import auth from '@react-native-firebase/auth';
import { passwordResetSchema } from '../utils';
import { Colors } from '../config';
import { View, TextInput, Button, FormErrorMessage } from '../components';

export const ForgotPasswordScreen = () => {
  const [errorState, setErrorState] = useState('');

  const handleSendPasswordResetEmail = values => {
    const { email } = values;

    auth().sendPasswordResetEmail(email)
      .then(() => {
        console.log('Success: Password Reset Email sent.');
      })
      .catch(error => setErrorState(error.message));
  };

  return (
    <View isSafe style={styles.container}>
```



```
      <View style={styles.innerContainer}>
        <Text style={styles.screenTitle}>Reset your password</Text>
      </View>
      <Formik
        initialValues={{ email: '' }}
        validationSchema={passwordResetSchema}
        onSubmit={values => handleSendPasswordResetEmail(values)}
      >
        {({
          values,
          touched,
          errors,
          handleChange,
          handleSubmit,
          handleBlur
        }) => (
          <>
            {/* Email input field */}
            <TextInput
              name='email'
              leftIconName='email'
              placeholder='Enter email'
              autoCapitalize='none'
              keyboardType='email-address'
              textContentType='emailAddress'
              value={values.email}
              onChangeText={handleChange('email')}
              onBlur={handleBlur('email')}
            />
            <FormErrorMessage error={errors.email} visible={touched.email} />
            {/* Display Screen Error Mesages */}
            {errorState !== '' ? (
              <FormErrorMessage error={errorState} visible={true} />
            ) : null}
            {/* Password Reset Send Email  button */}
            <Button style={styles.button} onPress={handleSubmit}>
              <Text style={styles.buttonText}>Send Reset Email</Text>
            </Button>
          </>
        )}
      </Formik>
      {/* Button to navigate to Login screen */}
      <Button
        style={styles.borderlessButtonContainer}
        borderless
        title={'Go back to Login'}
        onPress={() => {} }
      />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: Colors.white,
    paddingHorizontal: 12
  },
  innercontainer: {
    alignItems: 'center'
  },
  screenTitle: {
    fontSize: 32,
    fontWeight: '700',
    color: Colors.black,
    paddingTop: 20
  },
  button: {
    width: '100%',
    justifyContent: 'center',
    alignItems: 'center',
    marginTop: 8,
    backgroundColor: Colors.orange,
    padding: 10,
    borderRadius: 8
  },
  buttonText: {
    fontSize: 20,
    color: Colors.white,
    fontWeight: '700'
  },
  borderlessButtonContainer: {
    marginTop: 16,
    alignItems: 'center',
    justifyContent: 'center'
  }
});
```

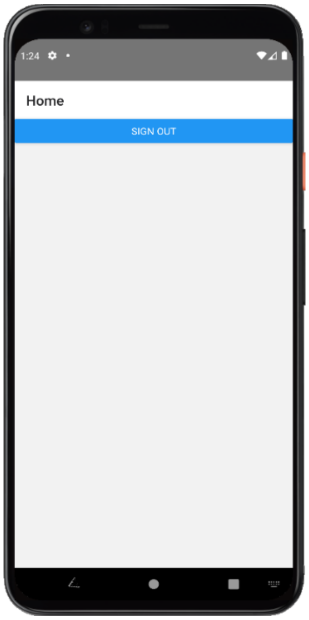
Update ./screens/index.js

```
import { LoginScreen } from './LoginScreen';
import { SignupScreen } from './SignupScreen';
import { ForgotPasswordScreen } from './ForgotPasswordScreen';

export { ForgotPasswordScreen ,LoginScreen, SignupScreen};
```

Test LoginScreen.js => Update App.js

```
import React from 'react';
import { SafeAreaProvider } from 'react-native-safe-area-context';
import { ForgotPasswordScreen ,LoginScreen, SignupScreen} from './screens';
const App = () => {
  return (

