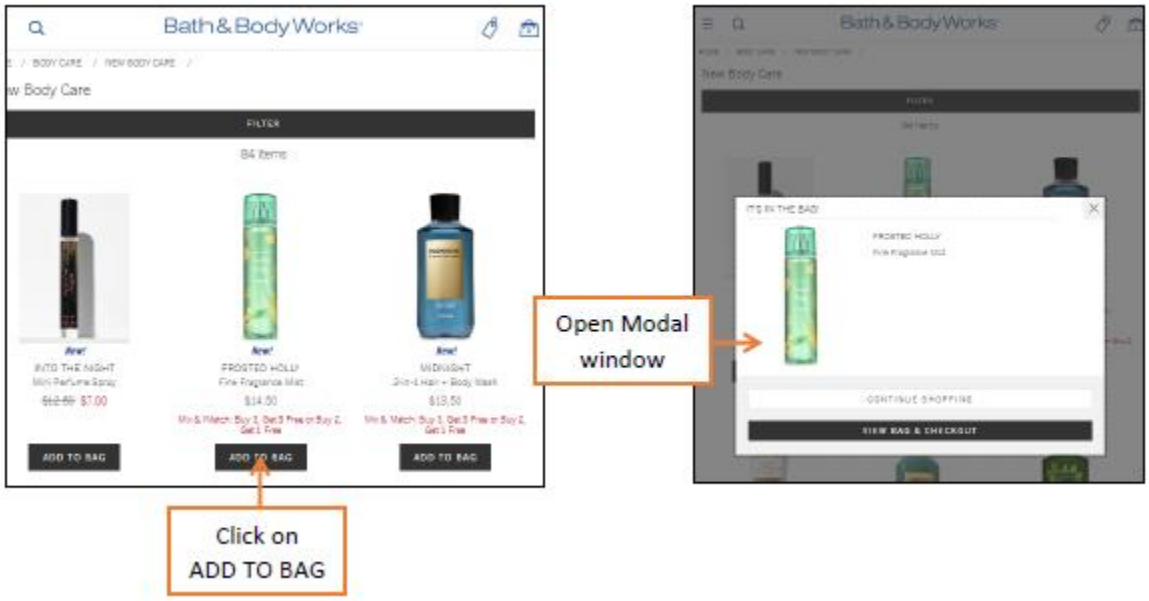
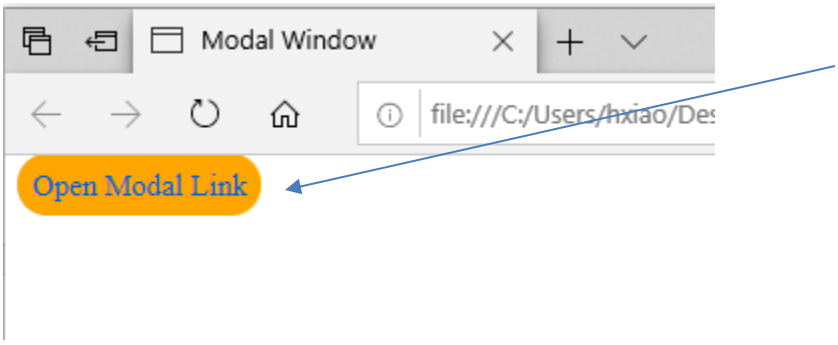
Creating a modal window using HTML and CSS

A modal is a dialog box/popup window that is displayed on top of the current page



There are different techniques to create modal windows, one way to create a modal window is by using CSS's attributes as transition, opacity, pointer-event, and background gradient properties.

To create a model window, we need to create open modal link:



HTML

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Modal Window</title>
    <link rel="stylesheet" href="modal.css" type="text/css">
  </head>
  <body>
    <a href="#openModal" class="linkModel">Open Modal Link</a>
  </body>
</html>
```

modal.css

