Web Design DeCal

Homework 7 November 12th

UI Modals in jQuery

Required Tools: Text Editor & Google Chrome

You're going to be building a Modal Box. In particular, you'll be making one that allows people to sign up onto your awesome site!

Assignment

First, we're going to build the Modal Box from scratch, using HTML and CSS. Next, we're going to define its behavior as a Modal Box using the power of jQuery. Please read through the entire assignment before you begin.

Download Assignment

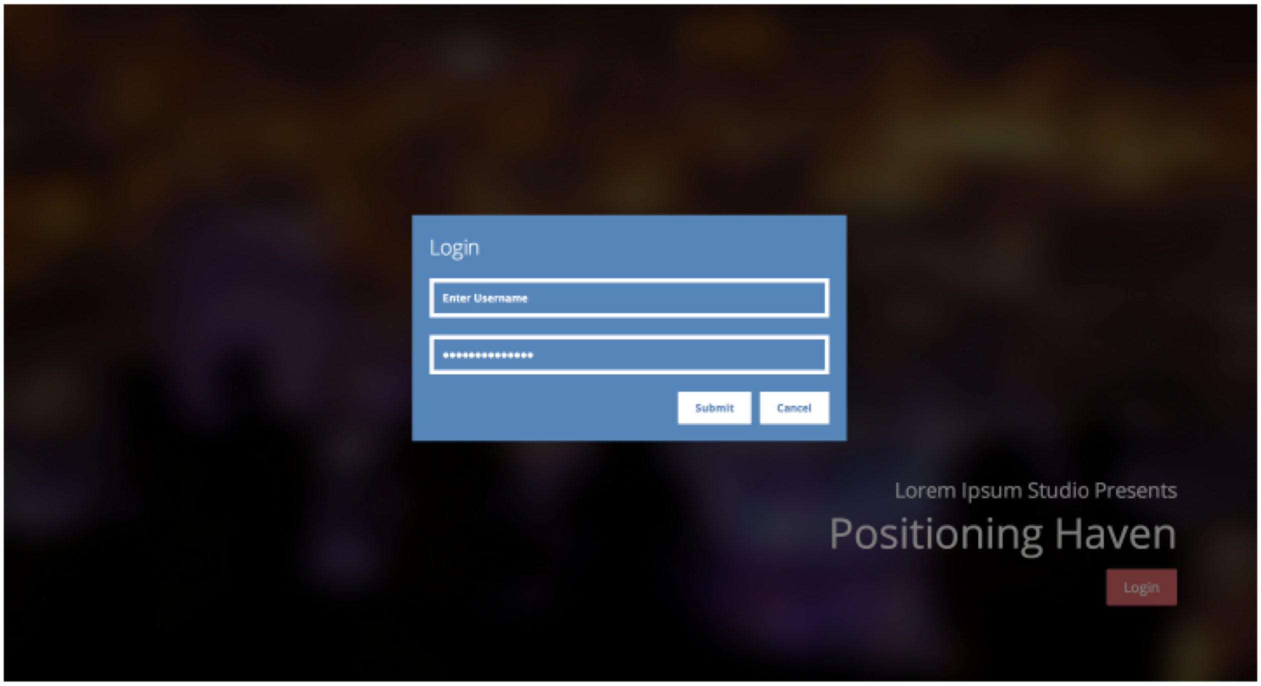
Part 1 - Creating the Modal

Modal Boxes are used to present information on top of a page. You can find them at the top of your Facebook profile, Pinterest page, Twitter feed, etc.

They usually consist of three main components:

1. Modal Container
2. Modal Overlay
3. Modal Box

This is what you will be creating for this homework:



The **Modal Container** encloses the Modal Overlay and Modal Box Wrapper.

The **Modal Overlay** is a semi-transparent black layer (div) that goes *on top* of the background but *below* the Modal Box. This is used to dim the background, so the user's attention is diverted towards the Modal Box.

The **Modal Box** is the star of the show. Along with it being both vertically and horizontally centered on the page, it is what takes in input from the user.

Question 1: Style the Container, Overlay and Modal Box

The **Modal Container** should:

1. Be full width and height
2. Have position: fixed
3. Have a z-index of 100

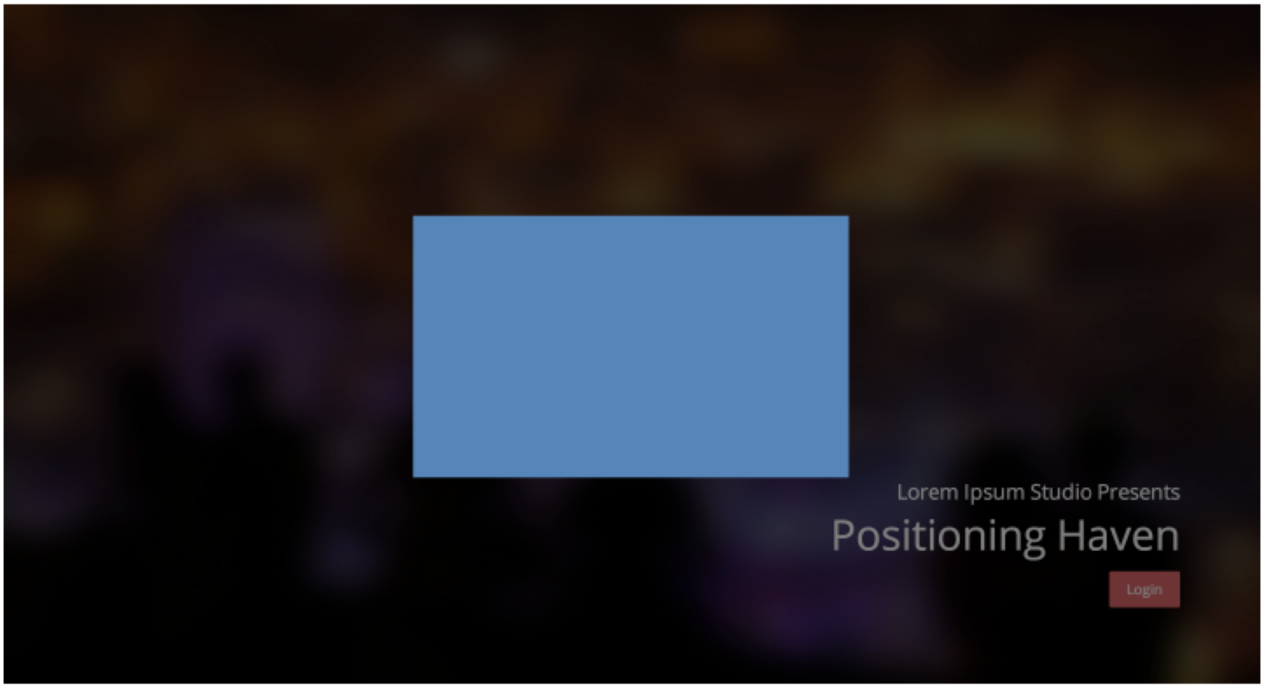
The **Modal Overlay** should:

1. Be full width and height
2. Be 50% transparent and black (background: rgba(0, 0, 0, 0.5);

The **Modal Box** should:

1. Be 500px wide
2. Have a height of 260px
3. Have a background color of #5485BB
4. Be positioned *on top* of Modal Overlay and background (absolute)
5. Be horizontally and vertically centered (try using calc!)
6. Have a z-index of 100

We should have something like this:



Part 2 - Adding the Content

Now that the Modal Container and its elements are set up, let's populate the Modal Box itself with some content!

Since our Modal Box will be for users to log in to our site, it should naturally have the following sections:

Modal Header:

- The header will contain the title, which should say "Login"

Modal Content:

- One input box that should prompt the user to "Enter Username"
- One input box that should prompt the user to "Enter Password"

Modal Footer:

