Final Project: Web Service APIs

This final project should serve to bring together almost all of the topics we have been covering throughout this class, from manipulating tidy data to designing and storing data in a relational database to working with hierarchical data to the underpinnings of Client-Server computing and working with Web APIs by providers.

It is also intended to be self-defined. Do some exploration and find data from a provider that you are interested in investigating.

Requirements

The requirements of this project include the following:

1. Work with a Web API designed by a Data System provider.
   - The selected provider must require Authentication and Delegated Authorization (OAuth 2) to enable access to data not available to an anonymous user for at least some subset of the data.
   - The API must use HTTP for its requests and responses.
2. Find a second data source, which could be openly provided, or could be obtained through web scraping techniques, that complements the data from 1.
3. Use the skills/knowledge from the Hierarchical Data Models unit to parse and extract tables of information from XML or JSON from your primary provider.
4. Build a relational database from your data sources.
5. Practice good software development techniques:
   - Practice functional abstraction and associated code documentation
   - Program defensively, checking for error codes/returns in *every* client/server interaction, and handling the error in a reasonable way
6. Give the acquired data meaning through exploratory data analysis and visualization to ask some questions interesting to the student group.
7. Prepare a 4 minute presentation on your results to be given on the last day of class.

As an alternative to item 1, a student group could, with instructor permission, use data obtained from a set of HTML tables populated by navigation on multiple web sites, and using the techniques of Web Scraping. Since this would typically not involve OAuth techniques, such a project would need to include some additional complexity by, for instance, using POST operations to mimic a web user filling out a form and performing some kind of submit operation to obtain the HTML data.

Process

Students will work in randomly generated teams of 2 on this project, and the project will be due the last day of class by **class time**. Students are **strongly encouraged** to start early and to document their progress through a journaling notebook.

The project specifics are left to the creativity and design by the student group, but must satisfy all of the above goals. Below, I will list some various providers that I have explored in the past. Explore the Providers and APIs, perhaps coming up with providers I have not listed, and think about data that excites/intrigues you.

A minimum standard for questions/visualization/data exploration is three visualizations of moderate complexity that depict the data in a way conducive to answering an interesting question.

Submission

For a typical submission/progression of this project, I would expect organization into three notebooks:

1. Token acquisition notebook: this notebook handles the code cells/steps needed to get an access token. For the most part, these steps only need be executed one time, to go from first registration of a client with a provider to the point where an access token is received (nominally Step 8 of the OAuth Dance). If tokens need to be refreshed, then the cell(s) needed to refresh a token could be here as well. If the secondary data source is open, or only needs an App ID/key, then this notebook will only entail the primary data source.
2. Data acquisition notebook: this notebook starts from the premise of a valid token/api id/key and takes the reader through the set of interactions acquiring and organizing the data from the provider. The notebook should detect and handle errors from the server, including, but not limited to, handling an expired token. There should be functions abstracting the work needed to get data from a provider endpoint, functions for organizing the data into normalized dataframes, and functions for dropping/creating tables and populating database tables with the results.
3. Results presentation notebook: this notebook documents, for a non-expert audience, the results of the data exploration, with the same standard of narrative and data presentation clarity as previous projects. In this case, since the input source to Tableau will be an SQL connection, any JOINS and the equivalent of GROUP-BY will likely occur in Tableau, but should be **documented** in this notebook.

For the first two notebooks, I additionally want you to describe and document the various successes and challenges the team encountered in finding solutions to the token access process and the correct interaction with the Service API for data acquisition. This is like a diary/project notebook, but should be **organized** and not just a date by date brain dump. It may also detail exploration of APIs with interactive tools available through many providers.

Data Providers

1. Facebook

   - **Graph API** allows querying and posting from a program and includes ability to get lists of friends and many other

2. LinkedIn

   - Uses OAuth 2
   - RESTful APIs

3. Fitbit

4. Tumblr

5. GitHub

6. Twitter

   - Note that this uses OAuth 1, not 2, so there will be additional learning curve
   - There are streaming APIs, but these will require additional complexity on the programming side, and to

be useful, you will want to start early so that you have sufficient time to collect data on a stream to allow for interesting analysis

7. Dropbox

    - Data is relative to whatever one deposits into Dropbox, so this will need something additional to generate interesting data.

8. Google Drive

    - Same comment as for `Dropbox`