MA_Tuesday08am_Team002

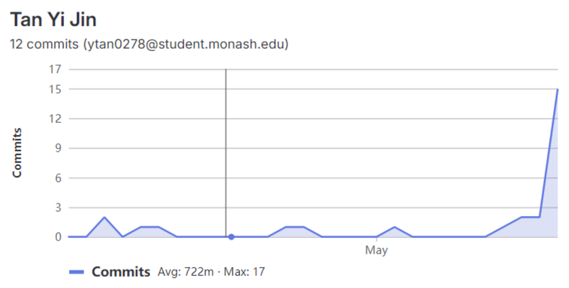
Team member 1: Amanda Goh

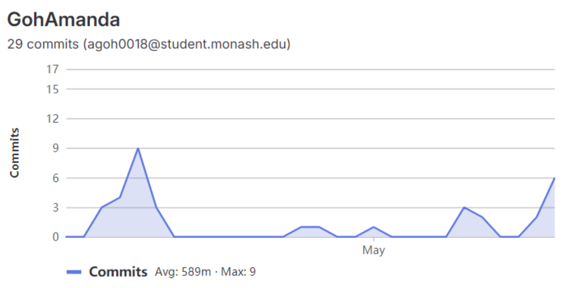
Team member 2: Tan Yi Jin

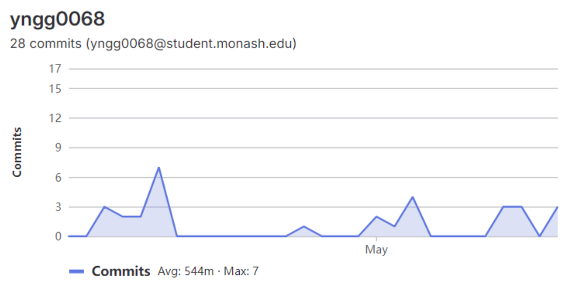
Team member 3: Ng Yu Mei

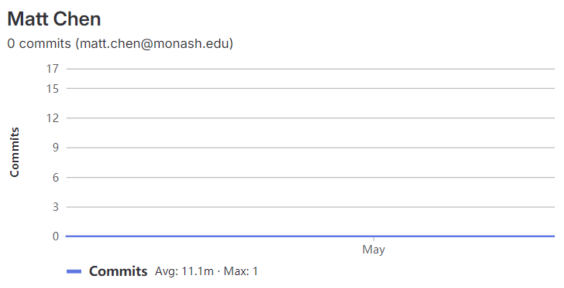
Contents

Contributor analytics









Self-defined extensions

Our team has decided to apply a timer for every player, which will restrict the amount of time they can use to flip a chit card, as a self-defined extension. By fostering a more dynamic and interesting gaming experience, this expansion seeks to further the human value of "Healthy." The game goes at a faster pace when each player's turn is accelerated, which lowers the possibility of prolonged inactive behavior and keeps players' minds active and engaged throughout the game. The time limit also encourages concentration and a sense of urgency, which can make the game more engaging and fun by keeping it from getting boring or unduly drawn out. As a result, this expansion supports encouraging a positive, active player base while also boosting the excitement of the game.

