CSCI 421 – Advanced Web Development

Lab 5 – Angular Website

For this lab you will start with your Blogger application from Lab 4 as the basis and modify it to move its front-end to AngularJS. All of your front-end code will be placed in a new **app_client** directory. Lastly, you will publish this version of your application to your previously created GitHub repository as a new branch named: "Lab 5".

PUBLIC SERVICE ANNOUNCEMENT: Like the last lab, this lab will take some time. Expect to spend at least 6 to 8 hours developing the AngularJS front-end and integrating it with the REST API you have already written. You can rely on the Manning book and chapters 8, 9 and 10 if you wish to pattern much of your code from it, or you can look online for more simple AngularJS samples (that 2nd step is recommended regardless which way you decide to go). Be warned, the book examples and what is in the author's Git repo for those chapters are not complete and, in many cases, more complex than what you need in a front-end. Because of the reading, the digging, the research needed, the trial and error you will likely encounter, and the coding and recoding (always go for simpler, smaller code) required, this lab will take significant time.

DUE DATE: *This lab is due to the instructor by Monday, March 25th at 11:59pm.*

Instructions:

1. Using your Amazon Lightsail MEAN instance, make a copy of your previous Lab 4 application either using Linux copy commands or GIT commands to pull down the previous application to a new (different than previous) directory.

2. Your application will be similar to the application you wrote in lab 4, it will use the REST API from lab 4, but its front-end will be moved to AngularJS.

3. Your web application must adhere to the following specification:

Specification:

i. **Port 80:** Your application must load from a browser via port 80, which is the standard HTTP port for a server.

ii. **Add an "app_client" directory, port front-end to AngularJS:** Your program should be updated such that it includes all of the AngularJS code in this directory and/or subdirectories. Note, the **app_server** directory will be obsolete and can and should be removed.

The majority of your work will be spent figuring out how best to develop the AngularJS front-end. Below are some hints/suggestions:

- **KISS – Keep It Simple….** You really only need to author two files at a minimum, more if you wish to break things up, but at a minimum the two files below are acceptable:
    1. **index.html** – This is the single-page application front-end, written in HTML and supporting the AngularJS app. It is acceptable for this one file to contain templates for each of your screens.
    2. **bloggerApp.js** - This the angular app code and it includes the "Router Provider", other providers (for example a "State Provider"), the needed controllers and the data access API functions.
- **Develop, test, update, repeat:** Iteratively develop the front-end. First by getting the *index.html* (or similar) written, working with *bloggerApp.js* (or similar) and hooked up to an Angular "Router Provider" so that navigation works with simple controllers. Then, add in the details for each "screen" (list, add, edit, delete), and then hook up the REST API by extending the controllers.

iii. **Download Necessary Angular.js Libraries**: Your front-end will likely require several Angular .js (JavaScript) library files. Download those that are required from a reputable source and place them within your **app_client** folder (or better yet, a **lib** subfolder) and load them from there in your application.

iv. **Consume existing REST API:** Your AngularJS front-end should use the REST API you developed prior for getting a list of blogs, adding a blog, updating a blog and deleting a blog.

v. **Testing:** When finished you will have a single-page application (SPA) that performs the following:

a) Show a list of blogs, allowing the user to click one to edit or delete

b) Add a new blog

c) Edit an existing blog and save it

d) Delete an existing blog (or cancel deletion)

Note: For the add, edit and delete case, your front-end should redirect back to the listing page upon successful completion of the task at hand to show the newly updated listing.

4. Your application must be setup to run even with your MEAN instance is not connected.

5. Using your GitHub account and the repository created and used prior and save this lab as a new branch (“Lab 5”).

6. Below is the instructor’s example from Lab 4. Trust that a Lab 5 instructor’s example does exist. If it were “stood up” and provided here, all the AngularJS front-end code would be readily viewable and for that reason it is not being provided:

   http://3.143.17.202/

7. Once you have the application working and it is available via port 80 even when disconnected from your MEAN instance and you have your app uploaded to your GitHub repo, please send the full URL of your app AND your GitHub repo to the instructor via email with subject of “Lab5”. Important:

   mailto:thomas.rogers@millersville.edu?subject=Lab5