TEAMMATES:

Pavan Kumar Jonnadula - 16324822 - pvjb6c@umsystem.edu

Vamsi Alapaty - 18230326 - vanny@umsystem.edu
Priya Varsha Tarlapally – 16332852 – ptf2x@umkc.edu

ICP6 – Report

Programming Tools:

GitHub, VS Code

Source Code:

Pavan – 16234822
https://github.com/JVSPAVAN/WebMobile-2022Spring/tree/ICP5/Web/ICP6

Vamsi – 18230326
https://github.com/VamsiAlapaty/Web-Mobile-Spring2022/tree/ICP6

Priya - 16332852
https://github.com/Privar11/Web_Mobile_Programming_ICP/tree/main/Web_Programming

Objective:

This task concentrates on understanding the concept of REST API in angular. We have a tutorial to explain the concept.

Task:

1. Creating an Angular application which displays nearby restaurants (Hint: Use FoursquareAPI).
2. Creating an application in Angular which displays recipes (Hint: Use EDAMAMAPI) If possible, combine both the APIs in a single application. (Given starter code has both together. Change the UI if you are going use this)

Steps done to achieve the above task:

1. Obtain the client ID and client secret from the Four square API.
2. Obtain the keys from the Edamam API.
3. Using http client call we have obtained the response in JSON format
4. Parsed JSON and appended to HTML using angular directives.

A detail description of the achieved tasks for Angular application which displays nearby restaurants and recipe

A service file is used to place all the API calls. This service file is imported into the 'ts' file. Using 'window.navigator.geolocation.getCurrentPosition()' function to obtain the latitudes and longitudes of the user current location.

Recipe Finder:

When we call the Edamam API using the HTTP Client, a JSON data is obtained in the response. API_key and API_Id are used to call the API. An array of recipes is obtained which is parsed and kept into an arraylist. These recipes are displayed in a list format in the HTML.

```
getrecipes(client_id,client_secret,recipevalue): Observable<any> {
  return this.httpClient
    .get<any>('https://api.edamam.com/api/recipes/v2'+'?&type=public&app_id='+client_id+"&app_key="+client_secret+"&q="+recipevalue)
    .pipe(
      map(
        response =>
          response
      )
    );
}
```

Once we click on one of the items, a modal window is used to display the details of the selected recipe. This window contains the details like

- recipe calories
- servings
- list of icons showing recipe tags
- ingredients
- website link for full recipe
- table of nutrients

Restaurant Finder:

When we call the Foursquare API using the HTTPClient, a JSON data is obtained in the response. Here Client_secret and Client_id are used to call the API. In addition to these, a API key which is generated in the foursquare API is used as part of authorization in the headers. An array of restaurants is obtained which is parsed and kept into an arraylist. These restaurants are displayed in a list format in the HTML.

```
24  getrestaurants(client_id,client_secret,recipevalue,placevalue): Observable<any> {
25    let headers = new HttpHeaders()
26      .set("Authorization", "fsq3rrCognvaRF7OppJlfX7CmQDsYwnil8QHH7nuhmQzpCI=");
27    const httpOptions = {
28      headers: headers
29    };
30    return this.httpClient
31      .get<any>('https://api.foursquare.com/v3/places/search'+'?client_id='+client_id+"&client_secret="+client_secret+"&query="+recipevalue+"&near="+placevalue+"&
      v=20220220",httpOptions)
32      .pipe(
33        map(
34          response =>
35            response)
36      );
37    }
38  }
```

Once we click on one of the items, a modal window is used to display the details of the selected restaurant. This window contains the details like

- full address of the restaurant
- restaurant type
- distance from the user location
- link to directly find the route to the restaurant.

Screenshots: