COMP1161 – PROJECT PARTS 2 & 3
SEMESTER 1, 2019/20
(Worth 20 coursework marks. Can be attempted individually, or in pairs

In Part 1 of the project, you wrote classes to store data for a simple system designed for lending and borrowing books and equipment. The overall structure of the application is depicted in the class diagram in Figure 1. Details of the classes that were developed in Part 1 of the project are suppressed here[1].

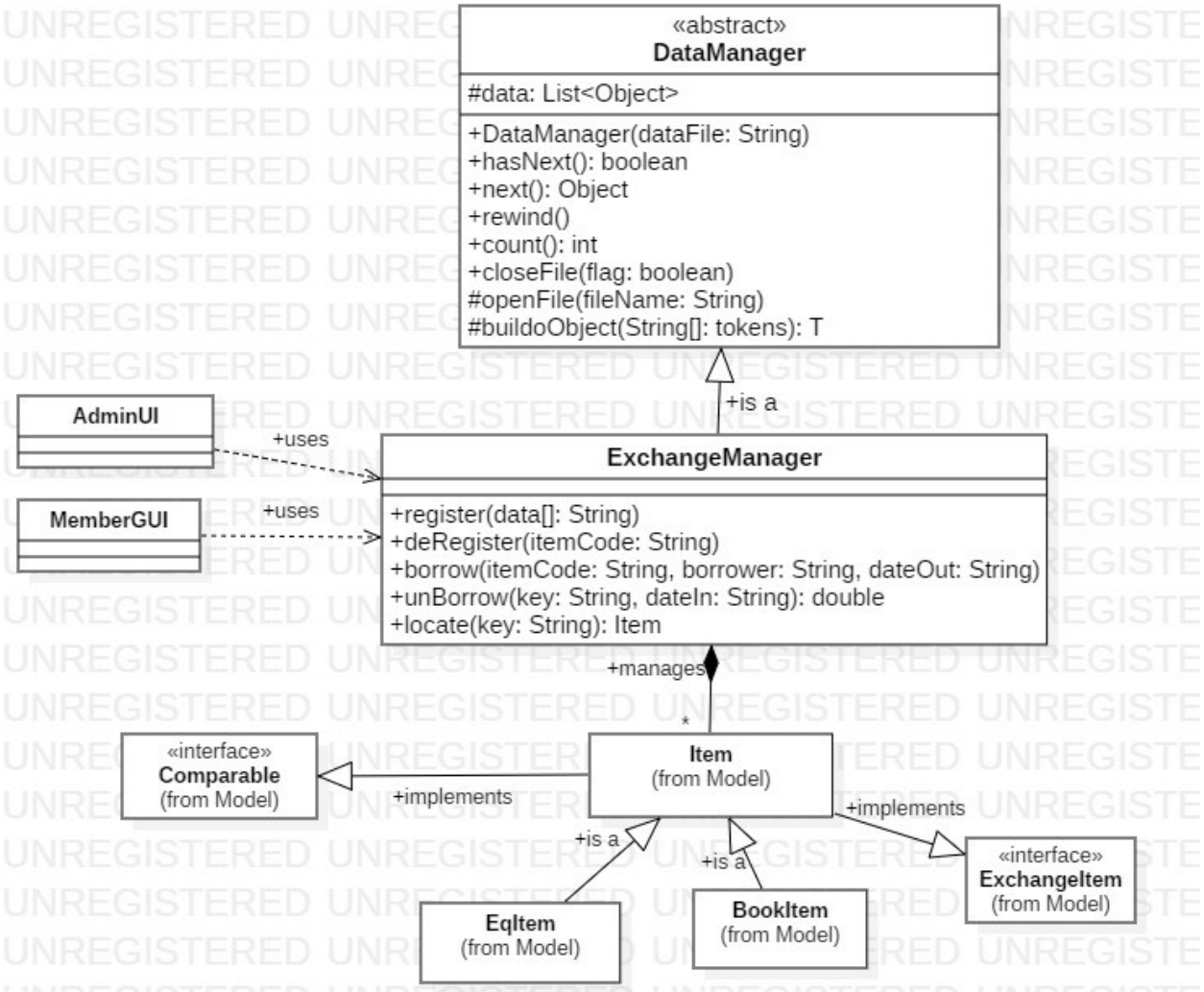

Figure 1: The Lending Exchange Application Class Diagram

[1] Abstract methods are usually written in italics but are not shown here due to tool limitations. Please see the sample code given for the abstract class DataManager to identify the abstract methods.

Part 2 of the project requires that you build an application that manages the collection of items on the Lending Exchange. You will also build a text-based user interface to manage the collection of items that are registered with the exchange.. Part 3 will require that you build a simple graphical user interface (GUI) that can be used to update the system's data each time an item is borrowed, or returned.

