Muhammad Aziz

CS102

Professor Bari

24 February 2020

Course Registration System Overall Design

Apart from the abundant relationships between abstract classes, classes, and interfaces shown in the UML diagram. The design consists of multiple object-oriented related concepts within the program. For example, the crux of the Admin and Student classes are based on the abstract User class which contains the common class attributes of username, password, first name, and last name which all of the students and administrators would have. I also added an implementation of systemExit() which is a method that all users would need. These links between the classes would basically be classified as Inheritance which is a mechanism in which one object acquires all the properties and behaviors of a parent object. Furthermore, polymorphism which can be seen between, for example, the User and Student class, is where an object created can be used as an instance of different classes. Although I didn’t use any overloaded constructors, method overriding took place when I had to implement the createCourse() and deleteCourse() methods. Lastly, the principle of encapsulation is used throughout the design with the use of private attributes (such as making the Course array list provided) and then providing getter and setter methods (like the .getCourses() method ). This hides the complexities of the code from the user. Lastly, an abstract data type is a type with associated operations, but whose representation is hidden. This can be seen when I used “Course” as a datatype for the Data.courses array list.

Apart from the self-explanatory workings of the Admin and Student classes and their exclusive methods, the design of the CRSystem class is where the main method and the backbone of the program occurs. One initiated a try/catch block checks if deserialization is possible. If there wasn’t a previously made byte file, an IO exception will be thrown activating the code to import courses from the CSV file. After this, a while loop will ask for user credentials. Based on these, either an admin main menu or student main menu will be activated. The student can then make whatever changes, and when they chose to exit the while loop within the student main menu, the exit method will be activated which will trigger serialization, exit the method, then break the while loop within the main method (ending the program). The administrator works in a similar way, however, the admin menu has two additional methods linked in for options for the course management and reports. These also work with while loops that break out and return to the basic admin menu method. Apart from these backbones, two niche and important classes I would want to mention are the Data classes containing all management of the student registry and course registry array list. This also contains the serialize and deserialize methods that are used with the larger context of the program.