**School of Engineering, Computing and Mathematics**

**Oxford Brookes University**

**COMPT6030: Software Engineering**

**(Semester 1, 2024-2025)**

Coursework Specification

Software Engineering of a Modern Computer Application

# Assessed Learning Outcomes

This coursework counts as **100%** of the total assessment of this module. It is designed to assess your attainment of the following learning outcomes.

|  |  |
| --- | --- |
| 1 | Demonstrate an understanding of the software lifecycle and be able to critically analyse a problem to decide on a relevant process model. |
| 2 | Demonstrate an understanding of the role of risk in a project and be able to identify and manage both risk and the impact of change on a project. |
| 3 | Demonstrate an understanding of the role of requirements analysis and elicitation and the relationship between requirements and design and use this knowledge to be able to create functional and nonfunctional requirements for a project. |
| 4 | Be able to critically evaluate and utilise design paradigms of object-oriented analysis and design, component-based design, and service-oriented design. |
| 5 | Describe and employ software techniques, in the implementation of designs to achieve desired software quality including reliability, efficiency and robustness. |
| 6 | Understand and be able to apply a range of testing methods and techniques. |
| 7 | Use refactoring in the process of modifying a software component. |
| 8 | Understand how software reliability contributes to system reliability and be able to apply multiple methods to develop reliability estimates for a software system. |

# The problem to be solved

In this coursework, you are required to work **in a group of 4 students** as a software engineering team to develop[[1]](#footnote-1) a given modern computer application by applying appropriate software engineering methodology and technologies. Each member of the team will conduct software engineering activities for developing one subsystem and collaborate with the other members of the team to ensure that these subsystems are integrated and interact with each other to form a consistent system. A brief description of the specific application is given in the document COMP6030-CW-Case Study, which is available on the module’s Moodle website.

You are required to perform a series of software engineering tasks specified in Section 3 and to produce a set of software engineering documents in order to demonstrate:

1. you understand the principles of software engineering methodologies,
2. you are capable of selecting and applying appropriate software engineering techniques, and
3. you have the skills of using advanced software engineering tools.

# Tasks to do

The following specifies the tasks to be completed in this coursework and the distribution of marks in the assessment. A more detailed marking schemeis given in the file COMP6030-CW-Detailed Marking Scheme, which is also available on the module’s Moodle website.

## **Task 0: Organisation of the Team and Project Management**

The students are required to form a team by themselves with four members from the class. The team must set up a management scheme and a collaboration mechanism and work closely throughout the whole semester as a software engineering project.

Each team must set up a GitHub repository for the coursework project to host the documents of the project.

The team must meet regularly (for example, at least once a week) at a fixed days and times. Each meeting should have an agenda before the meeting and a minute to record the meeting contents, especially, the attendances to the meeting, progress since the previous meeting, a list of actions to take place and issues to be solved. The meeting agendas and minutes must be uploaded to GitHub immediately after it is created.

Each member of the coursework team must be responsible to the development of one the following subsystems:

1. *ClaClo-Student*: a subsystem for students to receive information and learning materials as well as to do exercises and conduct and/or submit assessments works.
2. *ClaClo-Teacher*: a subsystem for teachers to assign exercises and coursework to the students, to receive and mark student works (include exercises and assessments) and provide feedback to the students.
3. *ClaClo-Admin*: a subsystem for administrators of courses.
4. *ClaClo-Manager*: a subsystem for university managers to manage the curriculum of the university.

***Note:*** The members of the team should work on different subsystems, while the whole team should develop a coherent system where the subsystems interact with each other.

Each subsystem will inevitably have two parts: one part on the user’s computer or mobile device and another part running on servers on the cloud. The parts on the Cloud that belong to different subsystems should be properly integrated so that they interact with each other to enable functions to deliver to various types of users. Moreover, there may well be components running on the Cloud common to more than one subsystem and/or do not belong to any specific subsystem. The team should also work together to perform the software engineering activities in order to develop such common components.

## **Task 1: Specification and Modelling Software Functional Requirements (20 Marks)**

In this task, you will work as a requirements analyst in the project to produce a UML model of the software system to be developed using a software modelling tool. The UML model should contain the following types of models.

1. *Use Case Model* (10 Marks, Individual effort): Each member of the team should develop one Use Case Diagram to define the use cases of the subsystem to specify the scope of the software engineering project.
2. *Activity Model* (10 Marks, Individual effort): Each team member should select one use case of your subsystem to produce one Activity Diagram for the selected use case to specify the interactions between a user and the subsystem.

## **Task 2: Software Architectural Design (30 Marks)**

In this task, you will work as a software architect to produce an ***Architectural Design*** of the system in the *microservices architectural style*. The design should be at two different abstraction levels as follows.

1. *Architecture of the subsystem* (15 Marks, Individual effort): Each member of the team should produce an architectural design of your subsystem with focus on the microservices your subsystem provides and the microservices that your subsystem requests and other subsystems provides.
2. *Architecture of the whole system* (15 Marks, Team effort): The team should produce an architectural design of the whole system through integrating the subsystems together.

