Mackenzie Holznecht
Assignment 3, part 2

You should have a live functioning API that is publicly accessible or that you provide the needed credentials to access.
http://holznecm-assign3.appspot.com

You should provide a full set of tests. These could be curl calls, a python script, commands entered into a browser plugin etc. It should show that you are able to do all the above actions. Additionally make sure they show that when an element is deleted, references to that element are cleaned up as well.
See curlTests.txt in Zip
It's very limited as I've struggled this entire few weeks, so I'm just surprised I was able to get anything working!! This is a huge mental victory for me ☺

You should provide the results of those tests including explanations for any tests that failed. (Things should ideally not fail, but if they do, it is better to explain it than to ignore it).
Again, VERY limited, but here's what I have so far. John Smith was added with the curl test, Mackenzie Holznecht is in as a patient from manually entering it into the datastore on the GAE platform. Great to see it is also returned with the newly added John Smith. Then I attempted to associate a patient with an insurance – no success.

cURL tests:

```
REQUEST: curl –data "username=John Smith" –H "Accept: application/json" http://holznecm-assign3.appspot.com/patient
RESPONSE: {"username": "John Smith", "dob": null, "insurance": [], "key": 5707702298738688}
```

```
REQUEST: curl –H "Accept: application/json" http://znecm-assign3.appspot.com/patient
RESPONSE: [{"username": "Mackenzie Holznecht", "key": 5629499534213120}, {"username": "John Smith", "key": 5707702298738688}]
```

```
REQUEST: curl -X PUT -d "" -H "Accept: application/json" http://holznecm-assign3.appspot.com/patient/5707702298738688/insurance/5649391675244544
RESPONSE:
<html>
 <head>
  <title>Internal Server Error</title>
  <style>
   body {
    padding: 20px;
    font-family: arial, sans-serif;
    font-size: 14px;
   }
   pre {
    background: #F2F2F2;
    padding: 10px;
   }
  </style>
 </head>
 <body>
  <h1>Internal Server Error</h1>
```

```
    <p>The server has either erred or is incapable of performing
    the requested operation.</p>
    <pre>Traceback (most recent call last):
  File
&quot;/base/data/home/runtimes/python27/python27_lib/versions/third_party/webapp2-
2.5.2/webapp2.py&quot;, line 1535, in __call__
    rv = self.handle_exception(request, response, e)
  File
&quot;/base/data/home/runtimes/python27/python27_lib/versions/third_party/webapp2-
2.5.2/webapp2.py&quot;, line 1529, in __call__
    rv = self.router.dispatch(request, response)
  File
&quot;/base/data/home/runtimes/python27/python27_lib/versions/third_party/webapp2-
2.5.2/webapp2.py&quot;, line 1278, in default_dispatcher
    return route.handler_adapter(request, response)
  File
&quot;/base/data/home/runtimes/python27/python27_lib/versions/third_party/webapp2-
2.5.2/webapp2.py&quot;, line 1102, in __call__
    return handler.dispatch()
  File
&quot;/base/data/home/runtimes/python27/python27_lib/versions/third_party/webapp2-
2.5.2/webapp2.py&quot;, line 572, in dispatch
    return self.handle_exception(e, self.app.debug)
  File
&quot;/base/data/home/runtimes/python27/python27_lib/versions/third_party/webapp2-
2.5.2/webapp2.py&quot;, line 570, in dispatch
    return method(*args, **kwargs)
  File &quot;/base/data/home/apps/s~holznecm-
assign3/20161023t211503.396552107235148730/patient.py&quot;, line 108, in put
    if not patient:
UnboundLocalError: local variable 'patient' referenced before assignment
</pre>
  </body>
```

I'm not sure why I got this error. But I'll keep trying to figure it out.

This should be hosted on some variety of cloud platform and use a non-relational database to store the data.
Hosted on GAE using the GAE datastore.

You should provide a 1-2 page pdf that includes:
An explanation of the URL structure used to access resources.
The URL structure used was just as the lecture demonstration used. I have /patient and /insurance as the major routes, with their functions as add-ons for the URLs. They can all be found in the main.py file. If it was related to the patient, then the path (/[*pathname*]) reflects the function in patient.[*method*] which can be found in the patient.py or insurance.py files.

Mackenzie Holznecht
Assignment 3, part 2

A description of which RESTful constraints you met and which you did not (you do not need to make a RESTful app, but you do need to know why it does not meet all the constraints of a RESTful API)

Uniform Interface – I believe my API complies with this aspect of a RESTful API because I made use of json and its strings to pass the data. This makes it so the client doesn't have direct access to the database information, but rather requires the use of json.

Stateless – to my knowledge this API complies with the stateless aspect of a RESTful API because there aren't global session variables, rather each request and its information is contained in itself so the application doesn't have to track across requests.

Cacheable – I'll be honest, I have no idea if my API does or doesn't comply with this aspect of a RESTful API.

Client-Server – I believe my API does comply with this aspect of a RESTful API because it separates the client from the server. This is evident in the fact that the API right now doesn't have a UI at all – the actual website only produces a json string of data. Only command line operations can be done on the server side with cURL tests to see if the API is functional.

Layered System – I think my API complies with this aspect of a RESTful API because there isn't any way for the client to see the server with which they're actually interacting. For all we know, google could be hosting my API from a few different servers (probably not because it's so small) but the reasoning still stands.

Code on Demand – I don't think my API does this optional feature of a RESTful API, but with the use of applets it may be possible. I was too engrossed in figuring it all out to look much into applets and how they work.

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

The major change I had to make from last week's design was to simplify the properties of each entity. Originally I had planned on the patient and insurance kinds being verbose with many possible properties (addresses, phone numbers, insurance ID numbers [different than an ID in the datastore table], copays, etc.). With this simplification I was able to scale everything down and make methods and EVERYTHING more simple, making it easier for me to figure out. I could have (and plan to) made the classes extremely detailed, but I believe I would have spent much unnecessary time doing this, when in reality I was just trying to figure out how to get it deployed successfully with the necessary components. I'm still not even sure I have all the functionality, as my cURL tests are EXTREMELY limited and only include one POST and one GET example (and one failed PUT example). I have the methods for the PUT and DELETE in place, but am simply running out of time to figure out the rest.

Having finished the API, what you would have done differently.

I would have liked to have had a better understanding of the general themes and set up prior to making this API. I think I would have had a lot more time to play around with authentication and authorization, as well as had a better experience making this API instead of worrying I wouldn't be able to figure out any of it. I went all the way back to week 1 material and tried to start from the beginning because I just wasn't understanding. I think I have a little better grasp on it, but I'm still not confident on getting my environment and project set up. The actual coded material from lectures and material, as well as references from the website and videos, is what I actually understand. It's the additions of templating, libraries, and various project components which throw me off. Hopefully with my current understanding I'll be able to keep building and have more success in this course.

Mackenzie Holznecht
Assignment 3, part 2

You should include a .zip file with all of your source code. This file should also include the previously mentioned pdf.
This file is included in the .zip, so hopefully everything is here ☺