**CAS CS412 Simple Web App**

This assignment is relatively large as it combines both front-end and back-end pieces. Broadly, you'll submit a form from an Angular application that is processed by a Node route. That route should call a third-party API to retrieve data related to the form's fields, process it, and return information to Angular.

1. **Angular**. Write a simple Angular application that queries a Node route.

   **a.** Use a **validated form** [require the input term and validate it is longer than 1 character] and a 'query' button on your Angular app to initiate the request, passing any search term that you might need for your back-end API.
   **b.** On the Angular side, use a **service** to make the call to the back-end Node route.
   **c.** Use **three** Angular components. The first component should hold the form, and the second should display the result of the query. Use @Input in the child to receive data from its parent for display. The third component will enclose and be the parent of the other two.
   d. When displaying data, use *ngIf and *ngFor to parse through an iterable containing your response data. When no data is present, *ngIf should hide that portion of your page. Note: If your back-end source returns just one item, duplicate it on the back end to present an array of data to the front end (just a few duplicates needed to demonstrate *ngFor).
   d. When displaying data, include an indicator to show whether the data was retrieved from cache on the backend, or directly from a third-party API call.

2. **Node**. On the Node side, choose a third-party API and provide routes that offer queries against it.

a. Use **redis** to cache data returned from the API for 60 seconds.
b. The route that receives form data from Angular should be a POST route; the API you choose will determine your calling strategy to it.

c. Use async/await and the **http** client for calls to your API.
d. Use config files to handle any hard-coded information (URL endpoints, keys, and so on).

3. Submit your component Typescript files and their associated HTML files, and any data classes from Angular, plus your route file from Node, on Gradescope to complete the assignment.

**Extra bits**

While we won't be grading on these items, if you'd like to dig a bit you can add extra features to your app. The two big ones would be to include OAuth logins on the back end and to use Angular Material to make the front end fancier.

**A note about choice of API**

It's helpful if the API you choose returns a data set that has different levels of information, to be able to drill down to get detail information. It also is useful to choose an API that returns an iterable set of data. An example of both might be to retrieve the weather for three cities, and each city has temperature, pressure, and humidity information.

Also, some APIs come with rate limits (ie 5 calls per day), cost, or take significant time to apply for and receive a key. Try to find a data set that satisfies everything above but also is free and offers instant access. You won't have time to wait for a key. For example, I applied for a last.fm dev key last semester, and still haven't received it.