```
.linkModel{
      text-align:center;
      background-color: orange;
      font-size:  20px;
      padding: 10px;
      border-radius: 20px;
      text-decoration:none;
}
```

Run the program to see if the modal link looks like this: 

From the HTML code, the code line:

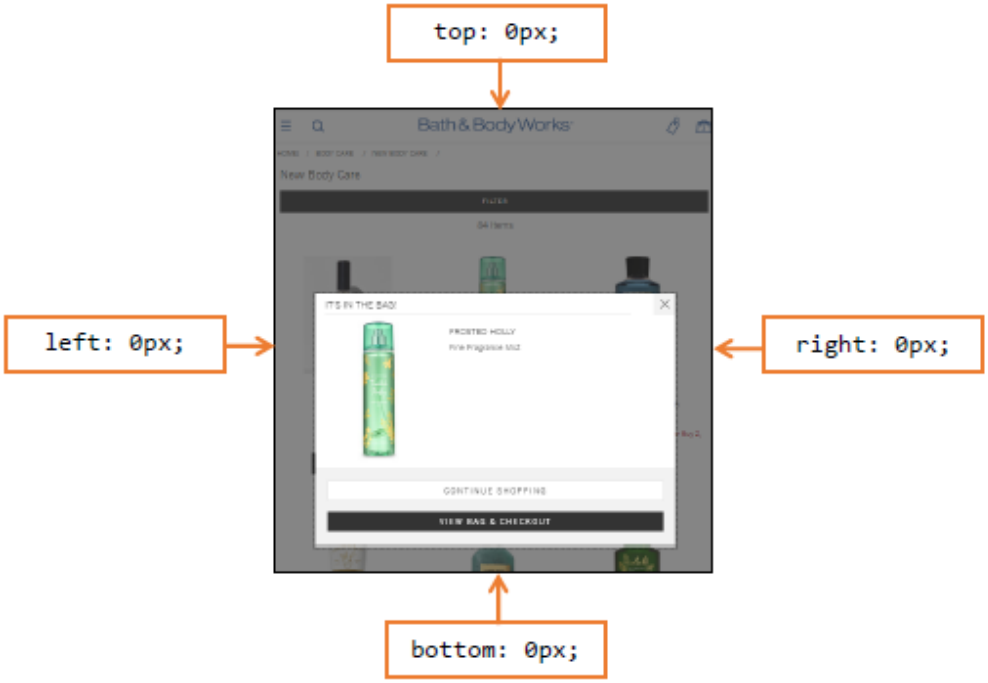
```
<a href="#openModal" class="linkModel">Open Modal Link</a>
```

we can see that the modal is linked to an element **id** as `"#openModal"`. Therefore, we have to create the modal element using `<div>` element and we will **id** it as: `id="openModal"`. This division we will also has a class name as: **`class="modalWindow"`** for css attributes.

```
    <a href="#openModal" class="linkModel">Open Modal Link</a>
    <!-- Modal Window container -->
    <div id="openModal" class="modalWindow">

    </div>
</body>
```

Since the modal window will cover the entire screen, we can set the top, right, left, and bottom to 0px. This setting meaning that there will be 0px of distance between the top, right, left, and bottom to the modal window.



Once the modal window is created, we have to apply transparency to the modal window. This property can be set by setting the window background to 80% of transparency using the **rgba** color property:

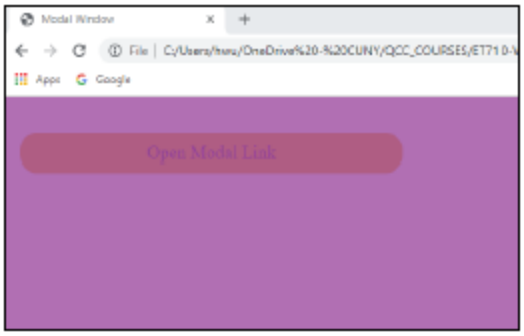
```
Background-color: rgba(160,80,160,0.8);
```

The CSS code should look as the following:

modal.css

