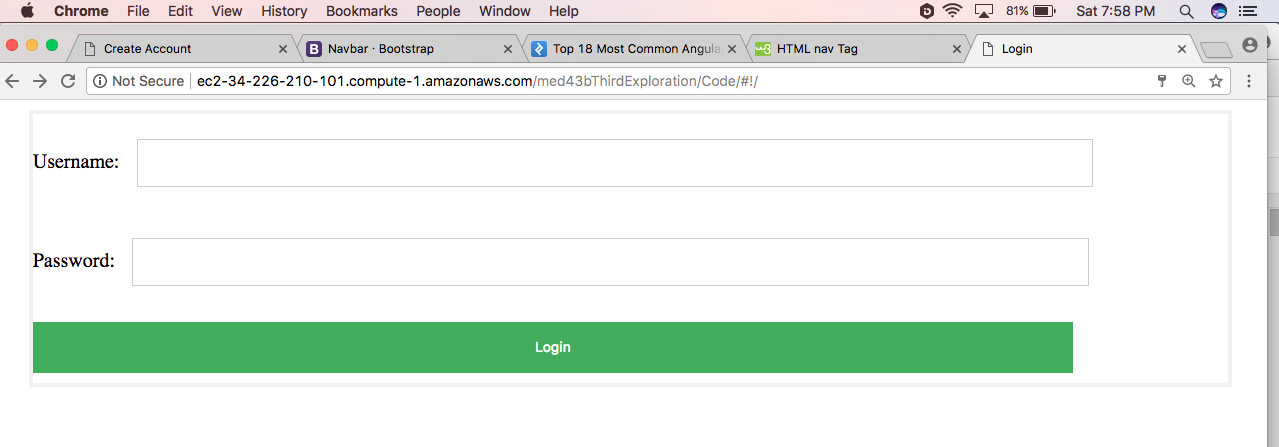
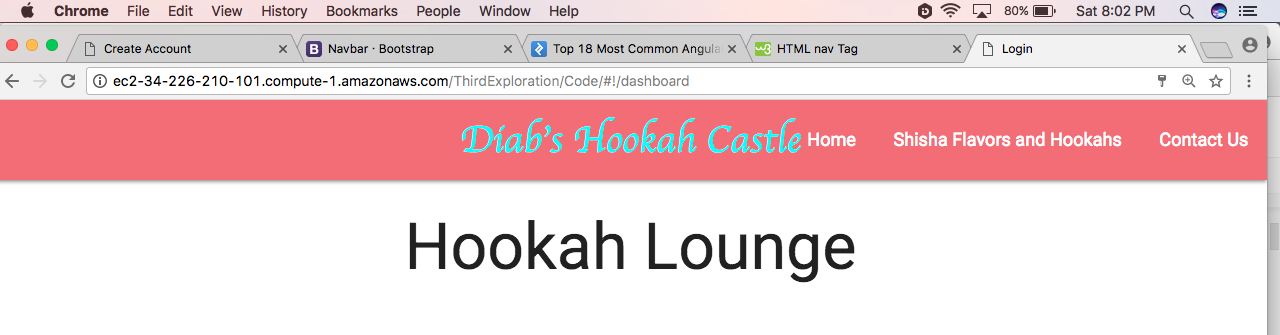
For my third exploration, I decided to merge both Exploration 1 and Exploration 2 together (since Exploration 1 was front-end and Exploration 2 was somewhat back-end). I have a login page which, if you have the right log-in credentials, will take you to my hookah page. I added a hookah and shisha page so it was like a real website.

