

# Authentication Forms

The most common type of authentication is email and password. This lesson covers setting up the forms as well as the functionality required to toggle between them as needed.

## We'll cover the following



- HTML for the Forms
- Style the Forms
- Access the forms
- The hide Class
- `hideAuthElements()`
- Modify the `showCreateUserForm` and `showSignInForm` functions
- We Now See the Correct Forms
- Showing the Forgot Password Form
  - `showForgotPasswordForm` Function
  - Adding `showForgotPasswordForm()` to the Authentication Actions Listener
- Toggle Between Forms With the Modal Open
  - HTML For the Have or Need Account Toggle
  - Make it Look Good
  - Hide or Show the Correct Dialog
- The Authentication Boilerplate Application

## HTML for the Forms #

We are going to have three separate forms, each to complete a different task.

The name of the forms explain what they do:

1. **Create user** form
2. **Sign in** form
3. **Forgot password** form

The HTML is straightforward and short so I am going to present all of it to you right now. All forms go inside your modal.

```
<!-- Modal -->
<div id="modal">
  <div id="modal-content">
    <button id="close">&times</button>
    <!-- Authentication -->
    <div id="authentication">
      <!-- Create user form -->
      <form id="create-user-form">
        <div id="create-user-inputs">
          <input type="text" id="create-user-display-name" placeholder="Display Name" autocomplete="off">
          <input type="email" id="create-user-email" placeholder="Email" autocomplete="off">
          <input type="password" id="create-user-password" placeholder="Password" autocomplete="off">
        </div>
        <button id="create-user-button" class="purple-button" type="submit">Create User</button>
      </form>

      <!-- Sign in form -->
      <form id="sign-in-form">
        <div id="sign-in-inputs">
          <input type="email" id="sign-in-email" placeholder="Email" autocomplete="off">
          <input type="password" id="sign-in-password" placeholder="Password" autocomplete="off">
        </div>

        <div id="forgot-password-link" class="auth" auth="show-forgot-password">
          <button id="sign-in-button" class="purple-button" type="submit">Sign In</button>
        </div>
      </form>

      <!-- Forgot password form -->
      <form id="forgot-password-form">
        <div id="forgot-password-inputs">
          <input type="email" id="forgot-password-email" placeholder="Email" autocomplete="off">
        </div>
        <button id="forgot-password-button" type="submit">Send Reset Link</button>
      </form>
    </div>
  </div>
</div>
```

HTML

## Style the Forms #

We add this CSS to make your forms look good.

```
input{
  width: 100%;
  font-size: 20px;
  padding: 10px 0px 10px 0px;
```

```
padding: 10px 0px 10px 0px;
border-width: 0px 0px 2px 0px;
}

input::placeholder{
  color: #989898
}

input:focus{
  border-color: green;
  outline: none;
}

#sign-in-link, #create-user-link{
  color: #32a1d7
}

#create-user-inputs, #sign-in-inputs{
  display: grid;
  grid-template-columns: 1fr 1fr;
  grid-gap: 20px;
  margin-bottom: 30px;
}

#create-user-display-name{
  grid-column-end: span 2;
}

#forgot-password-inputs{
  margin-bottom: 20px;
}

#forgot-password-link{
  text-align: right;
  font-size: 10px;
  margin-top: -17px;
}
```

CSS

## Access the forms #

From your JavaScript, get access to the forms by using the `querySelector`.

We need to do this for two reasons:

1. Creating functionality to toggle between the forms. This is what we are handling in this lesson.
2. Ability to submit the form information to Firebase to authenticate our users, or reset a password. We will do this in an upcoming lesson.

```
// Access the forms for email and password authentication
const createUserForm = document.getElementById('create-user-form')
const signInForm = document.getElementById('sign-in-form')
```



```
const forgotPasswordForm = document.getElementById('forgot-password-form')
```

JavaScript

## The `hide` Class #

We add this class to your CSS. This is important for the following code examples but we will also use it throughout our **Authentication Boiler Plate** application.

```
/* Important class used throughout the course */  
.hide{  
  display: none !important;  
}
```

CSS

## `hideAuthElements()` #

When we click the **sign in** link or **create user** button, we want to make sure we show the correct form in the modal. The easiest way to handle this is by hiding all forms before selectively showing the correct form. This is where the `hideAuthElements()` function comes in. Without it, we would be repeating a lot of code, but with it, our app is easier to read and more elegant.

