# AC51003 – Assignment 2 – Submission Template

Please use this document as a template for submitting your assignment. It provides space for you to insert the content that is required for this assessment. It also provides space for me to provide you with feedback and a mark before returning the document back to you. The document page size is intentionally set to be A3 to ensure that there is a larger space for you to insert the content required. In some cases, the content will be in the form of UML diagrams, and these may require additional viewing space to be legible. NOTE: feel free to change the page orientation from portrait to landscape and/or a mixture of the two throughout - whatever seems sensible to be able to present your diagram content in the most legible way.

There are two main sections to this document that correspond to the two main components of the assessment: sequence diagrams and design patterns. You can complete each of these sections in whatever order you prefer. Prior to this, you will see below the summary of the assessment information along with an area where your overall mark will be placed.

**The first thing you should do is write your name in the space below**. Later, when you submit this document to My Dundee, please remember to ensure that you name the document file as follows: *surname\_firstname.docx*. Example: *ramsay\_craig.docx*. **Please also remember to submit the document as a Microsoft Word file**. I need it to be in Word format to be able to mark your assignment effectively and efficiently.

|  |  |
| --- | --- |
| **Please enter your name here:** | Rakshith Ramureddy |

## YOUR GRADE

|  |  |  |
| --- | --- | --- |
| ***Component*** | ***Weighting*** | ***Grade*** |
| Sequence diagrams | 80% | xx |
| Design patterns task | 20% | xx |
| COMBINED GRADE BASED ON WEIGHTED ELEMENTS: | | xx |
| NUMBER OF DAYS LATE: | | 0 |
| ANY OTHER ADJUSTMENTS, IF APPLICABLE | | N/A |
| **FINAL GRADE:** | | **xx** |

## ASSIGNMENT 2 -DESIGN MODELS

Assessment Summary:

|  |  |
| --- | --- |
| **Deadline:** | Sunday 2nd April 2023, 23.59pm |
| **Hand-in method:** | Upload as a Microsoft Word file to My Dundee |
| **Date for feedback:** | Sunday 23rd April 2023 |
| **Late penalties:** | One grade point per day late (meaning if a submission is one day late and marked as a C2 it will receive a C3 grade). A day is defined as each 24-hour period following the submission deadline including weekends and holidays. Assignments submitted more than 5 days after the agreed deadline will receive a zero mark (AB). |
| **Percentage of module:** | This assessment is worth 20% of the module grade. |
| **What to do:** | Prepare and submit sequence diagrams for the Vending Machine system (to be embedded / inserted below). Prepare and submit a report for the Design Patterns task (to be embedded / inserted below). |
| **Learning outcomes:** | This assessment addresses the following learning outcomes of the module:   * To comprehend and apply software design principles, practices, and notations. * To comprehend and selectively utilize visual notations that pertain to the analysis and design of software systems. * To employ design techniques: to formulate and convey behavioural designs for a software system. * To demonstrate awareness of, and to apply software design principles and software quality characteristics. * To critically evaluate the quality of a software design against known design principles and quality characteristics. * To develop and apply problem-solving, reporting, time-management, and independent study skills. |
| **Academic conduct:** | By submitting this file, you agree that the work that it contains is entirely your own work, that relevant sources have been referenced and made clear where applicable, and that your work has not been intentionally copied from or shared with others. |

## SECTION ONE – SEQUENCE DIAGRAMS (80%)

You are being asked to create sequence diagrams for different areas of the system functionality. These different areas of functionality are outlined below, one after another. For each area of functionality, a space is provided for you to insert your sequence diagram solution. Underneath this is a box into which I will be adding any feedback pertaining to your diagram(s) as they are being assessed. For convenience, each diagram is started on a new page. Make sure that you peruse the entire document to ensure that you do not miss anything out that occurs nearer the end, including the Design Patterns section. For each diagram, a summary of the relevant functionality is provided. Please note that it really is just a summary, and you can refer to your system use cases or requirements for a reminder of more details where required. Whilst the summaries of the functionality are brief, remember that the actual logic required to convey them in the sequence diagrams, as an interaction between objects in the system and using the relevant UML notation, may be more complex. You may also need to consider the content of the diagrams holistically. If some of the messages in one diagram are already covered by another diagram, it is not usually necessary to duplicate these same messages again, you can cross-reference them instead, whilst making sure each diagram is also complete with the messaging it requires.

### sequence diagram one: customer buys a drink using a vending card

The first diagram that you create should show the main case of a customer wishing to buy a drink using a vending card. Specifically, the steps which are summarized below:

1. The customer inserts their card.
2. The system should respond in the appropriate way, e.g., determining and showing the drinks available for the card balance.
3. The customer selects the drink they desire.
4. The system goes through the stages of dispensing the drink and finalising the purchase.

