**Faculty of Science and Engineering**

Referred Coursework – 2018/19 Academic Year

**PLEASE NOTE: If you have been referred in the COURSEWORK element of this module and are required to be reassessed by COURSEWORK, please complete this referred work.**

Module Code: SOFT252

Module Title: Object Oriented Software Engineering with Design Patterns

Module Leader: Dr Alma Rahat (alma.rahat@plymouth.ac.uk)

School: Computing, Electronics and Mathematics

**DEADLINE FOR SUBMISSION: 3pm 15th August 2019**

|  |
| --- |
| **SUBMISSION INSTRUCTIONS FOR CANDIDATES** |
| Referred coursework must be submitted electronically using the online submission facility in the DLE by the published deadline.  If you have any queries on submission or in relation to the referred work, please contact the Module Leader in the first instance, if they are unavailable please contact the School of Computing, Electronics and Mathematics on 01752 586200 immediately so any problems can be rectified. |

**If you require any part of this publication in larger print, or an alternative format, please contact:**

**School of Computing, Electronics and Mathematics**

**T: +44 (0) 1752 586200**

**E:** [**compmath@plymouth.ac.uk**](mailto:compmath@plymouth.ac.uk)

# Coursework Assignment 2018/2019

## Assignment Brief

Assignment Title: Object Oriented Software Engineering with Design Patterns

Submission Deadlines: **3pm 15th August 2019**

Submitted at: online through the DLE site

Contribution to final grade: SOFT252 –70%

Individual / Group assignment Individual

Module: SOFT 252 - Object Oriented Software Engineering with Design Patterns

Module Leader: Dr Alma Rahat

Scenario: Please see**Appendix 1: Scenario.**

## Requirements

You are required to produce a **Java standalone software application** that meets the requirement specification in **Appendix 1** of this assignment. This application should follow the MVC pattern (as defined in lecture). In addition, your data model design must use at least two other software design patterns covered in the course.

You are required to use test driven development (TDD) to create your solution to the scenario. Your solution should be implemented in the Java programming language using the NetBeans IDE. The implementation should contain two NetBeans projects as follows:

1. A Java Class Library containing the implementation of your data model as per your UML Class diagram. This project should also include the set of JUnit tests used to create your data model. The data model should have a full set of documentation **generated** for it using the **Javadoc tool**. It is not sufficient to simply mark-up your classes you MUST GENERATE the Javadoc web site. A 5% marking penalty will be applied if this is not done.
2. A Java NetBeans project which provides a graphical user interface (GUI). This project **MUST** make use of the classes in your Java Class Library (see 1 above). The GUI should provide the functionality listed in **Appendix 1: Scenario.**

You **MUST** use *git* version control with *bitbucket.org* in your development process. You **MUST** add the module leader (user ***arahat*** in *bitbucket* or ***AlmaRahat*** in *github*) to your repository. You will be assessed on appropriate use of version control.

In addition to the software you are also required to write a short reflection on the design and implementation of your solution. There is no minimum word count for this section. You are expected to write no more than 1000 words. The reflection should cover the following:

* The design process leading up to the finally submitted system
* A description of the design choices made, and the motivations behind them
* A discussion of the degree to which the design meets good design criteria
* A discussion of any shortcomings that may exist in the submitted work, and how these might be addressed

## Submission Details

You are required to make two online submissions for this assignment. In each case you should place all the files being submitted into a single folder and create one .zip file for that submission. Your zip file should contain your **NetBeans class library**, **NetBeans project**, **Javadoc web site** and the reflective report. All the contents of the zip file **MUST** also appear in your *git* repository. Please submit via the link provided on the SOFT252 DLE. Your zip file submission should be named:

**SOFT252\_Coursework.zip**

You **MUST** add your *git* repository address in your report.

### Plagiarism:

This is an **individual** assignment and must reflect the work of that individual.

Thus, while you may discuss, **in general terms**, this assignment with your colleagues, the assignment **MUST** be your own work.