      <SafeAreaProvider>
        <ForgotPasswordScreen />
      </SafeAreaProvider>
  );
};

export default App;
```

Create ./screens/HomeScreen.js



```
import React from 'react';
import { View, StyleSheet, Button } from 'react-native';
import auth from '@react-native-firebase/auth';
export const HomeScreen = () => {
  const handleLogout = () => {
    auth().signOut().catch(error => console.log('Error logging out: ', error));
  };
  return (
    <View style={styles.container}>
      <Button title='Sign Out' onPress={handleLogout} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1
  }
});
```

Test tương tự cho **HomeScreen.js**

Sử dụng Context để share state **user** cho các screen

Create ./providers/ AuthenticatedUserProvider.js

```
import React, { useState, createContext } from 'react';

export const AuthenticatedUserContext = createContext({});

export const AuthenticatedUserProvider = ({ children }) => {
  const [user, setUser] = useState(null);

  return (
    <AuthenticatedUserContext.Provider value={{ user, setUser }}>
      {children}
    </AuthenticatedUserContext.Provider>
  );
};
```

Create ./providers/index.js

```
import {
  AuthenticatedUserContext,
  AuthenticatedUserProvider
} from './AuthenticatedUserProvider';
```

```
export { AuthenticatedUserContext, AuthenticatedUserProvider };
```

Update App.js

```
….
import { AuthenticatedUserProvider } from './providers';

const App = () => {
  return (
    <AuthenticatedUserProvider>
      <SafeAreaProvider>
        <ForgotPasswordScreen/>
      </SafeAreaProvider>
    </AuthenticatedUserProvider>
  );
};
```

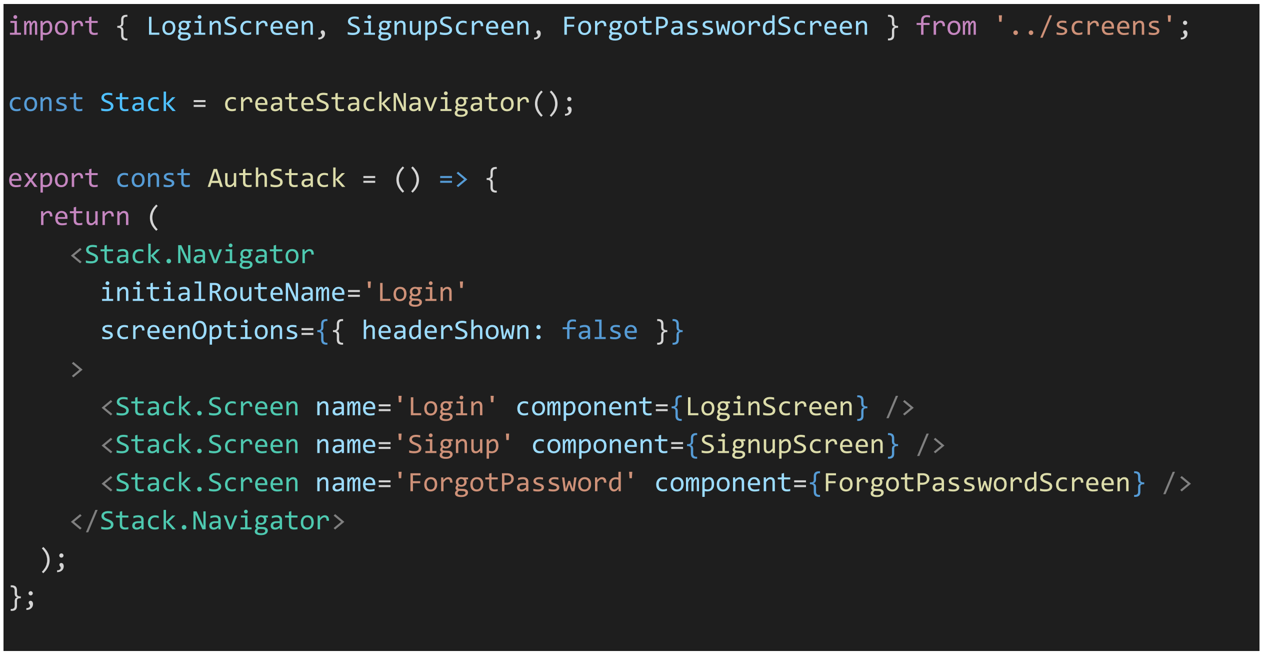
Create folder navigation

Create ./navigation/AppStack.js

