Making Your App Aware of Users

This lesson covers a powerful feature of the Firebase - the authentication state. With a tiny bit of code, your application will be aware of signed in users.

We'll cover the following

With any Firebase application, we need to add our configuration details to the top of our JavaScript file. For this app, we will also make a variable for `auth` so that we can access them easily throughout the app.

```
// Your web app's Firebase configuration
var firebaseConfig = {
  apiKey: "provided apiKey",
  authDomain: "provided authDomain",
  databaseURL: "provided databaseURL",
  projectId: "provided projectId",
  storageBucket: "provided storageBucket",
  messagingSenderId: "provided messagingSenderId",
  appId: "provided appId"
}

// Initialize Firebase
firebase.initializeApp(firebaseConfig)

// Reference to auth method of Firebase
const auth = firebase.auth()
```

JavaScript

Invoke `onAuthStateChanged` Firebase Method

Making your app aware of the user's authentication state is as easy as invoking the `onAuthStateChanged` Firebase method. It makes displaying different things to your users based on their state incredibly easy.

Firebase monitors the auth state in real-time. We can use an if/else statement to do different things based on that state.

```
// declare uid globally so you can access it throughout your app
let uid

auth.onAuthStateChanged(user => {
    if (user) {
        // Everything inside here happens if user is signed in
        console.log(user)
        // this assigns a value to the variable 'uid'
        uid = user.uid
        modal.style.display = `none`
    } else {
        // Everything inside here happens if user is not signed in
        console.log('not signed in')
    }
})
```

Check Your Console

After we run the code specified above, we will see the following message in the console: *not signed in*.

Making Authentication State Hide and Show Elements

In the next lesson, we are going to create a user. The act of creating a user also signs you in. When this happens, we want our program to show their name to them in the header. By doing this our users will have immediate feedback that the authentication worked.

`hide-when-signed-in`

We will add the class of `hide-when-signed-in` if we want to hide elements based on someone being **signed in**.

`hide-when-signed-out`

We will add the class of `hide-when-signed-out` if we want to hide elements based on someone being **signed out**.

Using the code below you will gain access to any element that the classes are present on.

```
// Access elements that need to be hidden or show based on auth state
const hideWhenSignedIn = document.querySelectorAll('.hide-when-signed-in')
const hideWhenSignedOut = document.querySelectorAll('.hide-when-signed-out')
```

JavaScript

Choosing What to Hide or Show

Now we will selectively add the classes to DOM elements that we want to show or hide.

```
        <!-- Header -->
        <div id="header">
                <div>
                        <img src="/udata/kvy8DD1x1RW/authentication-boilerplate-logo.jpg" alt
        </div>
        <div></div>
                <div class="hide-when-signed-in header-buttons-grid">
            <div>
                <button id="sign-in-link-header" class="auth gray-button" auth="show-sign-in-
            </div>
            <div>
                <button id="create-user-link-header" class="auth purple-button" auth="show-cr
            </div>
        </div>

                <div class="hide-when-signed-out hide" id="user-details-header">
                        <h1 id="display-name-header"></h1>
                </div>
        </div>


    <!-- Hero Banner -->
    <div class="hide-when-signed-in">
        <img src="/udata/GDpzwxlVjl2/hero-banner-desktop.jpg" alt="hero banner" id="hero-banr
        <img src="/udata/nWG9YbPorB4/hero-banner-tablet.jpg" alt="hero banner" id="hero-banne
        <img src="/udata/4Pb8eYMaPvn/hero-banner-phone.jpg" alt="hero banner" id="hero-banner
        </div>
```

```
</div>
```

HTML

Hero Banner Exchange

We want the *Hero Banner* to be hidden when our user signs in. We will be switching the *Hero Banner* in for the *Dashboard* while keeping the *Header** and *Footer** visible. Showing the dashboard to users will be covered soon, but for now, let's go ahead and hide the hero banner.

Modify the `onAuthStateChanged` firebase method

The `onAuthStateChanged` method is where we will control these classes. Classes are iterable, like arrays when selected with JavaScript. This is why we can use a `forEach` loop to get access to every element our classes are attached to. To make the magic happen, we add or remove the `class` of *hide* from elements.

While we are at it, let's also inject the user's display name into the header using the `id` of *display-name-header*.

```
// Makes your app aware of users
auth.onAuthStateChanged(user => {
  if (user) {

    // Everything inside here happens if user is signed in
    console.log(user)
    uid = user.uid
    modal.style.display = 'none'

    // Hides or shows elements depending on if user is signed in
    hideWhenSignedIn.forEach(eachItem => {
      eachItem.classList.add('hide')
    });
    hideWhenSignedOut.forEach(eachItem => {
      eachItem.classList.remove('hide')
    });

    // Greet the user with a message and make it personal by using their name
    if (user.displayName) {
      document.getElementById('display-name-header').textContent = `Hello, ${user.displayName
    }

  } else {
    // Everything inside here happens if user is not signed in
    console.log('not signed in');

    // Hides or shows elements depending on if user is signed out
    hideWhenSignedIn.forEach(eachItem => {
      eachItem.classList.remove('hide')
    });
```

```
    hideWhenSignedOut.forEach(eachItem => {
      eachItem.classList.add('hide')
    });


  };
});
```

JavaScript

A few more lines of CSS

Lastly, let's add a couple more lines of CSS to keep it all looking good.

```
#user-details-header{
        text-align: right
}

#have-or-need-account-dialog{
    padding-top: 10px;
}
```

CSS

The Authentication Boilerplate Application

Check your console; you should see the **not signed in** message because your app is now aware of the user's authentication state.

