**Assignment 6**

-REST Server (150 points)-

**Important: Students are supposed to do all the exercises and practices in the Module 8 and 9, prior to try this Assignment.**

**Assignment Submission Rules:**

1. **Detection of plagiarism will result in receiving the failing grade.**
2. After completion, students must submit the following three types of deliverables through D2L assignment box (previously Dropbox) by the deadline:
   1. **Your CODE (“.py”)** must be submitted in files separate from the report, so that I can compile and run it in my environment. It is better submitting a compressed project folder. Each function, method, class, and variable should be identified and their functions explained in comments.
   2. **Your CODE in document type:** All of your implementation must be converted into any of three format: “.txt”, “.doc or .docx”, or “.pdf”. You can simply copy and paste your CODE on “notepad” or “MS word”. The converted files must not be compressed and must be submitted separately and individually.
   3. **A REPORT FILE (MS word or pdf):** This text file should include i) A description of your solution; and ii) the output displayed when running your code. Your solutions description should include a synopsis of how your code is intended to work and the tests that you created to prove that it works as intended. The output can be a screen-shot from your computer.
   4. **Submission Example (only for Assignment 6):**
      1. Assignment folder: **Homework 6-Codes (zipped file)**
         1. Codes.zip (includes REST\_server.py)
      2. Assignment folder: **Homework 6-Codes in Document and Report**
         1. REST\_server.txt
         2. Reports.docx
   5. **Failure to follow the rules will result in deducting points.**
3. It is much better to submit a partial/failed-attempt solution than none. Include the circumstances of the incompletion in your report.

**Problems (Implementing a REST Server- 40 points, 10 points per Unittest):**

Develop a REST server script named measurement\_server.py. This script will respond to requests for data from the measurement database by sending back data that is JSON serialized. The server should listen on any interface at a particular port number.

Here are the paths and the requests that they correspond to

| **Path** | **Function** |
| --- | --- |
| /area | Get a list of all areas |
| /area/(\d+)/location | Get all locations for the given area id |
| /location/(\d+)/measurement | Get all the measurements for the given location id |
| /area/(\d+)/category | Get all the categories to which the given area belongs |
| /area/(\d+)/average\_measurement | Get the average measurement for the given area |
| /area/(\d+)/number\_locations | Get the number of locations in the given area |

Notes:

* In the first four request types, return a list of dictionaries, JSON encoded
* In the last two request types, return the numbers. JSON encoded

You may wish to follow the organization used in the sakila\_rest example in class.

**Overview**

This assignment will use the measurements example database as a subject. You will be providing data from this database as a service using REST.

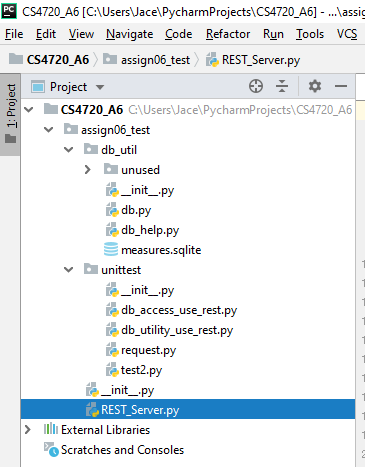
The measurements example is described in the Measurements Example.pdf. **But, please note!** Use the file [measures.sqlite](http://ksuweb.kennesaw.edu/~bsetzer/4720sp16/extra/examples/measures.sqlite) rather than recreate the file yourself.

**Test your implementation before submitting**

The project “assign06\_test.zip” contains a short test. The file test2.py is the test script, the other files are support. Note: the testing files should not be in the same directory with the server. Port number is set to “12345”. If you want to use a different port number, please modify it in request.py.