**Do not:** Share designs or code with anyone, OR submit a program design or code which is wholly or partially the work of someone else.

**The University treats plagiarism very seriously. In all cases of suspected plagiarism, formal action will be taken.**

The penalty for submitting work which is wholly or partially the work of someone else is usually, at least, a mark of zero for the assignment. Also do not be tempted to help a colleague out who is ‘stuck’, by giving them your code or design, as BOTH parties will be guilty of an assessment offence and BOTH face the risk of a zero mark. Please refer to your student handbook for guidance as to what constitutes original / individual work.

**The module leader may viva students on the contents of any part of their submission of their assignment.**

Ok so how much can you share with your colleagues? The following table gives an indication as to what is allowed / not allowed:

|  |
| --- |
| **Allowed** |
| Discussing in general terms, i.e. if started / finished assignment; how easy/difficult; clarifying requirements (but should email me to confirm). |
| Helping to fix compiler errors, but NOT suggesting a better way to do the coding. |
| **NOT Allowed** |
| Suggesting better way to do things, i.e. giving or suggesting an algorithm[[1]](#footnote-1) |
| Meeting as a group to discuss design and / or algorithms and then, later, individually coding in the design / algorithm |
| Give ANY: code, design or algorithms to a colleague. |
| Using someone else’s designs/ code found on some media (i.e. a hard-drive), or some printed document found somewhere |
| Working collaboratively with a colleague or colleagues to complete the assignment |
| Getting some else to complete the assignment for you! |

**Essentially if the level of collaboration is not covered by the allowed section, above, you MUST assume that it is NOT allowed. If you have any doubt then email the module leader BEFORE you do it!**

### Deadline

The **deadline for submission** is **XXXX** via the submission link on the modules Dynamic Learning Environment (DLE) home page.

### Backup

It is YOUR RESPONSIBILITY to ensure that the zip file submitted will open correctly and will run on a standard Plymouth University computer, using NetBeans 8. All files must be free from viruses.

You must also retain copies of all coursework until after the examination boards have met and final transcripts issued.

## Other Information

### Module Learning Outcomes Assessed

* ALO-1: Use requirements analysis artefacts to progress into software design.
* ALO-2: Identify and use suitable design patterns.
* ALO-3: Develop and evaluate an object-oriented program to solve a given problem.

### Assessment Criteria

In order to **PASS** this coursework you must demonstrate your ability:

* To design a software solution that meets user requirements
* To code some basic Java classes according to specific requirements.
* To run tests to ensure that the classes you have coded meet the requirements and produce correct results.
* To design a simple desktop application and link it to data model classes.
* To use version control system *git* appropriately.

To achieve a **second class honours mark** you need to demonstrate a greater ability in coding and testing of the more complex aspects of the classes and their associations. Examples would include:

* Using **inheritance** and **composition** appropriately in your design and implementation.
* Using **software interfaces** and **abstract classes** appropriately.
* Use proper Java **coding conventions** in your implementation
* Apply **software design patterns** in appropriate places in the design and implementation.

To achieve a **first class honours mark** you need to meet the pass & second class criteria and:

* Implement the data model to a high standard providing all requested functionality.
* Apply good human computer interface (HCI) to the creation of your GUI application.
* Make reasonable design choices showing an awareness of the real world consequences in the provided scenario.
* Go beyond the use of two design patterns.

### Contribution to overall module mark

In total this assignment is worth 70% of the overall module mark.

### Mark allocation within the assessment

|  |  |
| --- | --- |
| Segment | Percentage of marks­­­ |
| Design.  The design elements include:   * UML class diagram. * Development of JUnit tests. * Reflection of your design. | 50% |
| Software implementation.  The implementation elements include:   * Java Class Library containing your data model implementation. * Java GUI project with appropriate HCI that uses the class library. * The Javadoc web site produced from your class library. * Appropriate use of *git* version control. | 50% |

### Staff Responsible

This assignment has been prepared by the module leader, Dr Alma Rahat. In case of queries, please email Alma Rahat: [alma.rahat@plymouth.ac.uk](mailto:alma.rahat@plymouth.ac.uk)

# Appendix 1: Scenario.

Apex is a local library. They would like a brand new resource management and tracking system. They have provided the following scope of the project.

