Assignment 4
Pavin Disatapundhu

URI: cs419a3.appspot.com/customer
/customer/<customer id> to access customer info
/customer/search for search

URL structure:

The non-relational database API was implemented on google App Engine using python and NDB. The API doesn't consist of any user interface, but data can be viewed in through using the URIs. The idea behind this database was to create a simple hotel and customer database that shows the relationship between customers who have stayed at a specific hotel. Originally the database was suppose to support hotel book features, more on that later. In this API the user can create save the customer name, phone number and the customers email. Then the user can assign those customers to a hotel, assign a name to the hotel and specify the type of rooms available in each hotel.

In the schema there can be multiple customers to a hotel. The customer entity can be create separately from hotel. The hotels can also be create separately for the customers. However, the current state of the program only allows the user to add customers to a hotel during the creation of the hotel entity. The structure of the database simple; currently the Hotel is the root is the hotel and the child are the customers assigned to those hotels. An example of the data is represent below:

cs419a3.appspot.com/hotel/customer

RESTful:

This API satisfies the RESTful style by having an serve-client relationship, stateless, a layered system, and is uniform interface. Although non of these RESTful topics are currently implemented fully. The first thing is client-server relation, this is currently implemented, but because the nature of the project, interface was not fully implemented. However, computation is strictly occurs on the server side and the client side is only there from the UI, in which is currently handle through the terminal. Similar to the explanation in the client-server relationship. This program is also stateless because there is no state save to the client side that retains the information about the state. In other words it doesn't rely on sessions, once data is passed to the cloud server, the server computes the data and send the data back to the user browse or terminal.

The systems is also layered, as state early computation occurs separately and at different stages of interacted data is computer and the server side and sent back to the user side. Although this system is not very complex, once the client has requested data and received it does not need to compute any other data until another request is made. Lastly the system also has a uniform interface. Although the scope of this assignment isn't large there are multiple ways to access the resource and even manipulated it. The first way is directly in the web browser and the second through the Unix terminal.

Changes:

I originally intended to have 3 different entities for this non-relational project. Originally the 3 entities were the room, booking and customer entity. After carful consideration I realized the original scheme would be more reasonably represented in a relational database. I decided that from the first project I would keep the scope smaller, and decided to model the relationship of a hotel and multiple customers.

Different:

If I had to do the project I would like to add additional entities and actually make a hotel book application. I would like to implement more functions now that I have a better understanding of NDB and python. I would also like to implement error detection features, and features that will make the program have a more uniform interface. I would also like to implement a user interface.