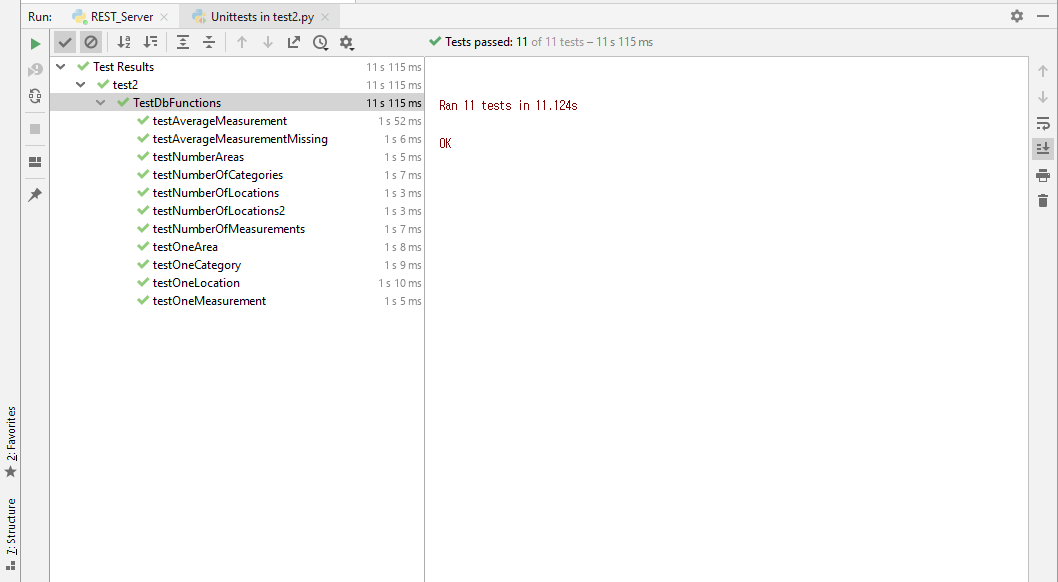
**Test Setup**

1. Unzip “assign06\_test.zip” and copy the folder “assign06\_test” to your project.
2. If your implementation file is located under the folder “assign06\_test”, you can use “db\_util”, “measures.sqlite”, and relevant APIs.
3. The following figure depicts the structure of my project. “REST\_Server.py” is my implementation:

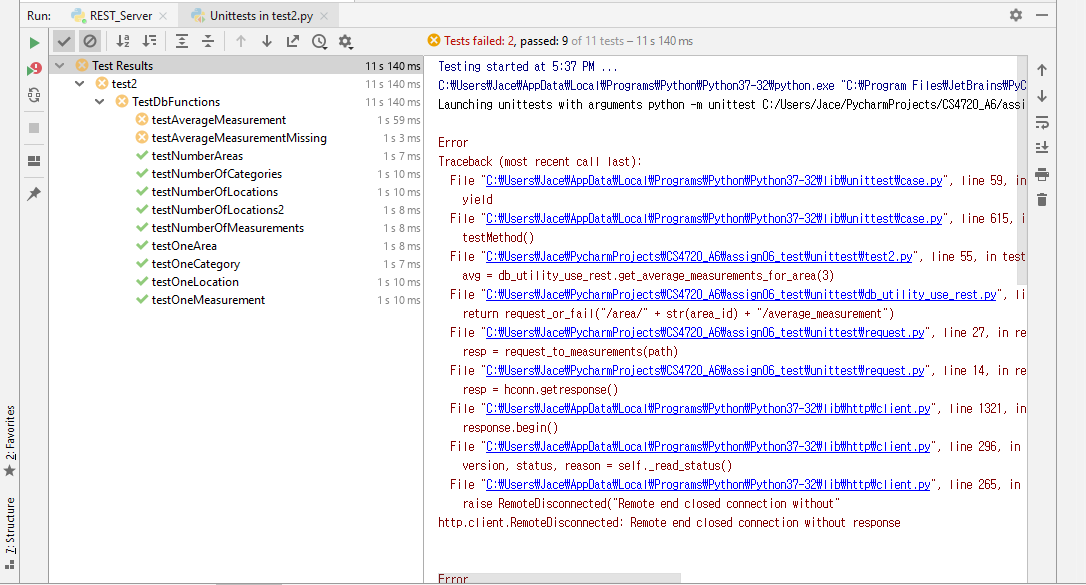


**Test Your Codes**

1. Execute your server code first. For instance, executing “REST\_Server.py” in my case.
2. Execute “test2.py” under “unittest” directory.
3. The “Unittests” has 11 tests, and thus you will get the following message if your implementation is correct:
   1. **Note that the aim of this assignment is to obtain “OK” with the green check symbol in front of “Test Results”**



1. If not, you will get yellow “X” mark in front of a test like the following figure:



1. It means your codes for “average\_measurement” and “average\_measurement\_missing” have errors. You must check the relevant methods again.