Class Diagrams

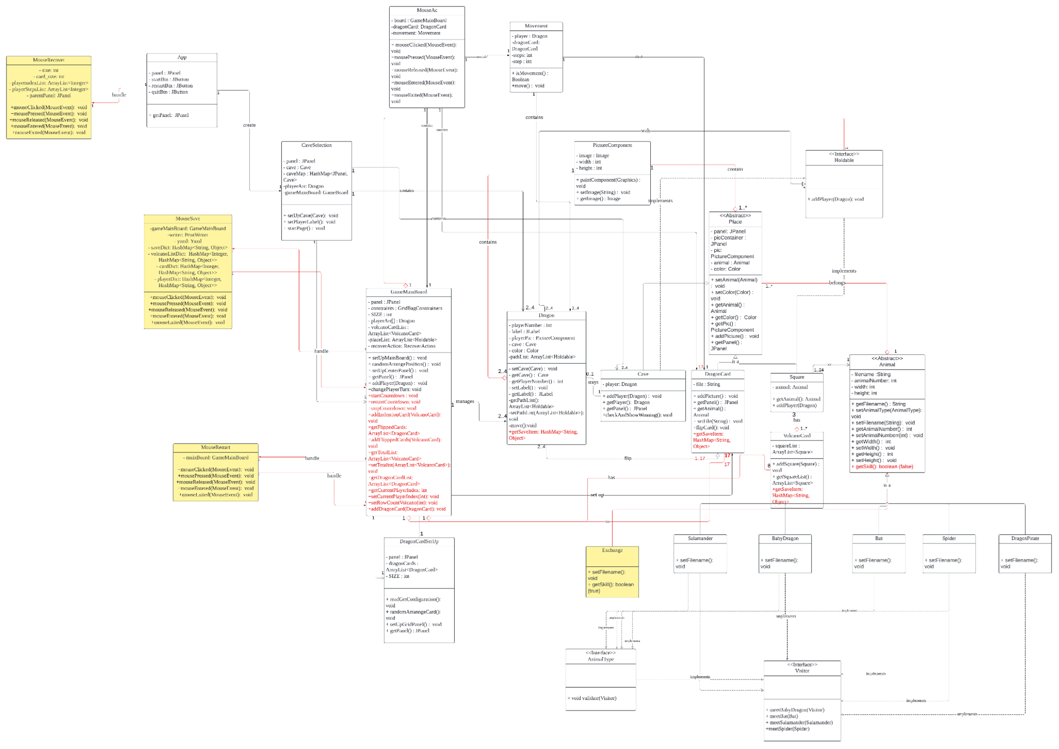


Diagram link: https://lucid.app/lucidchart/48d9fc54-85f2-4337-806e-b4486aaca0af/edit?invitationId=inv_fd663ed5-0e2e-417c-a50e-61e31bbb88b1&page=1LDF-8T2-74-#

We have added colors to our diagram to determine the differences that we have made for this sprint. A few new functions, classes, and some modifications of the relationships due to the feedback we have gotten for sprint 3 are added in our diagram. Some of the relationships have been modified due to the wrong symbol used and now have been modified to the correct symbol.

The 3 functions startCountdown, restartCountdown and stopCountdown is added in order to implement our self-defined extensions. This will be the few main function that will limit the time for each player to flip a chit card. This is being added in the GameMainBoard because the timer has a relationship with the flipping of the chit card and also affects the player's turn which were all triggered in the GameMainBoard. The startCountdown will trigger the timer and restartCountdown will reimplement the default values and trigger the startCountdown and stopCountdown again.

The swap class is added for our implementation of swap locations of dragon tokens. It is defined as a type of animal. We have added the getSkill function in the abstract class Animal and we will override the function in the swap class to return true.

The MouseRecover is added for loading the saved game from the configuration file. It is created as a domain model to ensure modularity since it is a unique function. App has an association relationship with the it because the recover button is instantiated in the App class. The App class is where the Start Page is implemented, and the recover button is displayed in the App class.

The MouseSave is added for saving the game into the configuration file. It is created as a domain model to ensure modularity since it is a unique function. It has an association relationship with the GameMainBoard and the GameMainBoard also has na association relationship with MouseSave class because all elements are set up in the GameMainBaord and GameMainBaord class has all the necessary information of other classes. MouseSave need to retrieve necessary information from GameMainBoard class. The save button is instantiated in the GameMainBoard class.

The MouseRestart is added for restarting the game. It is created as a domain model to ensure modularity since it is a unique function. It has an association relationship with the GameMainBoard and the GameMainBoard also has an association relationship with MouseRestart class because MouseRestart class requires information of GameMainBoard to restart the game board with same configuration of the original game board such as the number of player. Besides, GameMainBoard has an association relationship with MouseRestart class the restart button is instantiated in the GameMainBoard class.

Reflection on how much the design/implementation from Sprint 3 was suitable to incorporate the extended functionality

Self-extension: Timer countdown

Implementing the timer countdown into Sprint 3 was a relatively easier task.

The timer countdown feature was loosely connected to a number of pre-existing components, making integration simpler and the task's overall complexity lower. But managing timer resets and stops required extensive debugging to make sure the status of the timer was handled appropriately in a variety of scenarios, including system events and user interactions. Some sections of the original design used components that were tightly connected, which might have made it more difficult to integrate the timed countdown feature. However, because the timer module itself was designed to be loosely coupled, these code smells had a limited impact.