You are provided with code for the abstract class DataManager. This class represents the concepts involved in managing a collection of data that has been read from a text file. The data from the file is used to populate a collection of objects. Some of the methods in this class are implemented while other methods will be implemented when you extend the abstract class to create an application.

Please spend some time getting familiar with the code in the abstract class. Most of all, please make sure that you understand how generics are used to make the data manager reusable with any type of object. A simple program (with stubs in many places) is provided as an example.

Design Notes

Software applications that are used to manage data are typically referred to as CRUD applications since these applications typically allow a user to Create, retrieve, Update, and Delete data.

Please pay attention to the "uses" relationship that is depicted in the class diagram. A uses relationship is similar to aggregation except that one object does not keep the other as an attribute. Instead, one object is passed to the other as a parameter.

The design also demonstrates an important OO design principle, **Separation of Concerns**. The ExchangeManager class does not undertake user interface tasks. Instead, these tasks are the responsibility of the text user interface. This design approach allows us to use the application with different user interface objects as we will see in part 3 of the project.

Another important consideration is what is referred to as the architecture of the application. The classes that constitute the user interface communicate only with the classes that manage the application's functions. The class which implements the application's functions communicates, in turn, with the objects that store data. This the application can be considered to be organized by layers, This is commonly referred to as a layered architecture, and is very popular for information processing systems.

Task 2A: Build the Application (8 coursework marks)

1. Write the class ExchangeManager, which extends the DataManager class to provide functionality for the Lending Exchange. This class shall implement the abstract methods that were specified in the parent class. These are:
   a. The method openFile which opens a text file and returns a Scanner object that can be used to read lines of text from the file.
   b. The method buildObject that accepts an array of strings and uses the values in the array to create an Item object which is then added to the collection.
   c. The method closeFile that accepts a boolean flag as its only parameter. If the parameter has the value true then this method shall open the data file for writing (thereby deleting its contents), and shall write the contents of the collection to the file. You might want to use the PrintWriter class to do this.

2. In addition to implementing the abstract methods that were inherited from the parent class, the ExchangeManager shall provide the following methods
   a. A method named register that is used to register an item on the Exchange. This method shall accept an array of strings as its arguments where each array element is a data field for an Exchange Item (book or equipment). This methhod and shall create the appropriate object and add it to the collection
   b. A method named deRegister that is used to remove an item from the Exchange. This method shall accept the item code as its parameter and shall locate the item and remove it from the collection.
   c. A method named borrow that is used to update an item when the item is borrowed. This method shall accept, the item code, name of the borrower, and the date of borrowing as its parameters.
   d. A method named unBorrow that is used to update an item when the item is returned by a borrower. This method shall accept the item code and the date the item is returned and shall compute (and return) the calculated charge for borrowing the item.
   e. A method named locate that accepts an item code as its parameter, searches the collection for the item, and returns the item (if located), or null otherwise.

Task 2B: Build a Text-based User Interface (4 coursework marks)

Write the class AdminUI that implements a simple text-based user interface to the application as would be used to update the Exchange with items. This class shall:

1. Require the user to enter a user name and password which are authenticated to ensure that the user an administrative user of the system.
2. Display a menu with the options (Register an Item, Deregister an Item, List Items, and Quit)
3. Accept the user's selection and take the necessary action

a. If the user selects Register, then the program shall prompt the user to enter details of the item and add the item to the collection.
b. If the user selects De-Register then the program shall prompt the user to enter the item code for the item that is to be deleted and shall remove the item from the collection..
c. If the user selects List Items then the program shall display all items in the collection.
d. If the user chooses to quit then the program shall prompt the user on whether or not to save changes and save the data only if the user responds affirmatively.

PART 3: A GRAPHICAL USER INTERFACE (6 coursework marks)

Part 3 of the project requires that you build a simple Graphical User Interface (GUI) for managing borrowing and lending of items on the Exchange.

1. On startup the GUI shall prompt the user to enter his/her user name and password.
2. On successful login the GUI shall display a form that displays all items that are owned by the user. The display shall include details of each item and whether or not the item has been borrowed. If the item has been borrowed then the name of the borrower and the dat borrowed shall be displayed.
3. The GUI shall allow the user to update the system when an item is borrowed by inputting the name of the borrower, and the date borrowed.
4. The GUI shall allow the user to update the system when an item is returned by inputting the date the item is returned. The number of credits to be charged should be calculated and displayed.
5. You are free to layout your GUI as you wish subject to the following constraints: Your GUI should use at least one component from each of the following categories:
    a. Text box or Text Area
    b. Button or Menu
    c. Radio Button or Check Box
    d. List Box or Combo Box

GENERAL CONSIDERATIONS (2 Coursework marks)

1. Marks will be awarded for good programming practices:
    a. Good choice of identifiers
    b. Documentation of code
    c. Use of packages
    d. Consistent indentation of code