```
import * as React from 'react';
import { createStackNavigator } from '@react-navigation/stack';
import { AuthenticatedUserContext } from '../providers';
import { HomeScreen } from '../screens';

const Stack = createStackNavigator();

export const AppStack = () => {
  return (
    <Stack.Navigator>
      <Stack.Screen name='Home' component={HomeScreen} />
    </Stack.Navigator>
  );
};
```

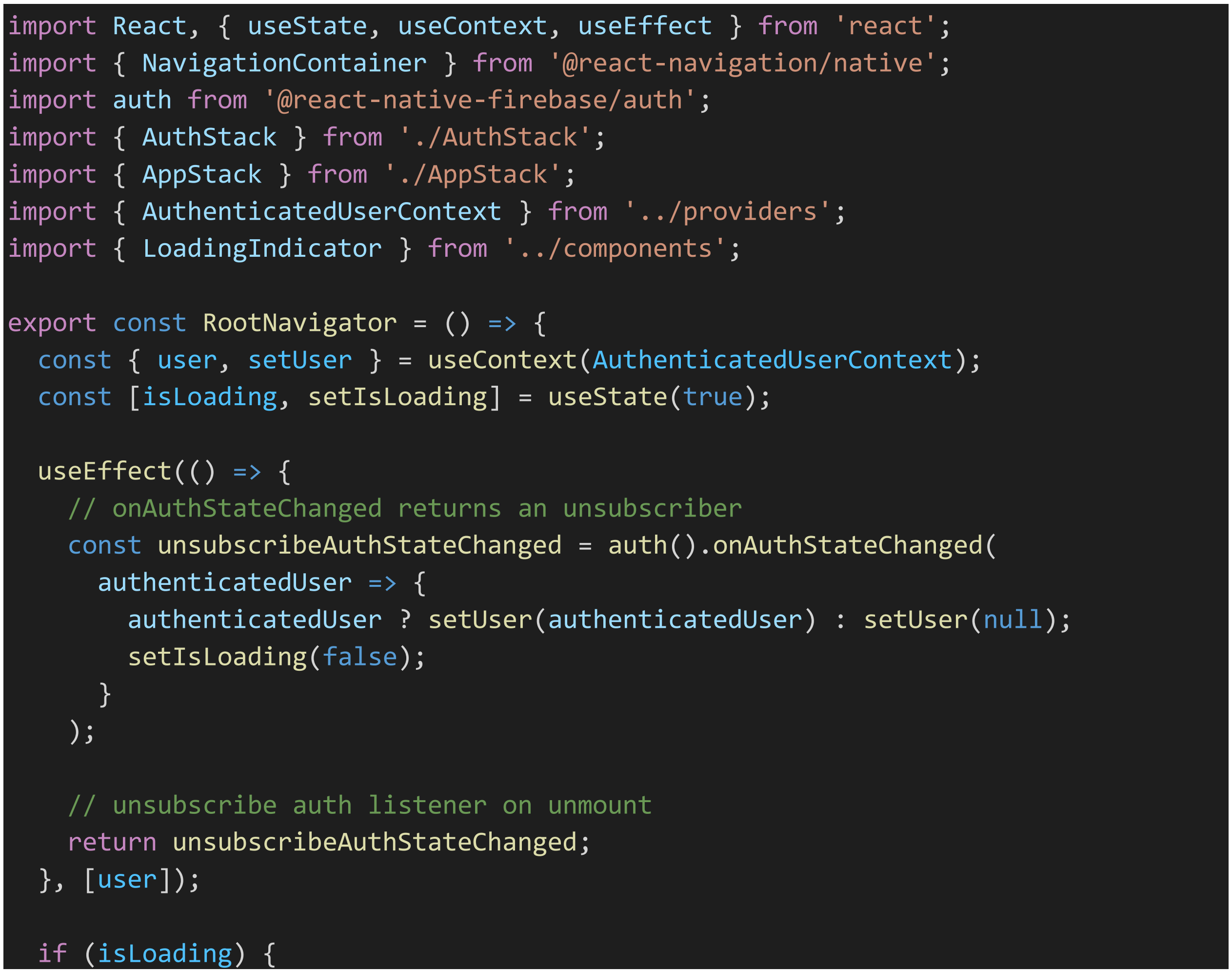
Create ./navigation/AuthStack.js

```
import * as React from 'react';
import { createStackNavigator } from '@react-navigation/stack';
```



```
import { LoginScreen, SignupScreen, ForgotPasswordScreen } from '../screens';

const Stack = createStackNavigator();

export const AuthStack = () => {
  return (
    <Stack.Navigator
      initialRouteName='Login'
      screenOptions={{ headerShown: false }}
    >
      <Stack.Screen name='Login' component={LoginScreen} />
      <Stack.Screen name='Signup' component={SignupScreen} />
      <Stack.Screen name='ForgotPassword' component={ForgotPasswordScreen} />
    </Stack.Navigator>
  );
};
```

Create ./navigation/RootNavigator.js



```
import React, { useState, useContext, useEffect } from 'react';
import { NavigationContainer } from '@react-navigation/native';
import auth from '@react-native-firebase/auth';
import { AuthStack } from './AuthStack';
import { AppStack } from './AppStack';
import { AuthenticatedUserContext } from '../providers';
import { LoadingIndicator } from '../components';

export const RootNavigator = () => {
  const { user, setUser } = useContext(AuthenticatedUserContext);
  const [isLoading, setIsLoading] = useState(true);

  useEffect(() => {
    // onAuthStateChanged returns an unsubscriber
    const unsubscribeAuthStateChanged = auth().onAuthStateChanged(
      authenticatedUser => {
        authenticatedUser ? setUser(authenticatedUser) : setUser(null);
        setIsLoading(false);
      }
    );

    // unsubscribe auth listener on unmount
    return unsubscribeAuthStateChanged;
  }, [user]);