The GameMainBoard class and mouseAc class designs made it easier to build the restart countdown timer in Sprint 3. Because the chit card flip initiates the timer reset, the mouseAc class, which manages chit card flipping, proved indispensable for resuming the countdown timer. Because mouseAc's roles and event processing were clearly defined, it was the most practical class to employ when resuming the timer. Furthermore, by centralising the feature within the main game board and assuring cogent integration and control of the timer functionality, the GameMainBoard class made the entire development of the countdown timer simpler. If we had to go back and redo Sprint 3, we would put more emphasis on using design patterns, like the Observer pattern, to better handle time-based events. This would entail putting in place a more modular framework for state management and event processing, which would lessen the need for intensive debugging and increase the extensibility of the system.

Swap position

Implementing the player swap position with the closest token feature in Sprint 3 proved to be a challenging task.

The primary challenge arose from the feature's deep integration with several critical game elements, such as player movement logic, and overall game state updates. The logic behind player position swapping was closely intertwined with other game dynamics, making it difficult to isolate and adjust without inadvertently affecting other system aspects. Additionally, incorporating new features smoothly was challenging due to similar existing architecture or design patterns. We employed the Template Method design pattern by creating a new class, Exchange, which extends the abstract class Animal. When a player selects the exchange chit card, the current player's location is swapped with the nearest other player. This pattern helps avoid redundant code. Additionally, we used the Double Dispatch design pattern to define player movement steps each round without excessive if-else statements. If the player's steps encounter another player's cave, the steps increment by one, as the current player cannot enter another player's cave.

The Movement class's well-defined structure and capabilities proved useful in the implementation of the switch position feature. The switch position feature was more challenging to build, though, because of the path list that was part of the class and several cave-related components. Particularly, there were a lot of obstacles because of the way the path list was coded and the circumstances that kept players out of the cave. These components added complexity, which made it more difficult to integrate the new feature smoothly. We would change the cave access requirements and how the path list is implemented if we could go back to Sprint 3. To avoid running into the same problems when adding the switch position capability, this would need reworking these elements to be more adaptable and versatile. Modifying these components would simplify the implementation procedure, enhance maintainability, and lower the likelihood of errors or conflicts.

Load and save

For the loading and saving feature of the game, the level of difficulty is moderate to implement it into Sprint 3.

The path for the player to move created as with ArrayList, ‘totalList’ is easy to be stored in the YAML file and reloaded from the YAML file. All the information of the path was stored in one ArrayList,‘totalList’ makes it easy to handle what needs to be stored into a YAML file and reload from the YAML file. There is no information of path will be overlooked. The path of the gameboard was set up by only that ‘totalList’ so only ‘totalList’ needed to be saved in the YAML file. The ArrayList was used as a data structure for the path to arrange the position of the Volcano card in the correct order for the path, and the volcano cards can be arranged in the correct order when the game is reloaded from the YAML file. Since elements were set up in the GameMainBoard class, only one GameMainBoard type parameter needs to be passed into the MouseSave class(The class that implements MouseListener which is used to handle save features). All information that is required to be saved such as volcano card, player information, dragon card information can be retrieved easily and sent to the MouseSave class. However, volcano card class did not store any information about if it has indention to put the cave, so that it is hard to set the position to the cave when reloading the game without that information. Besides, there is less getter and setter function used in Sprint 3, which makes the gameboard information is hard to be retrieved by MouseSave class. Overall, implementing the loading and saving feature into Sprint 3 is a moderate task that most necessary information is easy to be saved from and reload to the system, since there is no significant change made in Sprint 3 codebase when implementing the saving and loading feature.

If we were to go back to Sprint 3, we will implement all getter functions and setter functions for encapsulated data. Since there is a lack of implementation of getter and setter functions, there is a limit in sharing of data and manipulation of data of other classes. This will make the system unable to work smoothly since most classes are related to each other. Implementation of getter and setter functions can protect the data, and share the data with different classes, and make it easy to build the application.

Executable Description

How to create executable file from source code:

1) First, open Visual Studio.
2) Open the project folder on Visual Studio.
3) At the bottom left corner, we can find the “JAVA PROJECTS” where we can find the symbol saying “EXPORT JAR”, click this symbol.
4) Then, we should select the main class of our project. In my case, App.java is the main class.
5) Finally, the jar file is created.

