Mobile App Development
Assignment 15

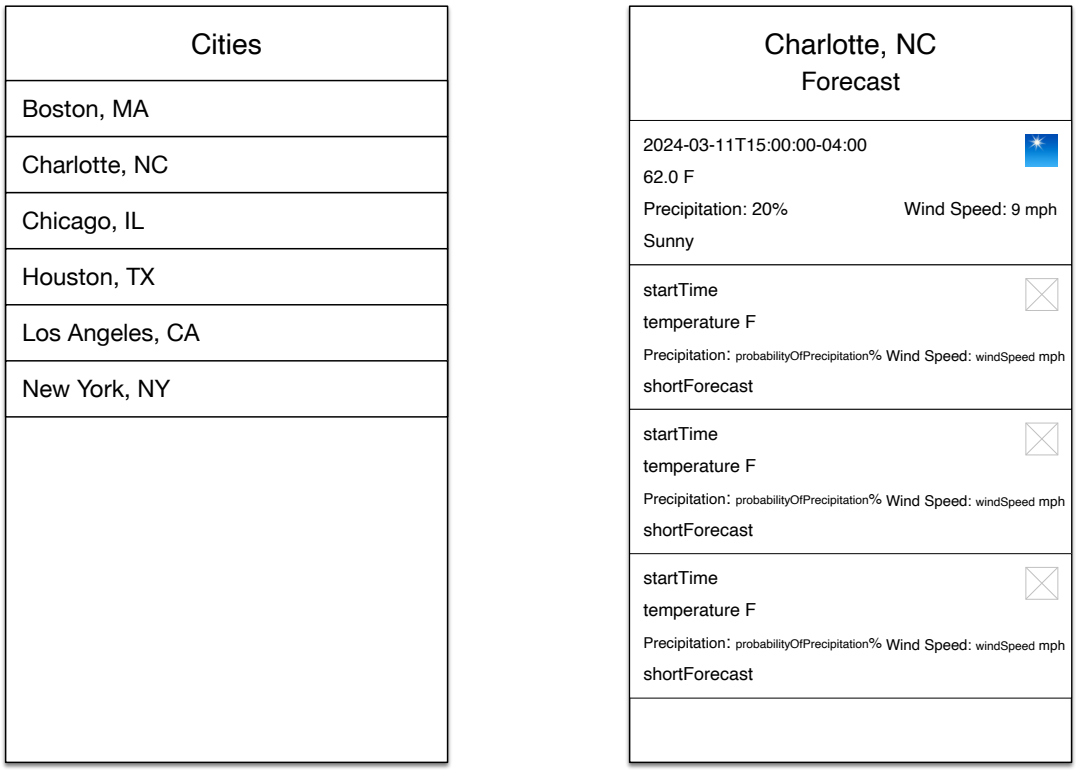
Basic Instructions:

1. In every file submitted you MUST place the following comments:
   a) Assignment #.
   b) File Name.
   c) Full name of the student.
2. Each group is required to submit the assignment on Canvas.
3. Submit Codes:
   a) Zip all the project folder to be submitted on canvas.
4. Submission details:
   a) The file name is very important and should follow the following format: **Assignment#.zip**
   b) You should submit the assignment through Canvas: Submit the zip file.
5. **Failure to follow the above instructions will result in point deductions.**

Assignment 15 (100 Points)

In this assignment you will be building a weather forecast app that makes REST API calls. The app requirements are as follows:

1. Please use the provided skeleton app.
2. All data should be retrieved in JSON format.



(a) Cities Screen **(b) Weather Forecast Screen**

Figure 1, Application User Interface

Part 1 : Cities Screen (30 Points)

The interface should be created to match the UI presented in Figure 1(a). The requirements are as follows:

1. This screens should retrieve the list of cities by using the following API:
   a. Make a GET request to https://www.theappsdr.com/api/cities
   b. Create a City class that should hold the data returned in the JSON API response.
2. Display the city name and state as shown in Figure 1(a).
3. Clicking on a city row item should transition to the Weather Forecast screen and should pass the required information to it.

Part 2 : Weather Forecast Screen (70 Points)

The interface should be created to match the UI presented in Figure 1(b). The requirements are as follows:

1. This screen receives the City object which includes, the city name, latitude and longitude.
2. In order to retrieve the weather forecast for the provided city it will require calling two apis.
3. **Step 1:**
    a. Make a GET request to https://api.weather.gov/points/{latitude},{longitude} for example, https://api.weather.gov/points/35.2033219,-81.0049789
    b. From the returned JSON response, you need to retrieve a forecast url which is present at properties.forecast in the returned JSON.
    c. Trigger Step 2.
4. **Step 2:**
    a. Make a GET request to forecast URL received from step 1, for example, https://api.weather.gov/gridpoints/GSP/113,63/forecast
    b. Create a Forecast class that should hold the data returned in the JSON API response.
    c. Display the list as shown in Figure 1(b).