```
.linkModel{
width: 60%;
text-align:center; background-color: orange; font-size:  20px; padding: 10px;
border-radius: 20px;
}
.linkModel a{
text-decoration:none;
}
.modalWindow {
position: fixed; top: 0;
right: 0;
bottom: 0;
left: 0;
background-color: rgba(160,80,160,0.8);
}
```

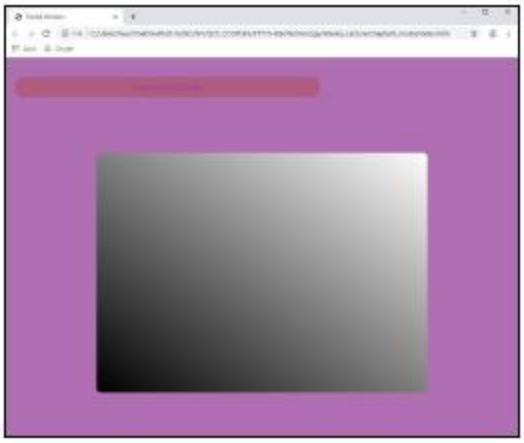
The internet browser should look as:



Having the background of the modal window set, we can add a message window on top of the modal window using a `<div>` with class name **`msgText.`** This message window is used to display the advertisement of the modal. Since the message window will display in-front of the modal window, therefore this division will have **`position: relative`**. To make the message window more colorful and organized, we can add linear-gradient background to it, set the width to 80%, the height to border- radius, and 40% of top margin:

```
.msgText{
position: relative;
width: 700px;
height: 500px;
border-radius: 10px;
margin: auto;
margin-top: 200px;
background: linear-gradient(to top right, black, white);
}
```

The internet browser should look as:



`msgText` is used as a message window. Therefore, we can add a title to this message window using **`<h3>`** and a text container using **`<p>.`**

To make my message window more styling, in CSS, we add padding to `<h3>` and `<p>`, and align the text to justify in `<p>`. Remember that to call the `<h3>` and `<p>` without using a class name, we will need to call the division that contains them first and then the element, for example:
`.msgText h3`

```
.msgText h3{
padding: 20px; background-color: black; color: white; }

.msgText p{
padding: 20px; text-align: justify; background-color: white;}
```

We can also add an image to the message window:

To style the image in the message window, we can make the width of the image to 50% of the message window width, and center the image by setting the margin-left or margin-right to 25%:

```
.msgText img{
  width: 50%;
  height: auto;
  margin-left: 25%;
}
```

The next thing to add to the message window `msgText` is a link to close the modal window. For this, we can use an anchor `<a>` tag and link it to the top of the web app using: `href="#"`. We can name it as `class="modal-close"` for CSS attributes.

Now is CSS, we can set attributes to `.modal-close` such as width, the font color, padding, font-weight, text-align, and set the text-decoration to none:

```
.modal-close {
      color: #00d9ff;
      padding: 12px;
      font-weight: bold;
      text-align: center;
      text-decoration: none; }
```

Once we have the modal window set, we can set the **opacity** attribute to it. The idea of using the **opacity** attribute is to make the modal window to invisible, **opacity:0;** , when the web app is loaded, then when the modal link is clicked, the modal window will become visible, **opacity: 1;** . To set this attribute, we add **opacity:0;** in the CSS of the modal window **.modalWindow**

```
.modalWindow {
position: fixed; top: 0;
right: 0;
bottom: 0;
left: 0;
background-color: rgba(160,80,160,0.8);
opacity: 0;
}
```

The other setting is to create a hover effect to the modal window to open to modal window using **opacity: 1;**

```
.modalWindow:hover{
  opacity: 1;
}
```

If we run the app, the message window will not close when we click on the CLOSE link. This happens because we did not set the **pointer-events** to **target** an element that has the CLOSE link. For this, we can include **pointer-events:none**; to the element that has the anchor tag **<a>** beneath it. In this activity, we target the modal window **.modalWindow** and when click on the CLOSE link, the model window will close, that is why we need to change **.modalWindow:hover** to **.modalWindow:target** and add a **pointer-events: auto;** which means that the any anchor **<a>** tag under **.modalWindow** will react to the pointer events. In other words, when we click on the CLOSE link, the anchor **<a>** tag will take us to the beginning to the web app as states with **href="#"**

```
.modalWindow {
position: fixed; top: 0;
right: 0;
bottom: 0;
left: 0;
background-color: rgba(160,80,160,0.8);
opacity: 0;
pointer-events: none;
}
.modalWindow:target{
  opacity: 1;
pointer-events: auto;
}
```

Add this line

Add this line

Change hover to target

We can also add some animation to our modal window, for example, we can make the modal window to **ease-in** when it opens.

```
.modalWindow:target{
  opacity: 1;
  pointer-events: auto;
transition: ease-in 0.5s;
}
```

Add this line