- Contains one "Submit" button
- Contains one "Cancel" button

Question 2: Write HTML for the Login Components

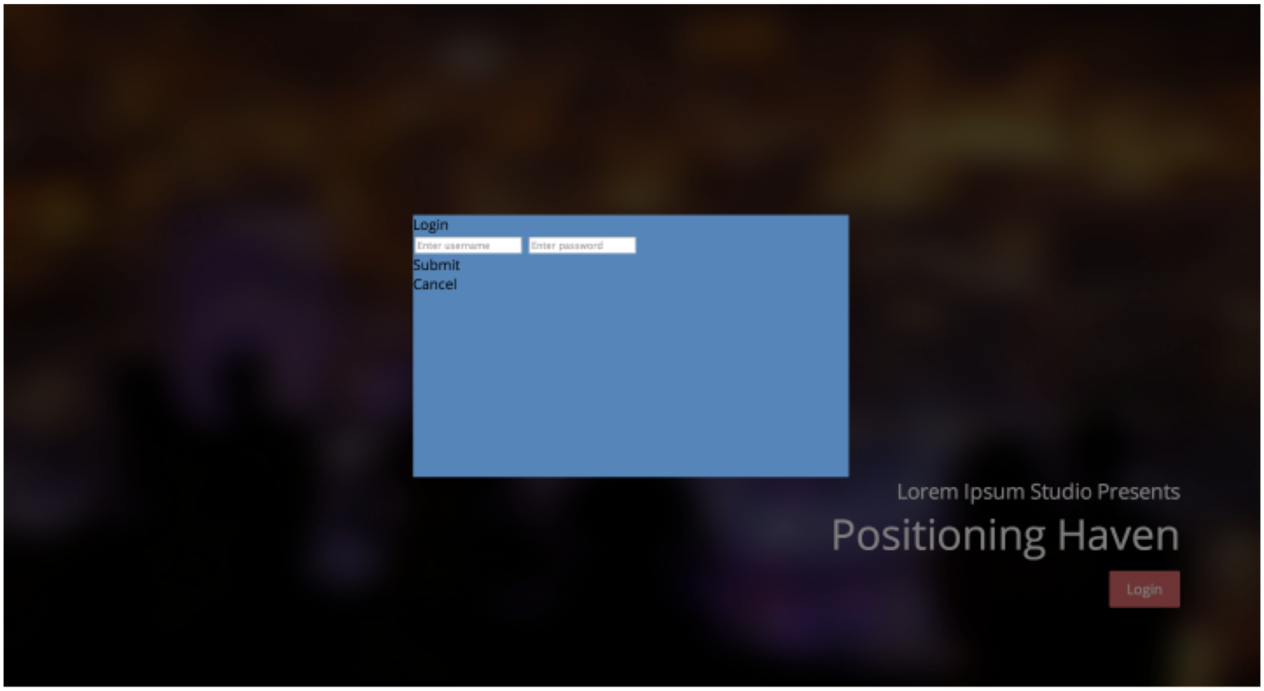
For the **Modal Header**, simply create a div with the text "Login".

For the **Modal Content**, create a div and place the two input boxes for the username and password inside it. The following code creates an input text field in HTML. Use it to create an input box for both the username and the password. Be sure to change the placeholder accordingly. Note that you'll want to add a class (choose a name) to each input box so you can style them later:

```
<input type = "text" class = "NAME_HERE" placeholder = "Enter Username" />
```

To create the **Modal Footer** and its buttons, just use divs inside divs!

You should have something like this so far:



Note: Although you are free to style the Modal Box components as you please, please make sure your Modal Box has a total height of 260px and a total width of 500px. It should also be horizontally and vertically centered as depicted above.

Question 3: Style Those Modal Components

Style the Modal Header using your knowledge of CSS so that it looks like it does in the first picture.

Next, Let's focus on the "Submit" and "Cancel" buttons. First of all, style them using basic CSS so they too look like they do in the first picture above. Next, add **cursor: pointer** to both buttons to change the cursor to a pointer when you hover over them. We also want the buttons to stack *horizontally*. There are the three main ways to horizontally align elements:

1. Using **floats**, along with clearfix
2. Using **position: relative** on the container and **position: absolute** on the children
3. Using **display: inline-block** on each element

Input boxes can be tricky to style using CSS, mainly because they come with default CSS styles. We don't want our input boxes to look as ugly as they do right now, so let's override those default styles.

Add the following code to your stylesheet:

