**RouteIt Web App homepage Writeup**

For this take-home project I made a web page for a hypothetical travel app where users can select a city and find a database of walking routes that pass by a variety of tourist sights within that city. The user would also be able to create their own routes using sites like googleMaps and TripAdvisor, or request RouteIt to add a route of a missing city via the Contact page. At the footer the user can find other helpful links as well.

While I could have coded the HTML/CSS entirely from scratch, I chose to use the Bootstrap framework to create a simple design that would be responsive to a variety of devices and inherently incorporate the interactive features that JavaScript provides. In addition to using Bootstrap’s grid system, I also took advantage of its carousel effect and simple form layouts using the tab effect. Although the site uses the bootstrap grid system, I edited the carousel images to be the size specified in the grid beforehand so they should be very close. Because this is a mock-page, I didn’t create each page that the links would connect to, however the links have placeholders of # followed by their respective titles.

Since I used Ruby on Rails for the previous project, I decided to use PHP to support access from the browser this time. I would have liked to customize the design of the index.html page much more, however I ran into some issues connecting to my MySQL database via PHP which ended up consuming most of my time. I didn’t end up successfully connecting so I could not test my PHP code, but I provided a back-up so the Login and Sign up buttons could still be tested. Although the working web app does not use MySQL and PHP validation, **please look at the PHP code commented out at the bottom of index.html (and dbconnect.php) to see my work**. This is where I did all of the validation of the forms’ fields, which includes making sure the email is valid, checking to see the required fields are not empty, making sure firstname and lastname have at least 1 character, making sure the password is at least 6 characters, and checking to make sure that the password matches the confirmation password when creating a new user. **Please also see the commented out php code within my html** to see where I would have displayed the error messages that the php validation creates so the user knows if their input is incorrect (or there was another issue).

Since I was not able to connect to my database, so I used php functions to create a makeshift hashmap with arrays. This works for this sample webpage, but for a large-scale web app this would be very inefficient. I definitely would have used the MySQL database if I had been able to connect. Additionally, using this array doesn’t realistically work because every time the page refreshes, the new user values get lost. Therefore for my example, I have loaded a couple of emails that can be used to login and a new email can be added and will be in the array (as shown by the contents being printed, however once you return to the home page this new user will disappear)… My emergency makeshift Hashtable functions don’t check to make sure all of the appropriate fields are filled out, but once again, please see the validation in the commented out php code at the bottom of index.html so see how I originally validated each form. I have also commented out the span and div classes within the html that would have displayed the error messages produced by the validation.

Pre-made users:

Email: test1@email.com Password: pwd1

Email: test2@email.com Password: pwd2

Email: test3@email.com Password: pwd3