The system should have two types of users: admin and client. The user-specific functionalities are given below.

|  |  |
| --- | --- |
| Functionality | User |
| Log in to the system with a unique ID and password, and see user-specific functionalities. | Admin and Client |
| Create a new resource. | Admin |
| Check a list of all available resources. | Admin and Client |
| Check the current status of the resources and essentially find the answer for the following questions.   1. Is the resource on-site, i.e. has it been borrowed or not? 2. If it has been borrowed, when is it due to be returned? Is the item over-due? | Admin and Client |
| Borrow and return an item. Return may include paying a fine if appropriate. | Client |
| Send reminders to clients to return any borrowed resource. | Admin |
| Set automated reminders or warnings for overdue items. | Admin |
| Send newsletters to clients informing them of new resources. | Admin |
| Create purchase request for new resources. | Client |
| Receive and read purchase requests. | Admin |
| Request a resource to be returned. | Admin |
| Rate the quality of a resource on a scale of 5 with higher number representing better quality. | Client |
| Request extension to the loan period. | Client |
| Approve or decline requests for extension to loan period. | Admin |

The library has various types of resources available for clients: books, DVDs, newspapers, etc. A user must be able to inquire at least the following aspect of a resource.

1. Name: This is the name or title of the resource.
2. Type: This is the type of the resources, e.g. book, DVDs, newspaper, etc.
3. Category: The library uses the Dewey Decimal Classification to organise the resources by discipline or field of study. A list of classes is given in **Appendix 2: Dewey Decimal System,** and each resource should have one of these numerical codes associated with it.
4. User rating: The resource may be rated by the users. The rating system gives the borrower an idea of how it was perceived by users before: 5 star is assigned for a great piece of work and 1 star for a particularly bad resource. If a rating is available, this should be accessible to the inquirer of the record.

In addition, each resource is assigned with one of three possible loan categories.

1. Short-term: the item may be borrowed for 2 weeks.
2. Long-term: the item may borrowed for 6 months.
3. Reference: the item may not be taken off-site, but available for access within the library premises.

There is also a penalty system in place for late returns: 10 pence is charged per day. The client is stopped from borrowing new resources if an item is overdue by 20 days; the restriction is lifted when they return the overdue item.

They have specified the following reminder and warning frequencies.

1. Remind the clients a week before the borrowed item is due.
2. Warn client that an item is overdue.
3. Warn clients that the item is overdue by 20 days and they are no longer allowed to borrow new resources.

However, they also requested that the admin should have complete control of the reminder/warning system.

Your task is to design and implement the system described above. You **MUST** generate a list of 20 resource items (of your choice) for the library to aid the assessment purposes. You should also add one admin account and five user accounts. Please use a variety of resources from different categories in **Appendix 2: Dewey Decimal System.**

# Appendix 2: Dewey Decimal System.

Many libraries around the globe uses the Dewey Decimal system to classify different resources. Essentially, the idea is to assign a numerical code to a category of resources, and organise items with similar numerical codes in close proximity so that it is easier to locate an item from a large repository.

In this exercise, you only need to assign a resource to one of the general classes from below.

* 000 – Computer science, information & general works+
* 100 – Philosophy and psychology+
* 200 – Religion+
* 300 – Social sciences+
* 400 – Language +
* 500 – Pure Science +
* 600 – Technology +
* 700 – Arts & recreation +
* 800 – Literature +
* 900 – History & geography +

1. Note an Algorithm in the table refers also to giving or sharing pseudo-code. [↑](#footnote-ref-1)