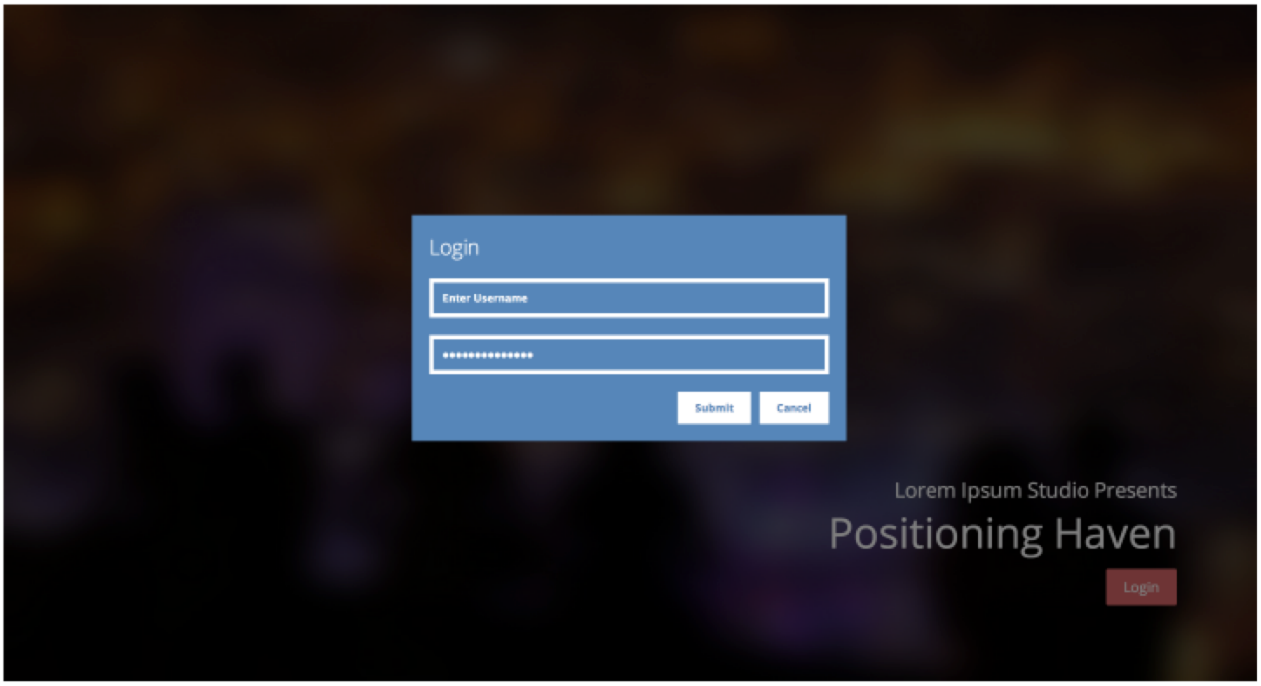
```
.INPUT_BOX_CLASS_NAME_HERE {
    border: 2px solid rgba(255, 255, 255, 0.6);
    padding: 10px;
    background: transparent;
    margin: 0px;
    margin-bottom: 20px;
    width: 460px;
    font-size: 12px;
    font-weight: 700;
    color: white;
}
```

One thing that's unique about input boxes is that they have something called *input placeholder text*, which is essentially a message that tells the user what input is expected. Notice how the placeholder text isn't white although we specified **color: white** above. Input boxes are annoying because you need to seperately override the default placeholder text styles. To do that, you can use the -webkit-input-placeholder CSS selector.

Add the following code to your stylesheet:

```
.INPUT_BOX_CLASS_NAME_HERE::-webkit-input-placeholder {
    color: white;
}
```

You, should have something that looks like this:



Part 3: Add Behavior Using jQuery

We want our Modal Box to have some functionality. In particular, here's how we want the Modal Box to work:

1. Clicking the red "Login" button on the home page brings up the Modal Box
2. Clicking the "Submit" or "Cancel" button hides the Modal Box
3. Clicking *anywhere outside* the Modal Box also closes the Modal Box (Good UX!)
4. Clicking the blue Modal Box itself, excluding the buttons, *does not* close the Modal Box

All of this means that the Modal Box doesn't appear initially. Add **display: none** to the Modal Container to hide both the Modal Box *and* the Modal Overlay when the page is loaded.

Question 4: Implement #1 and #2 Above

To get the Modal Box to show up when the user clicks on "Login", the entire Modal Container should show up. This is because not only do we want the Modal Box to show up, but also the Modal Overlay in order to dim the background! Likewise, to hide the Modal Box, the entire Modal Container should become hidden.

To do this, create a **click()** event listener for each of the three buttons, and use the **show()** and **hide()** functions to change the display state of the Modal Container.

Note: There are other implementations that could also work here, so feel free to use which ever one you prefer!

Question 5: Implement #3 and #4 Above

#3 uses some interesting jQuery. How can we use the **click()** event listener on the Modal Overlay?

Here is the basic idea:
Execute the **hide()** function when a user clicks outside the Modal Box. Since we set up our Modal Container in such a way that the Modal Box is *on top* of the Modal Overlay (thanks, z-index!), clicking the Modal Box itself won't count as a click on the Modal Overlay. Therefore, clicking the Modal Overlay is the same as "clicking outside of the Modal Box."

Just by creating a **click()** event listener for the Modal Overlay, you've effectively implemented #3 and #4!

Hurrah! Congratulations! You've made a working Modal Box. We definitely encourage you to experiment with different designs if you have time!

As always, feel free to ask questions on Piazza if you have trouble and need some help.

Submission

Submit the following files as a **zip** file through Assignments under Resources tab of the WDD Portal:

- index.html
- assets
    - style.css
    - script.js
    - jquery-1.9.1.js
    - background.jpg

Your assignment must be submitted as a **zip** file. Submission will automatically fail if your submission does not contain the index.html or if is not a zip file.

This assignment is due by Thursday, **November 12th** at 5PM.