Alternative flows may be considered, and these may attract additional marks.

Please insert your diagram solution in the space between the two lines below:

*Diagram

Description automatically generated*

*Diagram

Description automatically generatedDiagram

Description automatically generated*

*These are the sub diagrams that are part of purchase drinks.*

1. *Display based on card inserted – shares same details with that od service operator sequence diagram. So I have created 2 of them and will be including both here*
2. *Display based on credit available.*

|  |
| --- |
| ***Feedback***  *I will be inserting any feedback in here…* |

### sequence diagram two: servicing the machine

This diagram should show how a service operator services the machine. Specifically, the steps which are summarized below:

1. A service operator inserts their service card into the machine.
2. The system completes the steps necessary to facilitate the service access, and being mindful of service notifications required, etc.
3. The stages of the service being completed should be conveyed too.

Alternative flows may be considered, and these may attract additional marks.

NOTE: it is not necessary to cover the case of the maintenance operator, just the service operator.

Diagram

Description automatically generated with low confidence

Diagram

Description automatically generated

Initiate boot up sequence is as shown below:

Diagram

Description automatically generated

|  |
| --- |
| ***Feedback***  *I will be inserting any feedback in here…* |

### sequence diagram three: stock notifications

This diagram should show the stage at which the system detects that the stock of a given drink has fallen below an acceptable level and so the system must send a notification to the service company. Alternative flows may be considered, and these may attract additional marks.

Diagram

Description automatically generated

|  |
| --- |
| ***Feedback***  *I will be inserting any feedback in here…* |

### sequence diagram four: Top-Up Card

This diagram should show the process for when a customer wishes to top up the amount of credit on their vending card. You can assume that the customer has already inserted their vending card into the machine. A summary of the process to convey is as follows:

1. The customer selects the top-up option.
2. The customer must enter the details of their bank / credit card and the amount of top-up they require.
3. The request must be authorised with the payment system.
4. The customer’s card will be topped up.

Alternative flows may be considered, and these may attract additional marks.

A picture containing timeline

Description automatically generated

|  |
| --- |
| ***Feedback***  *I will be inserting any feedback in here…* |

## SECTION TWO – DESIGN PATTERNS TASK (20%)

Please see the separate ‘Design Pattern Brief’ for what you are required to do. Please insert your solution below. You can insert the necessary text for your answer as well as the UML diagram(s) required too. A reminder of what to provide: (i) a description of the design principles(s) which are being violated in ‘Process Outline 2’. (ii) a brief description of the problems you have identified with the current design (in ‘Figure 3’); (iii) a brief explanation of why you think the Bridge pattern is relevant to this scenario; (iv) a UML diagram representing your refactored design, in which the Bridge pattern has been applied; (v) a brief explanation of the benefits of your refactored design / how it addresses the problems you identified in (ii) above; (vi) a sequence diagram which shows an example of how some of the classes in your newly refactored design can be used. Please provide your answer in the space below. Expand the space to accommodate your answer.

1. The 2 principles that are being violated are reducing coupling and increasing abstraction.

Reducing Coupling: Coupling occurs when packages, modules, classes, or files are very interdependent. This is not the best implementation of any system because if one package has any changes or if it breaks then the entire system is compromised. From the design pattern given to us we see that the there is more interdepend ability.

Increase abstraction: If something is more abstract then it is a simplified technique of something technical. If it is more abstract, then it is more generalized. From the design given to us, we clearly see that it is not.

1. From the fig 3 shown, say if there is a bug in Report Handler and if fails to generate report, then all the report are halted as main component that generates report is faulty. Now service notification team will not know that there is an issue at the facility not they get stock reports. Maintenance operator nor service operator will be sent to repair the issue. This results in huge loss to the company and loss of clients.

Also, we see that too many functions need to implement unnecessarily to send simple reports which could be comprehended using simple task / functions.

1. Bridge pattern aids in decoupling an abstraction from its implementation so that the two can vary independently.

So, we can use the bridge pattern to further modify the design and the modified design is as shown below.

1. Diagram

   Description automatically generated
2. Based on the new design, we see that 2 principles that was violated before has now been accepted. We have reduced coupling creating a new Communication implementation class and generalizing all the modes of communication. We increased abstraction and now the design looks simpler and, in the future, new designs can be implemented with ease.

Chart

Description automatically generated with medium confidence

|  |
| --- |
| ***Feedback***  *I will be inserting any feedback in here…* |