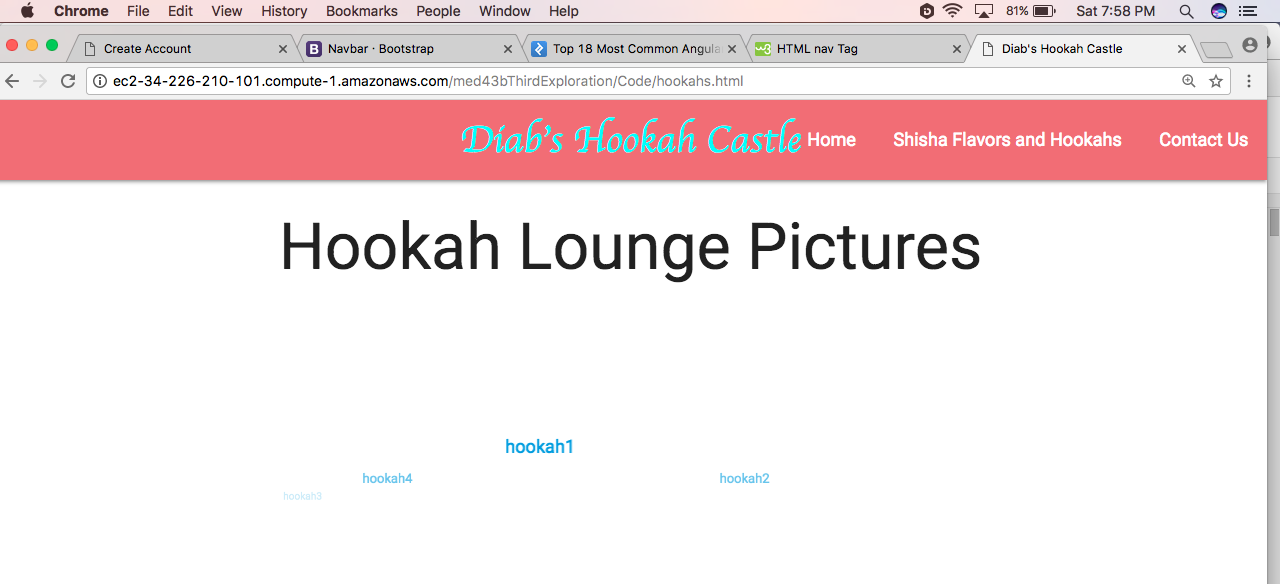
Putting It All Together



This is the new login page. Before, there was no CSS and it was pretty plain. You can login with Username: admin and Password: admin



This is the page you get when you come in from the login page. Like I said, none of the pictures would work at all. I’m not sure why, but it is really stressful. I tried literally everything. Like I said, it would work on the actual html file itself in Brackets, but nothing else, so I know it was going to the right directory.

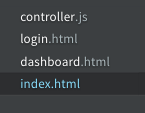


This is the Shisha and Hookahs page. Again, the carousel and alt tags are even showing up, but no pictures.

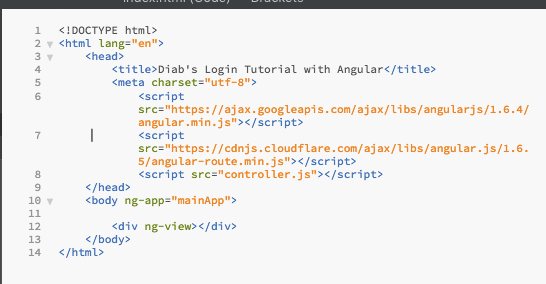
Using AngularJS Login A Database

I chose to use AngularJS as my Exploration. Initially, I made a mongoDB database to correlate with this, but things went south and I started to run out of time. So in this tutorial, I’m going to show you how to create a secure login, and redirect a user back to the login page if they don’t have a login. Essentially, the code is ready to bring in mongoDB database with users and passwords, mind just was not working like I wanted it to so I kept it out.

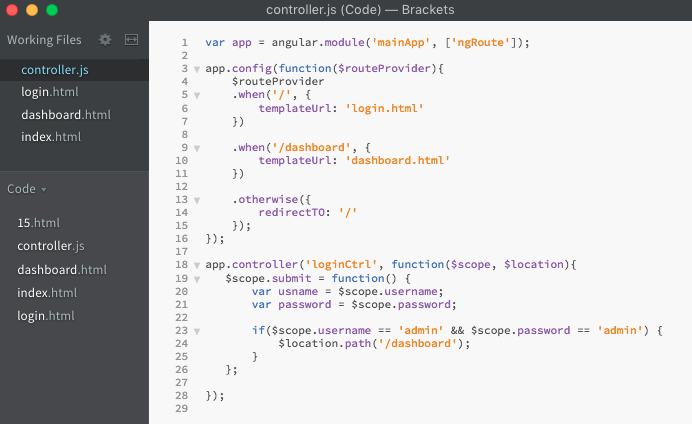
1. Set up all of the files that you are going to need.



