Assessment 3: Development Process and Artifact

Weighting: 40%
Due date: 5pm (Friday Week 11 and Week 12)

- 10% mark reduction per day applies in case of late submission

Learning outcomes: This assessment supports the following unit learning outcomes (ULOs)

- Implement a maintainable software system using Object-Oriented Principles (OOP);
- Effectively work and communicate in team-based software development projects;

Guidance for the use of Generative Artificial Intelligence (more details here)

Generative AI tools cannot be used in this assessment task.

In this assessment**,** you must not use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task.

Assessment overview

In this assessment, you will act as a software engineer to design and develop software based on the given requirements from clients using Agile software development practices, focusing on delivering the most valuable features/products to clients. Writing clean, maintainable, and efficient code is also a top priority.

Assessment details

In this assessment, you will act as a software engineer to design and develop text-based software based on the given requirements from clients.

Part A- Detailed Class diagram (5%)

You and your team will need to work out a detailed class diagram of the full system based on the best version of the initial class diagram. So, it is strongly recommended that you start this task after you have finished Assessment 2. You most likely need to make changes or refine your initial class diagram from A2, which is perfectly fine. For a detailed class diagram, you and your team need to:

1. Incorporate boundary and control classes into your initial class diagram.
2. Assign access/visibility modifiers and data types to the attributes in each class.
3. Identify ALL the operations that need to be performed by the system and assign those operations to the appropriate classes as methods with their access/visibility modifiers.

4. Include all the accessor and mutator methods, where appropriate.
5. Work out the formal parameters, as well as the return type for each method in your class.
6. Add a stereotype (<<Entity>>, <<Boundary>>, <<Control>>) to each class.
7. Revise the multiplicities and their directions.
8. Where possible, add direction to the association relationships.

Please be reminded again that in the detailed class diagram, you need to draw a detailed class diagram **of the full system.**

Please be reminded again that in the detailed class diagram, you need to draw a detailed class diagram of the full system.

Part B- Software Development and Delivery (25%)

(Breakdown: 15% code, 5% demo, 5% Process and Agile practice)

You are required to deliver a software with the following features and artifacts:

1. **Feature 1**: Customer Management
    a. The system should allow customer to log in with this email and password: Email: member@student.monash.edu, Password: Monash1234.
    b. Admin shall be able to login with Email: admin@merchant.monash.edu and Password: 12345678
    c. No need for a registration function, verify email, reset password, etc.
2. **Feature 2**: The system shall allow the supermarket to manage inventory products by adding new products or updating/removing an existing one.
    a. Sample data including available products needs to be generated and get entered into the system by the team.
3. **Feature 3**: The system shall allow the user to add their products to their shopping cart and proceed to checkout
    a. You need to deliver **only the basic functionality** of adding items to shopping cart and proceed to checkout, i.e. No need to deliver non-basic functionality like filter products or browse by category/price/etc, also no need to edit items in the shopping cart (e.g. increase/decrease quantity/empty shopping cart).
4. **Feature 4:** The system shall allow the user to Checkout
    a. This feature is only required if you are a team of 4 students. If you are a team of 4 students, you need to deliver **only the basic functionality of checkout,** i.e. no need to support discount/promo code, no need to choose between delivery/pickup option.
5. **A Readme File** containing how to configure your IDE and run your product. It is necessary to provide screenshots showing the above.
    a. Troubleshooting: Any troubleshooting-related information that you can provide.

Your team is required to follow Agile practices, focusing on delivering the most valuable features/products to clients. Your team will focus on software implementation based on 3 sprints (each sprint has 1-2 weeks), totalling over the next 5 weeks (Weeks 8-12). All software artifacts must be completed by Week 11, and subsequently be delivered and presented to your client by Week 12 (Demonstration).