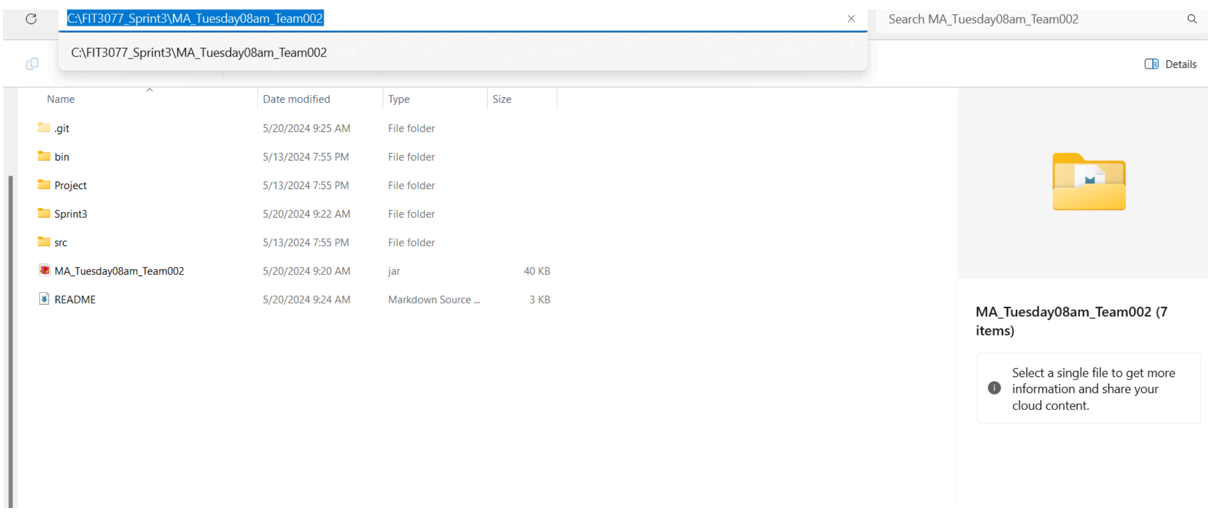
—---

How to run executable file

1) Download the zip file and unzip it.

2) Then, inside the extract folder, there is a jar file.

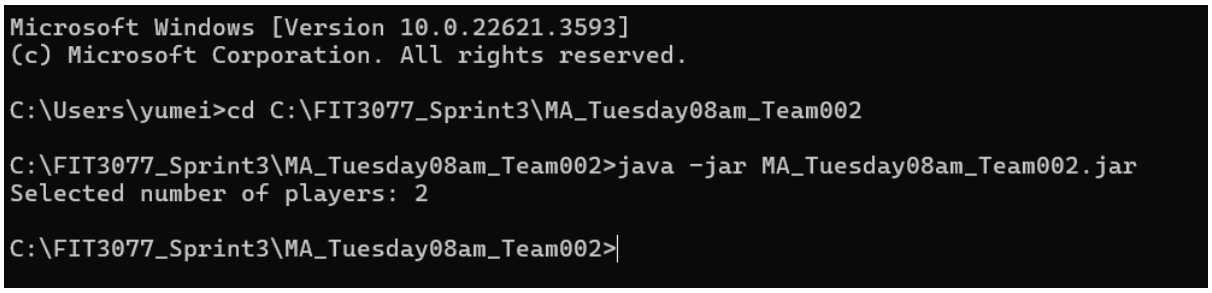
3) After that, go to the command prompt, use “cd {path}” to change directory to the folder where the .jar in. The path is the lead to the folder where the .jar file located. Copy the path(exclude the jar file name) . After changing directory to the location where the “.jar” is located in the command prompt. Use the command “java -jar {jarFileName}.jar to run and open the .jar file.


C:\FIT3077_Sprint3\MA_Tuesday08am_Team002

Name	Date modified	Type	Size
.git	5/20/2024 9:25 AM	File folder	
bin	5/13/2024 7:55 PM	File folder	
Project	5/13/2024 7:55 PM	File folder	
Sprint3	5/20/2024 9:22 AM	File folder	
src	5/13/2024 7:55 PM	File folder	
MA_Tuesday08am_Team002	5/20/2024 9:20 AM	jar	40 KB
README	5/20/2024 9:24 AM	Markdown Source ...	3 KB

MA_Tuesday08am_Team002 (7 items)

Select a single file to get more information and share your cloud content.

