COSC2101 Software Engineering Process & Tools

Assignment 3 (iteration 3)

Deadline: 11.59 pm Wed of week 12 (May 8, 2013)

1 Overview

In this assignment, you are to develop **QuickManage** - a Java desktop program that manage the daily operation of a local music and art school in HCMC. This document describes a subset of the requirements of the complete program. Additional requirements will be added in later assignments.

Note that the requirements given here are user requirements (high-level) that are intentionally vague, incomplete and may contain conflicts. The system requirements are not yet available. Part of your responsibilities is to figure out the system requirements by analyzing the user requirements carefully, perform relevant research, and asking the client representative (acted by your lecturer) appropriate questions to discover and specify the system requirements.

Despite consisting of only partial requirements in this assignment, you are expected to apply Rational Unified Process (RUP) and other relevant practices to complete a proper release by the end of the iteration. This means that at the end of this iteration, your program must:

1. Be able to run and met the given set of requirements.
2. Be easy to use (i.e. have an intuitive user interface).
3. Have an appropriate level of acceptance testing. This means you should have checked if your system satisfies the given requirements. Any defects discovered should be reported as a ticket in unfuddle.com and get fixed.
4. Have all of the required documents.

2 User Requirements

This is the last iteration of the project. The objective is to complete the whole program for the final delivery to the client. As a result, you must satisfy the requirements from previous iterations as well as the following new requirements.

- Dashboard: this is the first screen showed up when user log in the program. It shows the current number of classes, students, teachers, capacity utilization ratio, total paid tuition fee of last month, total unpaid tuition fee of last month, total paid tuition fee of the current month, total unpaid tuition fee of the current month.
- Allow managers to add/edit/delete class types (e.g. piano, violin, guitar single, guitar dual, etc.). Each class type has a name and a lesson fee. Please refer to http://sonatacenter.files.wordpress.com/2013/04/sonata-fee-schedule.pdf for the complete list of all class types at the moment.
- Update your program to allow user to generate more than one invoice per day for a

student. When generating invoice, the tuition fee for a student enrolled in a class is automatically calculated by the program base on the lesson fee and the number of lessons

- Allow managers to set an hourly pay rate for each skill (piano, guitar, etc.) of a teacher. You should combine this feature into the current add/edit teacher use cases.
- Pay slip: user (both staff and manager) can generate pay slips for one or more teachers. The monthly pay slip for each teacher is calculated base on the classes that he/she teach in a particular month. The monthly pay slip contains a list of classes. For each class, it must show the number of lessons, the total number of teaching hours, the hourly pay rate, the amount of money earned for teaching the class. Finally, a pay slip must have the total salary of the month. Once generated, a pay slip will be saved automatically and is associated with a teacher. User can view the list of all pay slips (newest pay slip is listed first) and each pay slip in detail from the GUI of a teacher record. In addition, managers have the right to delete pay slips.
- Export and import data to and from CSV (Comma Separated Values) files: allow managers to export all the data in the program to CSV files. It's up to you to determine the number of CSV files and their structures. Likewise, allow managers to import data from CSV files. When using together, this feature can serve as a backup solution or to initialize data in batch mode for the program.
- Create shortcuts (hot keys) for all functions in the menu bar and tool bar, e.g. Ctrl + X for "Exit".
- In addition to the current English interface, create a Vietnamese interface. Users are able to switch between the interfaces on-the-fly when the program is running. Note that this function only switches the interface, not the data.
- Create a Windows installer (please research to find a free tool to build the installer) to install the program in a specified folder (the default folder is C:\Program Files\QuickManage). The data folder (containing the binary data files) and other files and folders will be installed in this specified folder.
- Improve and write more unit tests as needed for all classes in the Model package and its sub-packages, as well as the classes that read and write to files.