1. Set Up Your index.html file And Get your CDN’s to link the AngularJS API, and import controller.js that will control your page functionality using Angular.
   1. I used <https://cdnjs.com/libraries/materialize> to get my CDN’s. I added comments to describe what was what.
   2. Add #mainApp id in order for controller.js to redirect the webpage to that index
   3. Also, create a dashboard.html with a header in order to redirect users after they log in to show that they are actually logged in



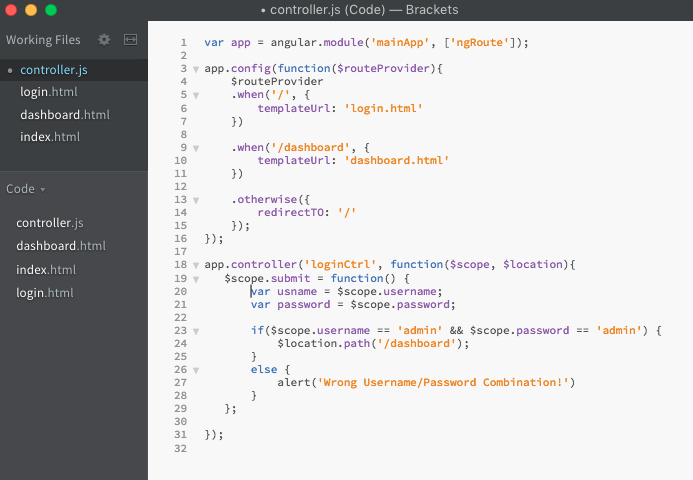
1. Create a Basic Controller Layout To controller.js
   1. I watched a youtube tutorial on how to do this, will explain functionality later on:



1. Set Up Your login.html File To Correlate With Variable and Functions in Controller
   1. We create a form like you normally you, except instead of name, we use ng-model in order for AngularJS to handle the variables from backend

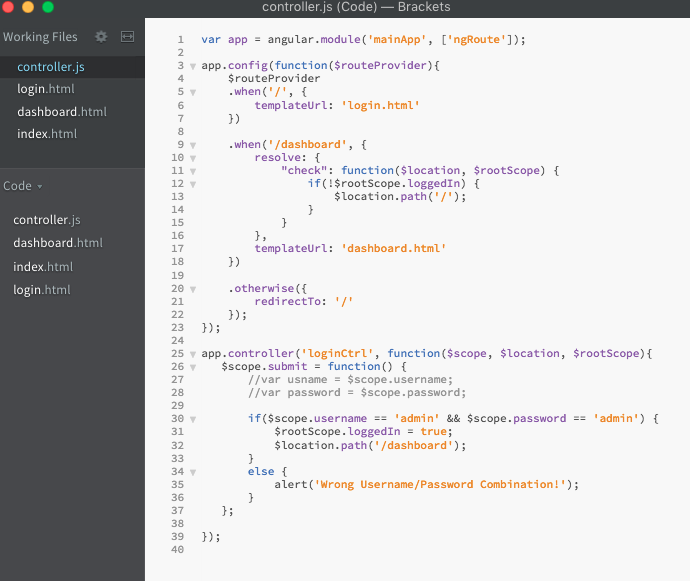


1. Now, We Can Create A Simple Alert If The User Does Not Enter In Correct Credentials



We can now log in as a user with admin as the Username and admin as the Password. But if you notice, you direct yourself straight to the dashboard from the URL, we’re going to fix that now.

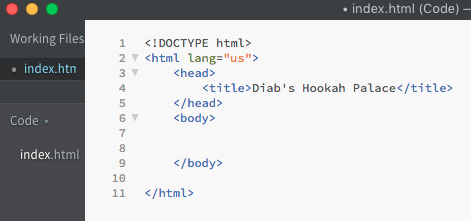
1. Preventing Unauthorized Dashboard Viewing With AngularJS
   1. Instead of just using $scope, we can use $rootScope as a global variable in order to use it throughout the controller file.
   2. Before, when a user logged in, it redirected the user to go straight to the Dashboard with no checks. Here, we will create a check to make sure a user is actually logged in. Make sure you add the $rootScope variable to the function call
   3. In the submit function where we create the log in relationship, comment out the old scope variables. Once the username and password is verified, we will set our $rootScope global variable to true. That verifies a user is actually logged in.



Different Ways of Displaying Images On Your WebPage

I chose to use MaterilizeCSS as my Exploration. I read that Google had a version like bootstrap, and I really like a lot of google's products, so I thought I give it a try. Materialize is just like bootstrap, so it has a lot of the same concepts. If you’ve used bootstrap, this will be a walk in the park. I added a nav bar from the documentation, it's identical to what we did in class.

