CSCE 156 – Assignment 5

Fall 2013

Introduction

You will modify the application you developed in Phases I & II to interact with the database you designed in Phase III. Your application will be modified to load and persist data from the database rather than from local flat data files. Specifically, you will implement an Application Programming Interface (API) to interact with your database using the Java Database Connectivity API (JDBC). Your API will provide methods to load and persist data to your database.

Phase IV – Database Connectivity

For grading purposes, the database you designed in the previous assignment will need to be setup on your MySQL account on the cse server. Your code should be configured to connect and interact with this database. You should add/remove/modify data to your database as needed for your own testing purposes, but when we grade your program, we will clear out all the data using the API described below (that you must implement) and add/test using our own data.

Coding

You will modify the Java classes you developed in the prior phase to make the following updates:

- As before, place all of your classes in the `unl.cse` package
- `PortfolioReport` – The driver class will retain its functionality; however instead of reading from data files, it will make a connection to your database, load the appropriate data and create the appropriate objects. The main method in this class will output the same summary report and portfolio detail reports as the previous phases. It is highly recommended that you implement (and reuse) several "factory" methods that retrieve instances of your defined classes by loading from your database.
- You have been provided with a new class called `PortfolioData` in which we have defined a collection of methods to interact with your database. You will fully implement each one of these methods. To better follow encapsulation, you may choose to actually implement some of these methods in the actual entity classes that correspond to the data types. You could then call those methods from the methods in the `PortfolioData` class. You may *add* any additional methods that you feel will simplify your task, but you may *not* modify or remove any of the methods already defined.

The methods you implement will be called by our grading program to load data from our test cases into your database. Your JAR file is then called to produce the reports and verify the correct output. In this context, our grading program can be viewed as the "client" program that interacts with your API. That is, your API is a generalized service that could also be used in a webapp, a GUI application, a mobile app, etc.

Artifacts

You will turn in all of your code as a runnable JAR file using the CSE webhandin (`PortfolioReport.jar`) with your source code included in a zip file (`PortfolioReport.zip`). Any library files (such as the mysql connector JAR) must also be included in your JAR file. If you follow the instructions as before, this should be automatic. JAR files can be included in your project like any other external library (instructions are available in Assignment 2, Appendix B or in Lab 1). Turn in a hardcopy of the grading rubric with your name(s) and login(s) filled out.

Design Write-up

You will update and modify your Design Document draft to include details on your new database API. You must hand in an updated printed hardcopy will be handed in 1 week prior to the assignment due date.

In the new section, make sure to given enough details so that a technically competent individual would be able to reasonably reproduce an implementation of your design. Be sure to address *at least* the following.

- How are records loaded from the database and into Java objects?
- How does your API persist data to your database?
- What are the various *side-effects* of each method? That is, what other records are consequently removed from the database in order to maintain data integrity?
- Document any additional helper methods that you designed and implemented
- What, if any, data validation do you perform? Where does the validation occur? What expectations do you place on the user with respect to the API?
- The API allows a user to submit null data—what consequences does this have on how have you handled it?
- Detail your testing strategies and any changes that were made as a result