***Note****:*

1. *You should specify the architectural designs in the UML component diagrams with a set of component nodes to represent microservices and a set of interfaces to represent the connectors between them.*
2. *The components and connectors, including their methods and parameters, should be specified in a textual documentation to define their functionalities and meanings.*
3. *In this part of the design, you are not required to give the internal structure of the components and connectors.*

## **Task 3: Software Detailed Design (20 Marks)**

In this task, you will work as a software designer to produce a ***Detailed Design*** of the system for object-oriented implementation of the components (i.e. the *microservices*) in your *architectural design* of the subsystem. The design should specify the components from both structural and behavioural aspects in UML.

1. *Structural Model* (10 Marks, Individual effort): Each student should **select one component** in the architectural design of your subsystem and develop a Class Diagram to define the structure of the component.

***Note***:

1. *The selected component must be in the architectural model of your subsystem, but not in any other subsystems.*
2. *The UML class diagram must contain the classes and relationships between them. You must give the attributes and methods of the classes.*
3. *Behaviour Model* (10 Marks, Individual effort): Each student should produce a Sequence Diagram for the same component selected in Task 3(a) to define the dynamic behaviour of the component.

***Note****:*

1. *The sequence diagram should specify the internal process of the interactions between the objects inside the component. It must be consistent with the structural model that you produced in Task 3(a).*
2. *The behaviour model must cover all possible scenarios of the operations of the component using advanced modelling facilities such as frames.*

## **Task 4: Software Testing (30 Marks)**

In this task, you will work as a software quality engineer to develop a ***Software Test Plan***. Your test plan should consist of two parts:

1. *Unit test plan* (15 Marks, Individual effort): You should select one microservice in your subsystem to develop a unit test plan.

***Note***:

1. *The unit test plan should aim at adequate testing of the same component (i.e. one microservice) that you have developed the structural model in Task 3(b).*
2. *It should include a set of test cases that adequately cover all services and corresponding responses of the selected microservice.*
3. *System test plan* (15 Marks, Individual effort): You should select one use case to develop a system test plan.

***Note****:*

1. *The system test plan should aim at testing your subsystem on the same use case that you selected in Task 1(b).*
2. *It should contain test cases that cover all different scenarios of the use case.*

# Submission of Coursework

## When to submit

The submission deadline is at **23:00pm on 6th Dec. 2024 (Friday of Week 11)**.

## What to be submitted

Each group should submit one compressed (zip) file that contains a set of files listed in the table below. The file names and their contents to be included in the coursework submission must follow the convention given in the table below.

***Note****: The text in red ink below should be replaced by your coursework group number.*

Table 1. Files to be included in submission

|  |  |  |  |
| --- | --- | --- | --- |
| **File name** | **Format** | **Example** | **Content** |
| COMP6030\_CW\_ Gxx.zip | Zip file | COMP6030\_CW\_G01.zip | The zip file should contain all the files of the coursework submission, where Gxx is your coursework group number. |
| Cover\_Gxx.pdf | PDF | Cover\_G01.pdf | The standard group coursework cover page. |
| TeamPart\_Gxx.pdf | PDF | TeamPart\_G01.pdf | It should contain the work of Task 2(a), i.e. the architectural design of the whole system, including the UML component model and the textual documentation of the design. |
| Member\_Gxx\_StdNum1.pdf | PDF | Member\_G01\_1900001.pdf | The work of student 1900001, including one UML use case diagram, one UML activity diagram, one architectural design of the subsystem in UML component diagram, one UML class diagram for one component of the subsystem, one UML sequence diagram for the same component of the subsystem, one unit test plan, and one system test plan. |
| Member\_Gxx\_StdNum2.pdf | PDF | Member\_G01\_1900002.pdf | The work of student 1900002; *See above.* |
| Member\_Gxx\_StdNum3.pdf | PDF | Member\_G01\_1900003.pdf | The work of student 1900003; *See above.* |
| Member\_Gxx\_StdNum4.pdf | PDF | Member\_G01\_1900004.pdf | The work of student 1900004; *See above.* |

* Your files must be formatted as follows:
* It must be word-processed in 11-point Arial font and single-spaced
* All pages must be numbered
* Margins must be 1 inch on top, bottom, and both left and right sides of the paper
* All pages must in A4 size
* Student numbers must be entered on the cover page and state which subsystem is developed by which student.
* The files should not contain your name(s).

*Please note that you are required to cite the work of others used in your solution and include a list of references using the university recommended referencing style. Please refer to university’s regulation on academic conduct and plagiarism. If you have used AI empowered tools, you must declare how you have used it in the coursework.*

## Where to submit

You must upload **one** zip file containing all components to the module's Moodle website.

The original editable files must be uploaded to GitHub with all the history of updating the artefacts.

## Feedback on your coursework

* The formal summative feedback on your coursework will be provided within 2 weeks after your submission of the coursework.
* Informal verbal feedback can be obtained from the lecturer during the delivery of the module during practical classes. The students are encouraged to show their work to the lecturer and seek comments and advice.

1. No coding is required in this coursework. [↑](#footnote-ref-1)