Jason Lee
MSDS 440
Project: Phase 1

1. **Examine the data source for Yelp reviews, and explain how we can filter business (restaurants, bars, coffee shops, etc) for the Chicago downtown area only. What keywords should be used to filter relevant places?**

In order to keep our Yelp reviews focused in the Chicago downtown area, we should filter our searches using a parameter for locations based on their Zip Codes. Any result we return for our application needs to be within the Zip Codes for the downtown Chicago area.

2. **We use Yelp API (Links to an external site.) to search for businesses with categories='Restaurants+Entertainment+Nightlife', do you think that is sufficient? Explain.**

I would also include 'Food' into the categories search parameter. I was looking through the category list on yelp's API and it looked like 'Restaurants' didn't include many other food options like coffee shops, ice cream, desserts, etc. I would have the search be something like this for categories='Restaurants+Entertainment+Nightlife+Food' to include these places. They are all social activities to do in the city that should be included in our application.

3. **We are interested in businesses that are located in the Chicago downtown neighborhoods like Magnificent Mile, Gold Coast, Old Town, Loop, South Loop, West Loop, etc. Visit this Chicago Zip Codes resource (Links to an external site.)and write the most relevant zip codes that we should use to search for Yelp reviews.**

This is the list of Zip Codes that we should use in our application:
60611, 60605, 60605, 60610, 60611, 60601, 60602, 60603, 60604, 60605, 60606, 60607, 60616, 60611, 60605, 60610, 60611, 60642, 60654, 60606, 60607, 60608, 60610, 60612, 60661, 60601, 60622, 60610, 60605, 60607, 60661, 60611, 60611, 60654, 60622, 60610, 60605, 60607, 60608, 60616, 60611, 60612, 60612, 60622, 60607, 60612, 60622, 60642, 60647, 60622

These Zip Codes cover these areas in Downtown Chicago:
Cathedral District, Central Station, Dearborn Park, Gold Coast, Loop, Magnificent Mile, Museum Campus, Near North Side, Near West Side, New East Side, Noble Square, Old Town, Printers Row, Randolph Market, River East, River North, River West, South Loop, Streeterville, Tri-Taylor, Ukrainian Village, West Loop, West Town, Wicker Park

4. **Divvy publishes its data (Links to an external site.) in near real-time for the status of its docking stations in Chicago, examine the published data for 3 docking stations for an hour time-range. How many times Divvy updated the status for the docking stations you selected for the hour time-range. How many docking stations are in the city of Chicago? Is there an exact street address for the docking station?**

I looked at docking stations in 3 different areas around downtown Chicago. I decided to look at a docking station along the Magnificent Mile on St. Clair St & Erie St. Then another docking station in Printers Row on State St. & Harrison St. The last location was by the Navy Pier on Streeter Dr & Grand Ave. The docking stations moved on average around +- 6 available docks over the hour. The docking stations updated 8 times, 5 times, and 12 times respectively within the hour.

Looking at https://gbfs.divvybikes.com/gbfs/en/station_status.json I saw a max station_id number of 673. I am assuming this is only including the stations in the Chicago area. There are street crossroads but not exact street address. However, there is a latitude and longitude in the data for each docking stations making it easy for us to map out the locations accurately.

5. **The Python process, running on the DSCC server, will pull the real-time data every 2 minutes from Divvy servers, do you think that is acceptable time-window to send the heartbeat to pull the data? Should the Python process pull data every 1 minute (smaller) or 5 every minute (bigger) rather than 2 minutes? Explain.**

Because the Divvy servers only publish their data every 2 minutes it would be pointless for us to try and pull the data more often than that. The 2 minute heartbeat Python process will suffice for our needs. This will give us the most up to date available information on the docking stations from the Divvy servers. It would be nice if the Divvy servers were able to update their data more often but this is good enough.

6. **When the Angular client requests Divvy docking stations status from Node.js/Express server, would the data received by the client be categorized real-time or near real-time data? Explain.**

The data that is returned from the Node.js/Express server from the Angular client requests would be classified as near real-time data. The reason that this is near real-time and not real-time is because the Divvy servers are only published and updated every 2 minutes. There is a slight delay between the events and the collection of the data meaning that it is near real-time but not actually real-time. In our use of the data, this slight delay should not negatively impact our users. The user will be able to use the total number of spaces available at the docking station to help them make their decision about where they will dock.

7. **After you examine the requirements, architecture, and data sources for the ChicagoSocialHub app, do you think we should use the pull-method (Angular client periodically pulls real-time data from Node.js/Express server) or the push-method (Node.js/Express application server periodically pushes the real-time data to Angular client)?**

I'm still not entirely familiar with the behind the scenes workings of the specific application that we will be creating for this class and I'm brand new to these push/pull methodologies… but from what I've gathered from the demo videos, readings, and online resources, I believe that we could use a Pull Method for the Chicago Social Hub screen where you search for the food and location. I believe the Pull Method would be perfect here. It will take in the request from the client/user and will go and get the needed data from the Node.js/Express server and return exactly what is needed.

Then for the heatmap and Divvy docking stations real time charts I believe could be a Push Method to periodically use the Node.js/Express application server to push the real-time data changes to our Angular client. This is beneficial when there are asynchronous data updates. Instead of needing to pull the entire request over from the server, we will be able to push out the changes when they occur and the application will continue to update.
Since I'm very new to this whole process, I'm not sure if it possible to do both of them within this single application.