  if (isLoading) {
    return <LoadingIndicator />;
  }

  return (
    <NavigationContainer>
      {user ? <AppStack /> : <AuthStack />}
    </NavigationContainer>
  );
};
```

Create ./navigation/index.js

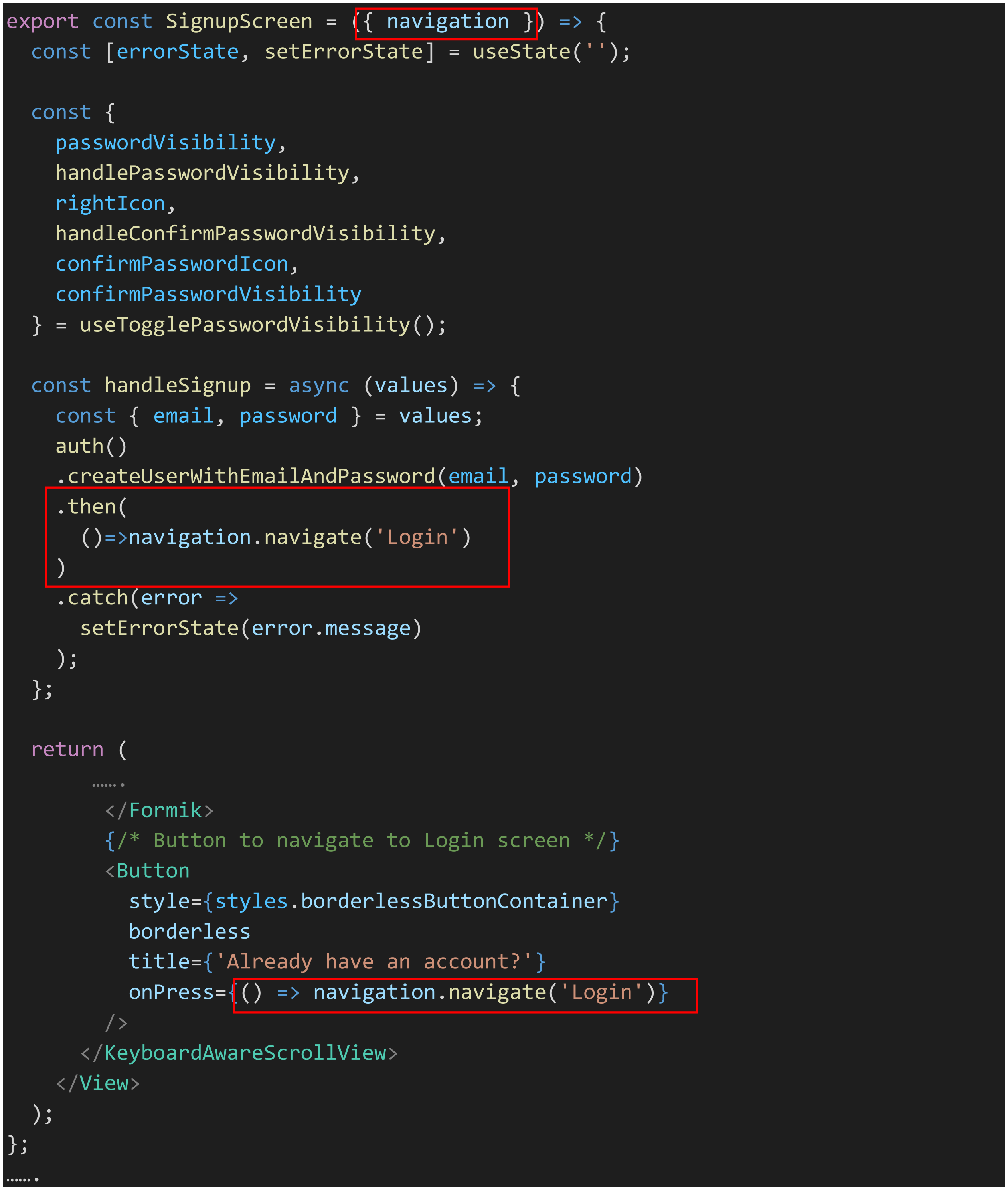
```
import { RootNavigator } from './RootNavigator';

export { RootNavigator };
```

Cập nhật App.js

```
import React from 'react';
import { SafeAreaProvider } from 'react-native-safe-area-context';

import {HomeScreen ,ForgotPasswordScreen, LoginScreen, SignupScreen} from
'./screens';
import { RootNavigator } from './navigation/RootNavigator';
import { AuthenticatedUserProvider } from './providers';

const App = () => {
  return (
    <AuthenticatedUserProvider>
      <SafeAreaProvider>
        <RootNavigator />
      </SafeAreaProvider>
    </AuthenticatedUserProvider>
  );
};

export default App;
```

Cập nhật navigation cho các màn hình

./screens/SignupScreen.js



```
export const SignupScreen = ({ navigation }) => {
  const [errorState, setErrorState] = useState('');

  const {
    passwordVisibility,
    handlePasswordVisibility,
    rightIcon,
    handleConfirmPasswordVisibility,
    confirmPasswordIcon,
    confirmPasswordVisibility
  } = useTogglePasswordVisibility();

  const handleSignup = async (values) => {
    const { email, password } = values;
    auth()
    .createUserWithEmailAndPassword(email, password)
    .then(
      ()=>navigation.navigate('Login')
    )
    .catch(error =>
      setErrorState(error.message)
    );
  };

  return (
       …….
        </Formik>
        {/* Button to navigate to Login screen */}
        <Button
          style={styles.borderlessButtonContainer}
          borderless
          title={'Already have an account?'}
          onPress={() => navigation.navigate('Login')}
        />
      </KeyboardAwareScrollView>
    </View>
  );
};
…….
```

Tương tự cập nhật cho LoginScreen.js, ForgotPasswordScreen.js