This code requires the following keys to execute:

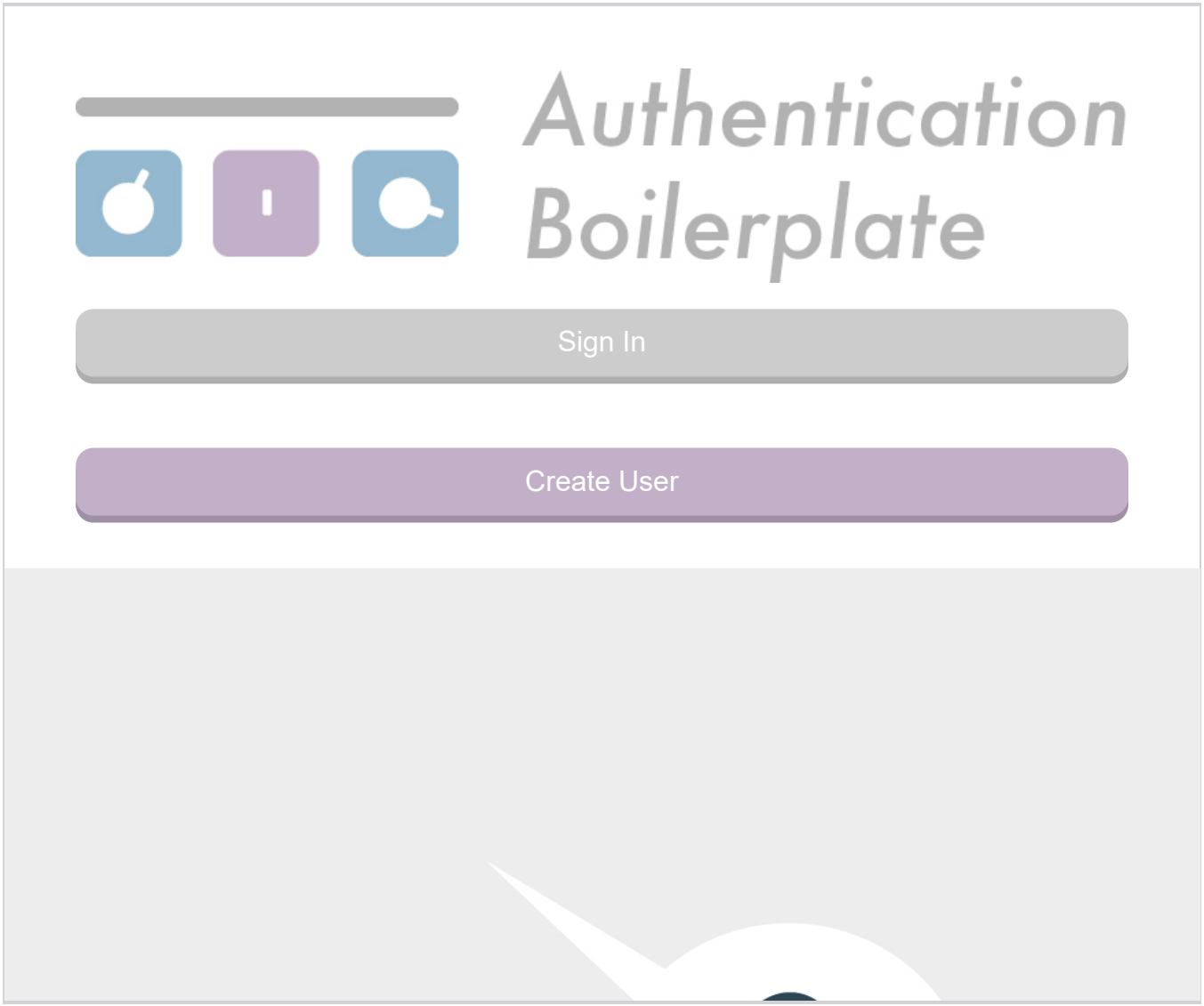
Key:	Value:
apiKey	Not Specified...
authDomain	Not Specified...
databaseURL	Not Specified...
projectId	Not Specified...
storageBucket	Not Specified...
messagingSenderId	Not Specified...
appId	Not Specified...

Output

JavaScript

HTML

CSS (SCSS)



In the next lesson, we create a user with the email and password method. Once the user is created, you will be able to see the `onAuthStateChanged` method you implemented in this lesson in action.