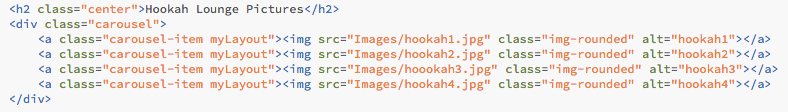
1. Set up your index.html in whatever editor you like, and use the 5 tags we learned about in 2830. I also put lang=”us” in the html tag, but it didn’t say it was required.



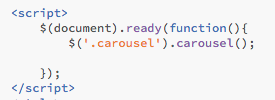
1. Get your CDN’s to link CSS, and import Javascript and jQuery libraries.
   1. I used <https://cdnjs.com/libraries/materialize> to get my CDN’s. I added comments to describe what was what.
   2. While I was at it, I added the viewport to help with mobile browsers



1. Adding a Carousel of Images
   1. To do this, I went to MaterializeCSS.com, and there is documentation on how to do it. I first added the html into the body. I added an h2 to personalize it and added a class (myLayout) to personalize the way my pics fit, but that’s all optional.



* 1. I then added in the jQuery needed for the Carousel to display when the document is ready



1. Another way to add images to your web page is called a Slider
   1. In order to do this, I also went to materializecss.com and looked at the documentation. I had trouble with the images getting cut off, but I couldn’t solve it. I have the link for this in my sources. I also first added the html to the body.



* 1. I then added the jQuery to display the Slider to the existing script tag



Recap:

Major Problems I had with angular and navigating between pages were actually outstanding. I will never use angular to try and make a web page again. I actually ended up stopping because I had gotten so frustrated and there was not anything online that I was having problems with. Navigating between pages caused a get error, which I’m not sure why since I addressed that function in controller.js. Another problem I had was accessing the images in my carousel. It would show up if I viewed (in Brackets) the individual web pages, but when I actually used the full functioning site with the log-in, they wouldn’t display.

I looked up on stack overflow, and I thought I had found an answer to my problem but it didn’t work. It told me to set a base tage <base = “/”> and that’s how angular determined the path it was on. Which was weird to me because that’s what I thought I had done in controller.js. I really would have loved to have finished this because I had a lot of great ideas from class to use. But if the foundation of the site isn’t working, it’s kind of dumb to try and build off of it. I will continue to work on this even though it’s not our assignment anymore, and I will figure it out.

Materialize was easy to use, and really didn’t stray far from Bootstrap. Like I said, I’ve always liked google’s products and wanted to give it a go. The reason I choose different ways to display images on a web page, is because I’m actually going to open my own hookah bar, and I want to have an awesome website. It’s awesome how much code is given to you, and all you have to do is implement it and add the images that you want, also like Bootstrap.

The biggest downfall was the image Slider (the thing I was actually most excited about using) didn’t display the images at the right size. I looked for so much help online, and people were having the same problem as me. I tried the solutions that were given to them, and nothing worked for me. I also got help from a work friend, and he also could not get it to work. The problem was, the height of the image wasn’t tall enough, so it was literally showing ¼ of the image, and the middle of the image at that. Another problem was that the width was too wide. I tried adjusting it using CSS ui-slider class, the actual slider class that was being called on div controlling the slider, actually setting inline style to the image, and even tried to change some jQuery, and nothing seemed to work. I hope to be able to fix this to actually use it for my web site. If I do, I will make sure to update how I did it.

Overall, AngularJS was easy to use, and really didn’t stray far from a regular php file you could set up. I did research beforehand, and it seems like there is the most info about how to use Angular on the web. Although I didn’t get to implement the database, I got the main view of how it will work with backend SQL or noSQL, and it will be easier for me to use when I get it figured out next time.

The biggest downfall was trying to implement the mongoDB database into my web page. What was happening was I could not establish a connection with mongoDB. It wasn’t showing that I had a database when I clearly did have one, and another downfall was the shortage of documentation of using mongoDB with AngularJS. I found a way to connect straight to it, and then it was not storing any users to my database collection. I tried several different ways, and I meant to get screenshots of it, but I restarted and tried so many different ways that I lost track. I ran out of time.

Side note: This is not a secure login at all. I used this as an example on how to create a LogIn using angularJS. Do not try to use this as a secure login to your web page.