CS 495 - Assignment 3 Part 2

Rosa Tung

Having finished the API, what you would have done differently

This was a learning experience for me so I wouldn't do anything differently. I think I've learned all the major things that the professor meant for us to learn in doing this assignment so that is what is important at the end of the day. I was able to implement the four major HTTP verbs properly with the associated entities (with the cascade delete), and implement cURL test cases correctly . I'm sure this program isn't perfect and it wouldn't be hard to find stuff that sucks about it, but I've succeded in implementing all the core elements and learning all the tools to move forward. (but if you see anything I'm missing please let me know) Anyways I'll talk about what I want to do next, here are the next three things I would do to improve upon this program if I had more time.

1. Implement update/caching/succeed in making it officially and fully RESTful.
   As I talk about below, this program is far from being RESTful so going forward with that is most important to me now.
2. Implement authorization and authentication.
   I probably will need this in the future if I want to be serious about releasing this API, although I probably won't because there are probably a ton of APIs that offer this information and more. I was more interesting in doing automatic graphs maybe since that is usually the part I'm more interested in when looking to see stock trends.
3. Each stock should only be associated with one country.
   Through testing I've forgotten that I never added in this constaint. Right now it is possible to just keep adding the same stock or one stock to multiple countries, I would want to add in some constraints so you can't do this, which would better reflect how the stock market is in the real world, although I'm not sure if you're allowed to have dual nation ownership of a stock.
4. Allow user to see all nations like how it is with companies
   For the company you are allowed to see all of them, but for nations you can't do this. If I had more time I would implement it for both, as well as other implementations that would be useful. Maybe a way to see a stock and it's country in the same line as well, that would be very useful.
5. Clean up admin side
   Trying to get everything to work I have a lot of random debugging lines (especially if you do a delete of a country that needs to cascade delete) so I would want to go in and clean those up and output cleaner, more succint lines.

A description of which RESTful constraints you met and which you did not

So as you can see already from my code while I do have a spot for updates I don't actually have updates working correctly. It doesn't have the URI's navigate or any cache properties sent out which is another thing I did not get to implement. So it's not cacheable (resources retrieved must be marked as cacheable or non-cacheable) However I do have things such as using POST to create a resource, PUT to change the state of a resource, GET to retrieve a resource, and DELETE to remove it which helps modularize things which is considered to be a RESTful quality, so I believe this results in a sort of layed system as well as forging a path for fulfilling the client-server constraint. I don't have any sessions but since I don't have updates working correctly I'm not sure if it's actually stateless in the way that the REST requirements are talking about. There is in fact no memory being used on the server to maintain what a client is currently doing, but usually you would have another way (updates) to get some information at least that are tied to the resource as mentioned in lecture. I believe there is uniform interface because the client can request different represenations of an order, even if they can only request one type which is application/json. But I do accept it, get the information and modify it to that kind of representation that I promise, and give it in that form to the user.

A discussion of any changes you needed to make to the schema from last week

So surprisingly I didn't need to make any changes to the schema from last week. I was able to create two kinds of entities, a stock entity and a country entity and have the country be associated with a list of stocks. The program I have now is on it's way to doing that. I almost want to add more attributes and relationships and entities to the scheme I had.

An Explanation of the URL Structure used to Access Resources

So the whole idea of my API was a getting information one. For context I used the GAE datastore and if you reference my assignment 3 part 1 it's basically a list of stocks and then a list of countries and the stocks are each associated with one country. I tried to make the structure of my URLs for accessing resources make sense. I have only ever worked with one API which was the Riot API which was also resource based. This was similar because in the end you only want the user to be able to retrieve information, not alter it in any way. So from a user perspective, you can search for a company or nation by the id.

http://marine-service-127317.appspot.com/company/{id}
Example: *http://marine-service-127317.appspot.com/nation/5631986051842048* will give you
{"key": 5631986051842048, "companies": [5749328048029696, 5668600916475904], "updates": [], "nname": "Rosa's Company"}
http://marine-service-127317.appspot.com/nation/{id}
Example: http://marine-service-127317.appspot.com/nation/5631986051842048 will give you
{"key": 5631986051842048, "companies": [5749328048029696, 5668600916475904], "updates": [], "nname": "Rosa's Company"}

From the admin side, I do have more URLs where you can
1. add or delete a company (will also remove a company from a nation)
2. add or delete a nation
3. make a company
4. make a nation with or without a set of companies
5. add a company to a nation's company set
6. delete a company from a nation's company set

I did these, and they can be done form an admin side, with the following cURL requests, all of which are in the curlTests.txt file, which contain my tests and the results. Below are a couple examples

Make Company (Post)
curl --data "cname=Facebook Inc" --data "symbol=FB" -H "Accept: application/json" http://marine-service-127317.appspot.com/company'
Delete Company (Delete)
curl -X DELETE -H "Accept: application/json" -H "Content-length: 0" http://marine-service-127317.appspot.com/company/deleteCompany/{id}
Add Company to Nation (Put)
curl -X PUT -H "Accept: application/json" -H "Content-length: 0" http://marine-service-127317.appspot.com/nation/{nid}/company/{cid}
Search Company by Name (Get)
curl --data "cname=Twitter Inc" -H "Accept: application/json" http://marine-service-127317.appspot.com/company/search

tests and results in curlTests.txt