# Full Stack Cat App VI

## Making the Login and the Profile Components Modals

* In this app, rather than actually using a pop-up, the modal effect is going to be created by CSS.
* The first step is to get the site portal view to be only the size of the screen.
* In portal.css set the position to absolute, give a padding at the top of 5em, set top to 0px and the z-index to -10 .
* The scrollbar on the right should disappear as the background goes 'underneath' the navbar.
* In views/layouts/login login.ejs and register.ejs add div containers and wrappers. Give everything class names:

````
<div class="login-container">
<div class="login-form-wrapper">
  <h1>Login</h1>
  <form action="/auth/login" method="POST" class="user-form">
    <h3>Please enter your email and password</h3>
    <p><code>GET /auth/login</code></p>
    <input id="uemail" type="email" name="email" class="user-input" placeholder="useremail">
    <input id="password" type="password" class="user-input" name="password"
placeholder="userpassword">
    <input type="submit" class="user-input" value="Submit">
    <br/>
    <br/>
    <input type="button" value="Register" class="user-input"
onClick="window.location.href='/new/register'">
  </form>

    <div class="message-div">
      <%- include('./partials/passport-message') %>
      <% if(success && success.length > 0){ %>
          <div>
          <%= success %>
          </div>
          <% } %>

      <% if(error && error.length > 1){ %>
          <div>
          <%= error %>
          </div>
      <% } %>
    </div>
</div>

</div>
````

* IMPORTANT - because the login and the registration components will be styled the same (to give the appearance of a modal) name the div classes "login-container" and "login-form-wrapper" on BOTH login.ejs and register.ejs. Just to be safe, name the "message-wrapper" in both files message-div.
* Pick a dynamic background, this will further the appearance of a modal.
* In portal.css, class .portal-container, set the background-image to an url(), set background-repeat to no-repeat, and background-size to cover.

````
.portal-container {
    height: 100vh;
    width: 100vw;
    position: absolute;
    padding-top: 5em;
    top: 0px;
    z-index: -10;
    background-image: url('../../images/infiniteCat.gif');
    background-repeat: no-repeat;
    background-size: cover;
}
````

* public/assets/css make a folder called login-styles . In that folder make a file called login.css . @import login.css to style.css .
* In public/css/login-styles/login.css make an entry for the class .login-container . Set the height to 100vh, the width to 100vw and set the background the same as .portal-container :

````
.login-container {
    height: 100vh;
    width: 100vw;
    background-image: url('../../images/infiniteCat.gif');
    background-repeat: no-repeat;
    background-size: cover;
}
````

* The gif of the horrible cats jumping in a loop is in public/assets/images
* When you arrive at the site (localhost:3000) you will see the horrible cats jumping, and when you press the login button and go to login or register it will look like the login popped up (sort of -- we'll work more on that).
* If everything seems to be working, save to GitHub.

## Starting the Form Re-styling (Finally!)

* Cut all the classes from .bold (.user-form, the various .user-input) to .code in style.css and paste them into login.css .
* In public/assets/css/login-styles make a file called profile.css and @import it into style.css.
* style.css should now look like this:

````
@import './site-styles/bar.css';
@import './site-styles/portal.css';
@import './site-styles/site.css';
@import './login-styles/login.css';
@import './login-styles/profile.css';

*, body, html, ::before, ::after {
    box-sizing: border-box;
    margin: 0;
    padding: 0;
}

img {
    height: 200px;
    width: auto;
}

.form-container {
    background-color: red;
    height: 100vh;
    width: 100vw;
}

.cat-container {
    background-color: cyan;
    height: 100vh;
    width: 100vw;
}

.cat-image-div {
    display: flex;
    cursor: pointer;
}

````

* In login.css make a class declaration for .login-form-wrapper, give it a height of 75% and a width of 55% and give it a rgba color of rgba(24, 3, 41, 0.8); (a dark semi-transparent blue-purple) :

````
.login-form-wrapper {
    height: 75%;
    width: 55%;
    background-color:rgba(24, 3, 41, 0.8);
}

````

* In login.css in ".login-container" and ".login-form-wrapper" set the display to flex, justify-content: center; and align-items: center;
* Give ".login-form-wrapper" a border-radius of .5em , and set flex-direction to column and color to white.

````
.login-container {
    height: 100vh;
    width: 100vw;
    background-image: url('../../images/infiniteCat.gif');
    background-repeat: no-repeat;
    background-size: cover;
    display: flex;
    justify-content: center;
    align-items: center;
}

.login-form-wrapper {
    height: 75%;
    width: 55%;
    background-color:rgba(24, 3, 41, 0.8);
````

```
    display: flex;
    justify-content: center;
    align-items: center;
    border-radius: .5em;
    flex-direction: column;
    color: white;
}

````

* In ".user-form" change the background color to the same color as the ".login-form-wrapper" but solid : rgb(24, 3, 41) .
* In ".user-form" change the max-width to 75% , add a height of 100% and set the max-height to 75%;
* In register.ejs and login.ejs remove the <p><code> section and the <br /> . The code style can be removed from login.css.
* In login.css in ".user-form" change the display to flex, the flex-direction to column and the justify-content to space-around.
* In bar.css set the ".bar" display to flex;
* In bar.css ".login-bar" change the background-color to rgba(24, 3, 41, 0.8) and align-items to center.
* Give ".login-bar-btn" a margin-left of 30px, a background-color of rgb(24, 3, 41) and a color of white.
* Remove the <h1> from portal.ejs so that it just shows the horrible cats and the login bar.
* On hover, change all the button/button-like inputs to coral.
* The styling all needs to be refined. This is just a rough start. Really refine the modal as much as possible.

## Continue Styling Entire Site

* Continue styling the entire site the same way. On the profile module and on rest-of-site.ejs use inline styling to set the background image rather than the css, in order to use the user's favorite cat image. <div class="site-container" style="background-image: url('<%= fci %>')";> The sizing, etc. can still be done in css.

## Final Modal Element, The Close X .

* In both login.ejs and register.ejs make an opening and closing div before <h1>Login</h1> and <h1>Register</h1> respectively. In the div put an <a> tag with an href of '/' . Give it the text of &#10005; which will print out as a closing X . Give the <a> tag a class of "login-modal-anchor" .

````
<div class="login-container">
<div class="login-form-wrapper">
````

````
<div class="close-modal-login">
<a class="login-modal-anchor" href="/">&#10005;</a>
</div>
<h1>Login</h1>

````

* Here is the styling to put the closing X in the right-hand corner :

````
.close-modal-login {
   margin-top: -35px;
   width: 100%;
   height: 100%;
   max-height: 2em;
   display: flex;
   flex-direction: row;
   justify-content: flex-end;
}

.login-modal-anchor {
   text-decoration: none;
   margin-right: 10px;
   font-size: 2em;
   color: white;
}


````

* In order to get some styling, routes might need to be added or changed. For example, to get the customized cat background on the edit email page, CatFancier needs to be added to the route to extract the favoriteCatImg data. To add the X to the profile modal a get route for restOfSite needs to be made so that the X href can point to /restOfSite/<%= id %> . Probably the delete profile landing page can be removed and the delete function redirected to '/'.

## What's left?

* Well, the site still needs functionality, but everything else is done! Yay!