3 Submission

3.1 Deadline & Submission

The deadline of this assignment is **11.59 pm Wed of week 12 (May 8, 2013)**. By the deadline, the Team Leader must submit to Blackboard the following items in a zip file:

1. **README**: contains acknowledgments, references, descriptions and locations of the required documents and source codes in the submitted folder. In addition, it must contain the list of team members, their roles, responsibilities and percentage of effort (e.g. Minh Nguyen 25%) in the assignment as well as other relevant information. The sum of all efforts must be 100%. Because each member has a different level of knowledge and skills, effort in this context should be roughly understood as the number of working hours rather than the amount of work accomplished by each member. What we want in this course is that each member should put in a similar number of working hours into the assignment rather than letting a few members doing most of the work. The individual mark received by each member may be adjusted from the team mark base on his/her effort percentage.
2. **Software Requirements Specification (SRS)**: a PDF file containing the Use Case

Diagram, all Use Case Specifications, and all non-functional requirements.

3. **Software Architecture Document (SAD)**: a PDF file containing the package and class diagrams of your design model and one page explanation of your design. Here you should explain issues such as your choice of one data structure over another, the reason for selecting a particular type of relationship between two classes, the reason for using an inheritance hierarchy, an interface, an abstract class, etc. In addition, explain how well your design follows the MVC patterns and the two design principles: low coupling (between classes) and high cohesion (between methods in a class)
4. **Source code**: must be loaded, compiled and run in NetBeans 7.2. The source codes submitted must not contain any metadata from SVN.
5. **Windows installer: when double-clicking on this file, the program will be installed and a shortcut of the program will be added onto the desktop. User can run the program by double-clicking on the shortcut icons or clicking at Start -> All Programs -> QuickManage**
6. **User Guide**: must be written in HTML but it should be formatted to allow pretty printout without breaking the pages. The User Guide contains acknowledgements and detailed instructions on how to install and use the program. Include relevant screenshots for the usage instructions to help the end-user following your instruction easily.

For your convenience, please use the simplified templates of SRS and SAD provided for you in Blackboard. More information on RUP and its templates can be found at:

- http://www.ts.mah.se/RUP/RationalUnifiedProcess/
- http://www.ts.mah.se/RUP/RationalUnifiedProcess/wordtmpl/index.htm

For User Guide, please refer to a sample user guide from Google Earth:

- http://earth.google.com/support/bin/static.py?page=guide_toc.cs

You should use StarUML or other suitable UML tools to create the original UML diagrams for your documents. However, the diagrams must be eventually exported and embedded into the documents in JPEG, GIF or PNG to ensure that your work can be assessed. Make sure that your diagrams are clear and large enough for the marker to read.

Put your submitted files into a folder and compress it into a single **.zip** file before uploading it to Blackboard. **DO NOT** use any other compression formats. Use of other formats (e.g. RAR, 7zip, etc.) may lead to delays in marking and/or a deduction of assignment marks. There should be only one submission per team.

3.2 Marking Guide

Criteria	**Marks**
SRS (SMART requirements)	5
SAD (apply MVC correctly, good package and class diagrams for the design model, sound design justifications)	5
User Guide (complete, good layout and structure, clear and easy to follow instructions, etc.)	5
GUI (intuitive and easy to use)	10
Features (all requested requirements meet client's expectation)	60
Unit test (high quality unit tests for the required classes)	15
Total	100

3.3 Plagiarism Notice

This is a teamwork assignment. While discussion and assistance between teams is encouraged, your attention is draw to the RMIT policies that outline the requirements and penalties for submission of work that is not your own. Any assistance received with the assignment must be acknowledged in the README file and in the code if relevant. Discussion about program features, design, implementation and quality issues is encouraged but exchange of files and/or copying of designs are regarded as plagiarism and will be penalized to the maximum extent.

Students should be aware that excessive collaboration can be a problem and can lead to disciplinary action being taken. Excessive collaboration can occur when students work on separate pieces of the assignment and share the resultant design/code to each to submit as a whole assignment. Consult your lecturer for advices on this issue when you are in doubt.