Department of Computing

University of Staffordshire

Module Name: **Clean Coding and Concurrent Programming**

Module Code: **COMP63038**

Title of Assignment: **Assignment 1**

Module Learning Outcomes for This Assignment

1. Demonstrate a critical understanding of the principles of clean code and clean testing, being able to refactor code into clean code.	Knowledge & Understanding Analysis
2. Design and implement a suite of clean tests for a given application.	Application Communication Problem solving

Submission deadline

As specified in the Consolidated Student Assessment Information Sheet (via LMS)

Feedback deadlines

You will be given oral verbal feedback during the demonstration.

Further written feedback will be made available via LMS within 20 working days afteryour demonstration

Introduction

This assignment, which contributes 50% to the module's marks, consists of

1. Implement the scenario briefly to adhere to the object-oriented principles and clean coding concepts.
2. Developing a set of clean tests for that application.
3. The assignment will be assessed by demonstration and report.
4. You should read the mark scheme carefully to understand how each assignment element's grades will be allocated.
5. You can choose Java or C# to implement the scenario given in this assignment.

Fundamental requirements

In this assignment, you must satisfy **all** the following fundamental requirements; otherwise, you could be awarded zero marks.

1. You must be the only author of the work you submit for assessment. You must not have help with this assignment from anyone except for the regular teaching

2. team members. You are reminded of the university's policy about academic misconduct, as described at:
http://www.staffs.ac.uk/support_depts/info_centre/handbook/academic-life.jsp

3. During the demonstration, if asked, you must be able to explain in detail the software you present for assessment.

4. You must submit your assignment via this module's LMS presence using the link provided in the Assessment section. Follow the instructions given with the link.

Assignment details

1. Download from LMS

Download the scenario brief from the LMS. This will not be available until the end of Week 2.

2. Write clean tests

Using the concepts of clean testing considered in this module, design a set of clean tests that you can use to verify the functionality.

Review your tests and Program class with your tutor to ensure you are on the right track for this assignment.

3. Design Architecture for the Solution

Design an architecture for the solution according to the given scenario, considering object-oriented methods and clean coding principles.

4. Implement the Solution and produce clean code

Implement the above solution to demonstrate as many clean coding concepts considered in this module as possible.

5. Write a report

Write a report (up to 3,000 words) in which you provide evidence for the various criteria in the marking rubric (see Appendix A). As a minimum, you should:

a) Critically analyze the scenario and develop a design.
b) Show a screenshot of the test results after completing step 2.
c) Describe the clean code concepts you introduced in step 4, explaining how thishas improved the application.
d) Show a screenshot of the test results after completing step 4.

6. Submit your work

Please clean your solution before submission to reduce the zip file size.

On or before the submission deadline date, use the link on LMS to upload a zip file containing:

- Your final application after completing step 4, including the clean tests.
- Your report
- Any other files you have created (e.g., the SQL commands for creating andpopulating your database)

Mark Scheme

You should refer frequently to the marking scheme during your work on this assignment.

How your work will be assessed

Your submission will be assessed using the rubric shown in Appendix A, which will give you quantitative and qualitative feedback on your work. You should ensure that your report presents evidence to support the awarding of marks in each rubric criterion.

After submitting your work, you will be required to demonstrate your software to a teaching team member for this module. You will run your software to prove that it works, and you will be asked questions about your code. You are expected to answer these questions correctly and confidently. Incorrect or unconfident answers might result in your marks being reduced. You are reminded of the university's policy about academic misconduct, as described at: http://www.staffs.ac.uk/support_depts/info_centre/handbook/academic-life.jsp

You will score zero marks for the assignment if you do not demonstrate your software.

If you wish, you can use your own laptop for the demonstration.

Instructions for the demonstration

You can find your demonstration appointment time on LMS using the link provided inthe Assessment section.

To prevent delays in the demonstration, it is essential that you arrive **at least** ten minutes early for your appointment and do the following before your demonstration start-time:

1. Log on to a PC (or your own laptop) in the demonstration room at a location indicated by your assessor;
2. Open solution, and prepare your application, which includes running the SQL commands to create and populate your database;
3. Be ready to show the tutor that the dates on your files are the same as those submitted via LMS;
4. Make sure that your database is in its initial state (i.e. ready to run without delay);
5. Await instructions from the tutor who will assess your demonstration.

Appendix A: Rubric

Assignment 1

Criterion (weighting)	Mark (%)				
	0-39	40-49	50-59	60-69	70-100
Critical analysis of the scenario of the application (10%)	Basic design diagrams. No analysis. The class diagram is required.	Very limited analysis	Most design problems are identified, and potential remedies are discussed.	Good exposition of the design problems with a good discussion of appropriate remedies	Very clear exposition of the design problems with insightful discussion that weighs the merits of appropriate remedies
Clean tests (35%)	No clean tests or The application does not compile	A few clean tests but little sophistication	A good number of clean tests that cover the essential aspects of each use case	Many clean tests that cover the essential aspects of each use case and most of the application's classes	Many clean tests that cover all the essential aspects of each use case and all the application's classes
Design patterns (20%)	No design patterns or The application does not compile	Two or three design patterns used (of those presented in the module)	Four or five design patterns used (of those presented in the module)	Six to Seven design patterns used (of those presented in the module)	Eight to Eleven design patterns were used(of those presented in the module)
Clean architecture (35%)	No clean architecture or The application does not compile	One or two SOLID principles used	All SOLID principles used	Components are clearlydefined in the code.	Most of all, the clean architecture concepts used in the code (of those presented in the module)