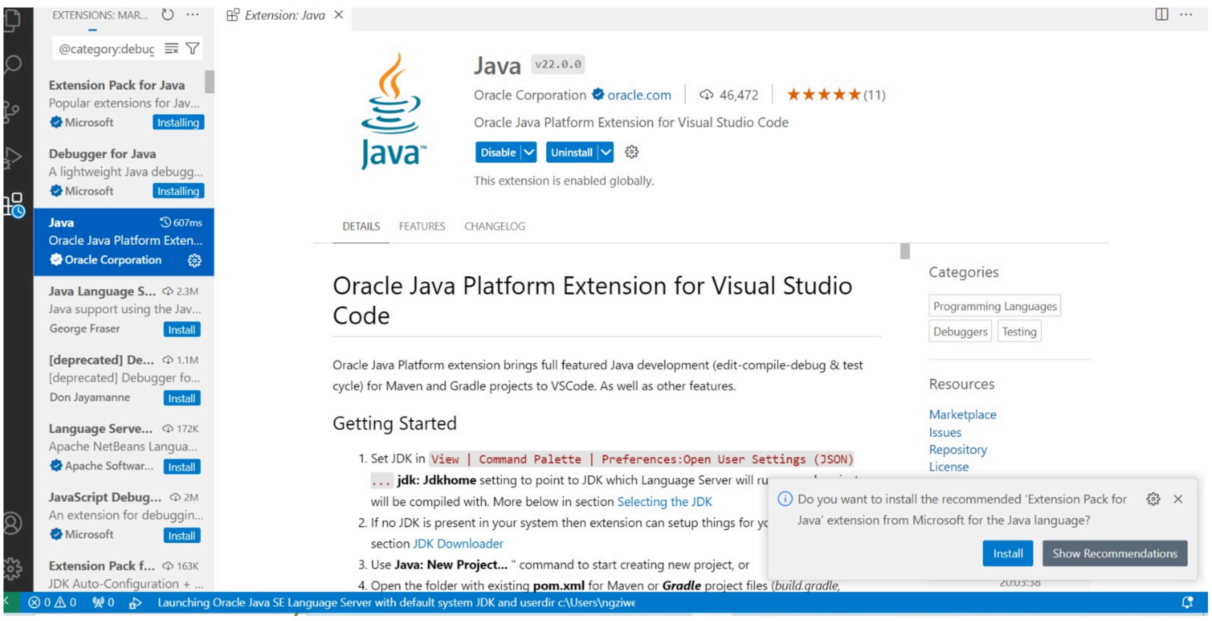

```
Microsoft Windows [Version 10.0.22621.3593]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yumei>cd C:\FIT3077_Sprint3\MA_Tuesday08am_Team002

C:\FIT3077_Sprint3\MA_Tuesday08am_Team002>java -jar MA_Tuesday08am_Team002.jar
Selected number of players: 2

C:\FIT3077_Sprint3\MA_Tuesday08am_Team002>
```

4) After the program is opened, the main interface of the program is displayed.

5) Aside from that, there are some requirements need to follow to ensure the program can run on your machine:

- Operating system: only run for Windows version 10 (64-bit) and above
- RAM (Memory) : 8 GB and above
- Processor: Intel Core i5
- Storage: 512 GB
- Platforms runnable: Visual Studio Code
- JDK/java version: 1.8.0_381 and above, better use the latest version
- Library: Java Swing and Java AWT

6) The tools need to be downloaded in Visual Studio are shown by pictures below:



Java v22.0.0

Oracle Corporation oracle.com | 46,472 | ★★★★★(11)

Oracle Java Platform Extension for Visual Studio Code

This extension is enabled globally.