Your team is required to actively use and maintain your Trello board. The Trello board should have the following columns: Backlog, Sprint 1, Sprint 2, Sprint 3, Todo, and Done. Frequent meetings are required to brainstorm and discuss about sprint planning, task estimation, task prioritisation, task allocation, etc.

For each sprint, we suggest to perform the following activities:

1. Start planning your sprint.
    a. What top-priority tasks/features should be in the sprint?
    b. What tasks should be performed?
    c. What features should be implemented and delivered at the end of the Sprint?
    d. How much effort is required?
    e. Take a screenshot of the Trello board at the beginning of the Sprint planning session.
2. Working on your product
    a. Your team is required to actively use FIT's GitLab version control system (https://git.infotech.monash.edu/fit5136/fit5136-s1-2024) to deliver working software products. Please do not use other platforms to do your version control. Any contributions outside of FIT's GitLab version control system will not be acknowledged, considered, and marked.
3. Wrapping up the sprint
    a. At the end of the sprint, you must demonstrate your working software at a sprint review (your applied session).
    b. You'll be expected to show your PO/client the new feature you have completed and merged into the main repository.

Below is the suggested sprint goal and tentative schedule for you.

Sprint	Sprint Goal
1 (Week 8-9)	Detailed Class Diagram + Deliver Feature 1 + 2
2 (Week 10-11)	Deliver Feature 3 [+ 4] + QA the product
3 (Week 12)	Demo

Submission Requirement for Week 11

- A PDF report focusing on Agile software development practices (Week 11)
    - The PDF report should include the following items:
        - The last version of your detailed class diagram, based on which your code is implemented, in a clear and readable format.
        - **A link to your Trello board** (please invite your tutor email address to for access).
        - **A link to your team's git repository:** The final project to be delivered should be on the **main branch** of your git repository
        - Three screenshots of the Trello board at the beginning of the Sprint planning session.
        - A summary of the individual contribution.
- A zip file of your software code from your GitLab repository.

Demonstration Requirement for Week 12

In the Week 12 applied session, your team will be asked to demonstrate software that you develop to your client (your tutor). The demo should be **strictly within 15 minutes** with an additional 5 minutes for Q&A. During the demo, you should focus on:

- Run the software and demo all the implemented features of your working software product.
- You need to have some sample data (e.g. products, categories, etc.) previously entered into the software, so that you can demo all the features.
- Note: Before running your software, you must update your local repository based on the code on the GitLab repository, via running "**git pull**" command or equivalent alternatives, to make sure the demo is according to the last version on GitLab.

Each team member must contribute equally to the demonstration. If a team member is absent, the team can still do the demo, but the absent member will receive a score of 0 for the demonstration part of the assessment.

Part C- Peer review in a group assignment (10%).

While Assessments 1-3 are all team based assignments, we acknowledge there may be differences in individual performance. To be fair to all students, we will apply individual variations to team marks based on our assessment of individual performance discernible from the available evidence (e.g., contributions on git, worklogs, evidence of activity on your project management tool, other evidence of

active participation in workshop tasks) and peer assessment/feedback (details to be shared closer to the time). Minor individual differences will not be made a big deal of during the marking in order to help preserve your teamwork spirit. However, individuals with little to no contributions to the team/project will receive little to no marks. Remember, we can also assess what is documented, so please document the evidence of your practice well. Having said that, there are plenty of opportunities to contribute and demonstrate individual and team performance. You can contribute towards various aspects of the sprint such as contribution to code (or test code) development, substantial planning, research, spike development/prototyping and agile process and management support.

Feedback Fruit Peer assessment will be performed to gauge the individual contributions to the group assignments. It will be used as an adjustment factor to an individual's grade based on the teammate's evaluation. Feedback Fruit will be made available after the submission deadline. **If the whole team did not complete the Feedback fruit, the entire assessment of the whole team will be awarded 0 marks**.

- 3% for Requirement Analysis (for Assessment 1)
- 3% for Software Design (for Assessment 2)
- 4% for Implementation (for Assessment 3)
