

# Lesson 8: Becoming a React Native native

Auth, Text Input, Pages and more

---

Karl Steven Velasco Orjalo

May 31, 2023

React Native Quickstart

# Lesson Outline

1. Firebase Auth
2. Making nested pages

# Firestore Auth

---

# Handling user sign-ins

To handle user authentication as well as Sign-up/in, we'll just use Firebase's Authentication service.

Go to to the Firebase console and find 'Authentication' under the Build tab on the left. Create an authentication service with **Email/Password**.

## Handling user sign-ins (cont.)

In our app, we have to have at least 2 inputs, an email and a password as well as a Sign-up button.

In `./src/pages/Account.jsx` import `TextInput` and `Pressable`.

```
import { View, FlatList, Text, TextInput, Pressable } from 'react-native'
```

## Handling user sign-ins (cont.)

Next we can use the `useState()` hook to store our email and password.

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

## Handling user sign-ins (cont.)

We'll need some functions from `firebase/auth` so import them.

```
import { getAuth, createUserWithEmailAndPassword } from "firebase/auth";
```

As you can probably guess, we now have a function to check the current auth-ed user, create a new account and sign-in.

## Handling user sign-ins (cont.)

Now make two pairs of `<Text />` and `<TextInput />`, one for email and one for password.

```
<Text>Email</Text>
<TextInput
  placeholder='email'
  textContentType='emailAddress'
  onChangeText={newText => setEmail(newText)}
/>
<Text>Password</Text>
<TextInput
  placeholder = 'password'
  textContentType='password'
  onChangeText={newText => setPassword(newText)}
/>
```

As you can probably guess, we now have a function to check the current auth-ed user, create a new account and sign-in.



## Handling user sign-ins (cont.)

Then call the `createUserWithEmailAndPassword()` function with a `<Pressable />`.

```
<Pressable
onPress = {()=>{
  createUserWithEmailAndPassword(auth, email, password)
    .then((userCredential) => {
      const user = userCredential.user
      console.log("User created with email: "+user.email)
    })
    .catch((error) => {
      const errorCode = error.code;
      const errorMessage = error.message;
    });
}}
>
  <Text style = {{fontSize: 16, color: "black"}}> Sign Up </Text>
</Pressable>
```

## Making nested pages

---

# Making a Stack of Pages

In order to push the user into an info page after they've signed up, we can create a Stack of Pages.

First, import the following function as such, and **yarn add** the two modules needed. (Small challenge!)

```
import {createNativeStackNavigator} from '@react-navigation/native-stack';
```

## Making a Stack of Pages (cont.)

Initialize the Stack with the following code by calling `createNativeStackNavigator`:

```
const Stack = createNativeStackNavigator();
```

## Making a Stack of Pages (cont.)

Rename the existing `const Account` into `const AccountScreen`.

Above `const AccountScreen`, create the following page:

```
const LoggedInScreen = () => {  
  const auth = getAuth()  
  
  return(  
    <Text>{auth.currentUser.email}</Text>  
  )  
}
```

## Making a Stack of Pages (cont.)

Below `const AccountScreen` create a `const Account` with the following code:

```
const Account = () => {  
  return (  
    <Stack.Navigator>  
      <Stack.Screen  
        name = "AccountScreen"  
        component={AccountScreen}  
        options={{ headerShown: false }}  
      />  
      <Stack.Screen  
        name = "LoggedInScreen"  
        component={LoggedInScreen}  
        options={{ headerShown: false }}  
      />  
    </Stack.Navigator>  
  )  
}
```

## Making a Stack of Pages (cont.)

Below the `console.log()` put the following code:

```
navigation.navigate("LoggedInScreen")
```

## End of Lesson

Questions? Reach out at: [karlorjalo@gmail.com](mailto:karlorjalo@gmail.com)