Oracle Java Platform Extension for Visual Studio Code

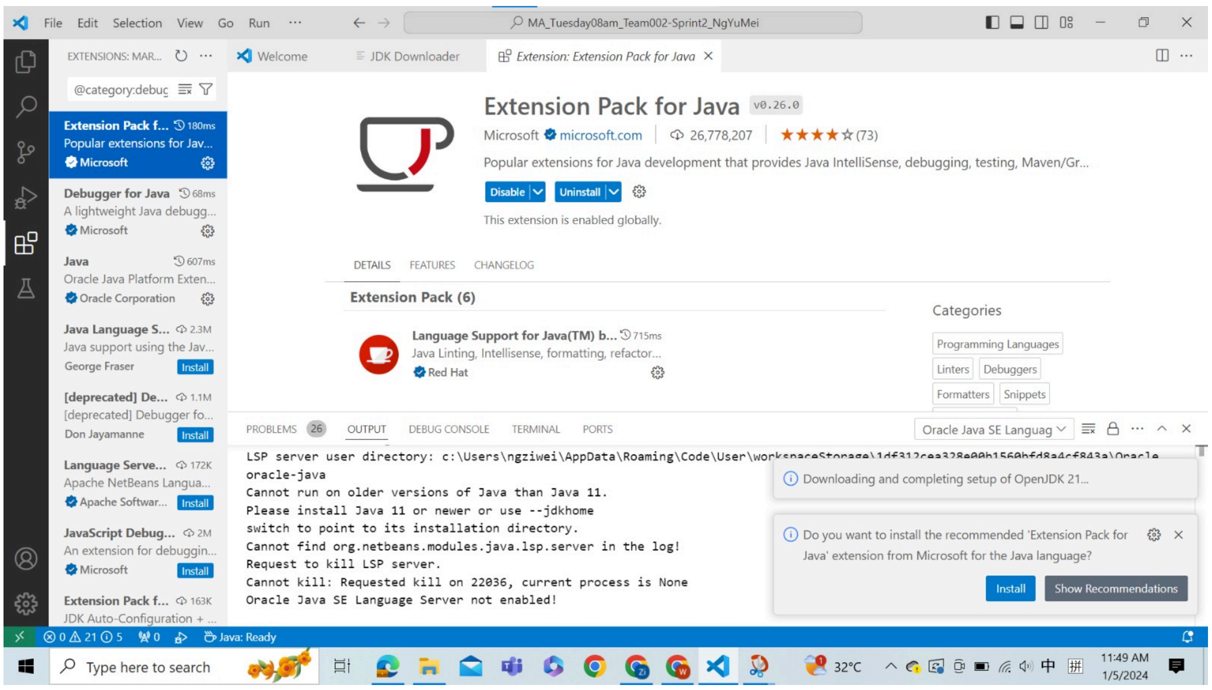
Oracle Java Platform extension brings full featured Java development (edit-compile-debug & test cycle) for Maven and Gradle projects to VSCode. As well as other features.

Getting Started

1. Set JDK in `View | Command Palette | Preferences:Open User Settings (JSON) ...` **jdk: Jdkhome** setting to point to JDK which Language Server will ru
will be compiled with. More below in section Selecting the JDK
2. If no JDK is present in your system then extension can setup things for yc
section JDK Downloader
3. Use **Java: New Project...** " command to start creating new project, or
4. Open the folder with existing **pom.xml** for Maven or ***Gradle*** project files *(build.gradle,*

Do you want to install the recommended 'Extension Pack for Java' extension from Microsoft for the Java language?

Launching Oracle Java SE Language Server with default system JDK and userdir c:\Users\ngziwe



Extension Pack for Java v0.26.0

Microsoft microsoft.com | 26,778,207 | ★★★★☆(73)

Popular extensions for Java development that provides Java IntelliSense, debugging, testing, Maven/Gr...

This extension is enabled globally.

Extension Pack (6)

Language Support for Java(TM) b... — Java Linting, Intellisense, formatting, refactor... — Red Hat

```
LSP server user directory: c:\Users\ngziwei\AppData\Roaming\Code\User\workspaceStorage\1df312cea328e00b1560bfd8a4cf843a\Oracle
oracle-java
Cannot run on older versions of Java than Java 11.
Please install Java 11 or newer or use --jdkhome
switch to point to its installation directory.
Cannot find org.netbeans.modules.java.lsp.server in the log!
Request to kill LSP server.
Cannot kill: Requested kill on 22036, current process is None
Oracle Java SE Language Server not enabled!
```

Downloading and completing setup of OpenJDK 21...

Do you want to install the recommended 'Extension Pack for Java' extension from Microsoft for the Java language?

<u>Any other aspects of your Sprint 4 work that you think are relevant for the teaching team to assess your submission?</u>

<u>ChatGPT and Quillbot declaration</u>

We acknowledge the use of Chatgpt and Quillbot to rephrase my paragraphs. I have entered the following prompts from 7th to 11th June 2024:

- Rephrase the following paragraphs/sentences
- What are the possible ways/patterns to make loading and saving easier?