Coursework Assignment 2016/17

Assignment Brief

Assignment Title:	Object Oriented Software Engineering with Design Patterns
Submission Deadlines:	**Monday 16 January 2017 at 1000**
Submitted at:	online through the DLE site
Contribution to final grade:	SOFT252 –70%
Individual / Group assignment	Groups of three
Module:	SOFT 252 - Object Oriented Software Engineering with Design Patterns
Module Leader:	Nigel Barlow (set by Mary Squire)
Scenario:	Please see ***Appendix 1***

Requirements

You are required to produce a Java software application that meets the requirement specification in *Appendix 1* of this specification. This application must follow the Model-View-Controller (MVC) pattern. In addition your data model design must use at least two other software design patterns from this list:

- Strategy Pattern
- State Pattern
- Observer Pattern
- Command Pattern
- Decorator Pattern
- Factory Pattern

You are required to use test driven development (TDD) to create your solution to the scenario. Your solution must include design documentation in the form of a UML Class diagram for your data model and a set of JUnit tests for this model. Your solution must 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).

You are required to submit the following to complete this assignment:

- The two Java NetBeans projects described above:
  - The Java Class Library containing your data model (this must include suitable JUnit tests for the classes in your library) and
  - The Java GUI project that uses your Java Class Library.
- A UML diagram showing the relationships between the classes in your Java Class Library.
- A Javadoc web site that documents all the classes in your Java Class Library.

A short 1000 word reflective report on the ***design*** and ***implementation*** of your solution. The reflection should cover the following:

- How your software implements good design principles or criteria.
- How your class design has evolved during implementation.
- How you managed the work among your group members.

Demonstration

In lieu of a live demo, you are to submit a 'paper-based' demo. You need to do this using a Word document or PowerPoint file showing screenshots of the progression through your applicati on, with suitable annotation to explain what is happening. You must include this in the zip file you submit.

Note: I will also run your application to check various aspects so please ensure that you provide everything needed for me to do this, including any data files, and a readme.txt so that I know what to run and how to load/save data.

The demo you submit will be the main way in which the operation of your application as a whole will be judged, so please ensure that you do it justice!

Submission Details

You are required to make one online submission for this assignment: please submit via the link provided on the SOFT252 DLE. You must place all the files being submitted into a **single folder** and create one .zip file for that submission. Your zip file must contain your **NetBeans class library**, **NetBeans GUI project**, **UML diagram**, **Javadoc web site, the reflective report, and the demo file**. Your zip file submission must be named:

SOFT252_Coursework.zip

Anonymous marking will be used. Please ensure your zip file is named correctly and do not add your name or student number or group id to the above – the DLE will track who submitted which files.

Plagiarism:

This is a **group** assignment and must reflect the work of the members of the submitting group.

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

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

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.

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]
Meeting as a larger 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 outside your designated group 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 of** is **Monday 16 January 2017 at 1000** via the submission link on the SOFT252 DLE home page.

[1] Note an Algorithm in the table refers also to giving or sharing pseudo-code.

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
- Demonstrate your ability to code some basic Java classes according to specific requirements.
- Demonstrate your ability to run tests to ensure that the classes you have coded meet the requirements and produce correct results.
- Demonstrate your ability to design a simple graphical user interface and link it to data model classes.

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 / 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.
- Make reasonable design choices showing an awareness of the real world consequences in the provided scenario.
- Go beyond the use of two design patterns.

Feedback and return of marked work

Coursework marking will be completed and returned to the students by 30 January 2017.

Contribution to overall referral mark

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

Allocation of marks within this assignment

The design elements are worth 50% of the assignment mark. These elements are:

1. UML Class Diagram.
2. Development of JUnit tests.
3. Reflection of your design.

The software implementation elements are worth 50% of the assignment mark. These elements are:

1. Java Class Library containing your data model implementation
2. Java GUI Project with appropriate HCI that uses the class library
3. The Javadoc web site produced from your class library

Allocation of marks within your group

Unless you inform me otherwise, each group member will be allocated the same marks. If the group feels that this allocation is not appropriate, you must inform the module leader as soon as the work has been submitted.

Appendix 1 - Scenario

Problem Specification

4.1 Purpose of the System

Unisystems provides cars to its staff for business use. Some cars are allocated to specific staff members on a long-term basis and some are kept in a pool for ad hoc use. The latter can be used for one day at a time, but cannot be reserved in advance. A desktop application is to be created to maintain details of the cars, and you have been tasked with conducting the analysis, design and implementation of a prototype of the application.

4.2 Users and functionality

Users of the application will be staff in the Transport Office who are responsible for maintaining all the data and for allocating the long-term cars. They will also take calls from staff members who need a car for the day and use the application to access the details of the cars available and allocate a suitable car.

Maintaining the data involves tasks such as adding new cars, deleting cars that have been sold, and updating cars, e.g. recording service dates (when a car is due for a service and when it was last serviced), changing the status of a car when it is no longer required and waiting to be sold, making a car unavailable if it is damaged and awaiting repair.

Cars booked from the pool are booked for whole days, so, for example, if a member of staff needed a car, he/she would contact the Transport Office who would access the system on the day it was required and select a car from those available. The car would then be allocated to that member of staff for that day only.

The transport office staff also need to be able to view cars that are due for servicing or under repair.

The application must provide suitable facilities for the above functionality, and also be able to store and retrieve the data to and from disc using serialization.

4.3 Getting clarification

You may find that you need to clarify certain aspects of the scenario as you work on the assignment.

The first port of call will be the FAQ which will be maintained on the DLE which can be found in the module assessment section. If the answer is not there, then please ask either in person or by email. If appropriate the question and answer will be added to the FAQ.