Below you see the new `hideAuthElements()` with functionality built in to hide the create user, sign in, and forgot password form. This is done with a CSS class we just made called `hide`.

```
// Invoked at the start of auth functions in order to hide everything before selectively showing the  
hideAuthElements = () => {  
  createUserForm.classList.add('hide')  
  signInForm.classList.add('hide')  
  forgotPasswordForm.classList.add('hide')  
}
```

JavaScript

## Modify the `showCreateUserForm` and `showSignInForm` functions #

Now we can invoke the `hideAuthElements` function inside of the `showCreateUserForm` and `showSignInForm` functions. After the code below is implemented, the **sign in** button in the header will show the modal and show the **sign in** form. The same thing will happen with the **create user** button; it will show the modal and the **create user** form.

Notice on lines 5 and 12, we remove the CSS property `hide` from the correct elements so that it is the form that gets shown when invoking that function.

```
// Invoked when user wants to create a new user account
showCreateUserForm = () => {
  hideAuthElements()
  modal.style.display = 'block'
  createUserForm.classList.remove('hide')
}

// Invoked when a user wants to sign in
showSignInForm = () => {
  hideAuthElements()
  modal.style.display = 'block'
  signInForm.classList.remove('hide')
}
```

JavaScript

## We Now See the Correct Forms #

When clicking the **sign in** or **create user** buttons our modal is showing the correct form.

## Showing the Forgot Password Form #

What about the **forgot password** form? There is a little “**Forgot?**” right below the **showSign in** form that we are going to use to show the **forgot password** form.

### `showForgotPasswordForm` Function #

The first step is to create a function that handles the logic. It’s basically identical to the `showSignInForm` and `showCreateUserInForm` functions we created earlier.

```
// Invoked when a user wants reset their password
showForgotPasswordForm = () => {
  hideAuthElements()
  modal.style.display = 'block'
  forgotPasswordForm.classList.remove('hide')
}
```

JavaScript

Adding `showForgotPasswordForm()` to the Authentication Actions Listener #

```
// Loop through elements and use the associated auth attribute to determine what action to take when clicked
authAction.forEach(eachItem => {
  eachItem.addEventListener('click', event => {

    let chosen = event.target.getAttribute('auth')
    if (chosen === 'show-create-user-form'){
      showCreateUserForm()
    }
    else if (chosen === 'show-sign-in-form'){
      showSignInForm()
    }
    else if (chosen === 'show-forgot-password-form'){
      showForgotPasswordForm()
    }
  })
})
```

JavaScript

## Toggle Between Forms With the Modal Open #

Have you ever clicked a sign in link when you really needed to click on create a user? Let's solve this problem. Right now our modal has to be closed then another button or button has to be clicked to get to the other form. Let's fix that by making a toggle inside the modal itself. This will allow the user to switch between the **sign in** and **create user** forms without the irritation of closing and reopening the modal.

## HTML For the *Have or Need Account Toggle* #

```
<!-- Forgot password form form -->
<form id="forgot-password-form">
  <div id="forgot-password-inputs">
    <input type="email" id="forgot-password-email" />
  </div>
  <button id="forgot-password-button" type="submit">Forgot Password</button>
</form>

<!-- Have or need account dialog -->
<div id="have-or-need-account-dialog">
  <p id="create-user-dialog">
    Don't have an account?
    <span id="create-user-link" class="auth" auth="show-create-user-form">Create User</span>
  </p>
  <p id="sign-in-dialog">
    Already have an account?
    <span id="sign-in-link" class="auth" auth="show-sign-in-form">Sign In</span>
  </p>
</div>
```

## Make it Look Good #

I added a little padding to the top of this dialog area.

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



## Hide or Show the Correct Dialog #

From the **sign in** form, we want users to see:

*Already have an account? **Sign in***

From the **create user** form, we want users to see:

*Don't have an account? **Create User***

We can use our already created `showSignInForm` and `showCreateUserForm` functions to hide or show these dialogs at the appropriate times.

Modify them like this:

```
// Invoked when user wants to create a new user account
showCreateUserForm = () => {
  hideAuthElements()
  modal.style.display = 'block'
  createUserForm.classList.remove('hide')
  signInDialog.classList.remove('hide')
  haveOrNeedAccountDialog.classList.remove('hide')
}

// Invoked when a user wants to sign in
showSignInForm = () => {
  hideAuthElements()
  modal.style.display = 'block'
  signInForm.classList.remove('hide')
  createUserDialog.classList.remove('hide')
  haveOrNeedAccountDialog.classList.remove('hide')
}
```



## The Authentication Boilerplate Application #

There was a lot of set up to get this far but it finally starts paying off now! Click **sign in** to see the *sign in form*. Click **create user** to see the *create user form*.

Output

JavaScript

HTML

CSS (SCSS)



# Authentication Boilerplate

Sign In

Create User



Console

Clear

In the next lesson, you will learn how to make your app aware of users.