ICP 6: APIs and Angular Routing

Joe Moon
Email: jmn5y@umsystem.edu
Github: https://github.com/joemoon-0/WebMobile-2022Spring/tree/main/2022Spring-Web/Web_ICP6

Nathan Cheney
Email: ncxn8@umsystem.edu
Github:

Introduction

Web applications utilize APIs to load dynamic information onto a webpage by making a request to a server for that information. When combined with forms for receiving user input and Angular routing components, web apps can be greatly expanded to display a variety of customized information which this application will demonstrate.

Recipe and Restaurant Search

From the application's root component, a basic navigation bar is displayed to help users access different functions. Clicking on a button (in this case, "Recipes"), routes the user to the appropriate component. In addition, a wildcard route is added to handle erroneous urls.

```
7  const routes: Routes = [
8    { path: '', redirectTo: '/home', pathMatch: 'full' },
9    { path: 'home', component: HomeComponent},
10   { path: 'recipes', component: RecipeSearchComponent },
11   { path: '**', component: PageNotFoundComponent }      // Wildcard Route
12 ];
```

app-routing.module.ts

The recipe-search component uses two API calls to generate the information that will be ultimately displayed on the UI. Based on the user input fields, a fetch call is made to the edamam API to receive recipe information.

```
46    if (this.recipeValue !== null) {
47      fetch(`https://api.edamam.com/api/recipes/v2?type=public&q=${this.recipeValue}
        &app_id=49ade2ef&app_key=3224e517a474cf2eb69f977986751d47`)
48      .then(response => response.json())
49      .then(data => {
50          (data.hits).forEach((result: any) => {
51            const recipeResult: IRecipe = {
52              name: result.recipe.label,
53              url: result.recipe.url,
54              icon: result.recipe.image
55            }
56            this.recipeList.push(recipeResult);
57          })
58        }
59      );
```

recipe-search.component.ts

If a user enters a location input, a second API call is made to the foursquare API to generate information on venues that serve food similar to the recipe/food that the user entered. Note that version 2 of the foursquare API which accepted a string for location (e.g. Kansas City) was deprecated.

Version 2 deprecated. Requires version 3

Version 3 requires geocoordinates. An attempt was made to convert a location string into geocoordinates using a third API (mapbox) however the different formats could not be reconciled. For demonstration purposes, **this.geocode** has been hardcoded for Kansas City.

```
25    convertLat: any;
26    convertLong: any;
27    geocode: string =  encodeURIComponent("39.097,-94.58"); // Kansas City
```

```
80      const options = {
81        method: 'GET',
82        headers: {
83          Accept: 'application/json',
84          Authorization: 'fsq3GKoQk47nPZ0w1boKmB8FalPYTevOzbAOy4p8m2rWo2w='
85        }
86      };
87
88      fetch(`https://api.foursquare.com/v3/places/nearby?ll=${this.geocode}&query=$
        {this.recipeValue}`, options)
89          .then(response => response.json())
90          .then(data => {
91            (data.results).forEach((result: any) => {
92              const placeResult: IPlace = {
93                name: result.name,
94                location: result.location.formatted_address,
95                uriCoordinates: encodeURIComponent(`${result.geocodes.main.latitude}
                  ,${result.geocodes.main.longitude}`)
96              }
97              this.venueList.push(placeResult);
98            })
99          });
```

recipe-search.component.ts

Two interfaces (IRecipe and IPlace) are used to populate the venueList and recipeList arrays which are subsequently used to display the API results on the UI.

Contribution

The code shown in this report is from Joe Moon's repository.