# OOP2 Assignment 3 – 2022-2023 – Term 2

## Introduction

The total assignment consists of three parts, each with their own weight in the final grade. This is the first one. The deadline is January 29th 2023, 23:00. This is an individual assignment; it is not allowed to work in groups.

When scoring less than a 5.5 for an assignment you get feedback and have two weeks starting from the moment of feedback to hand in an improved version. When not handing in the first version before the deadline no feedback is given and the deadline for the improved version is February 12th 2023, 23:00.

## Context

## Grading

Your work is graded according to the following table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Weight** | **Topic** | **Excellent (10)** | **Good (8)** | **Sufficient (6)** | **Minimal (4)** | **Poor (2)** |
| 1 | GUI Design | The overall design of the graphical user interface follows industry standards  The interface has an intuitive approach  The interface is overall consistent  No new windows are opened to handle user actions; dialog boxes are limited to simple dialogs only  User mistakes and errors are gracefully handled | The overall design of the graphical user interface follows industry standards  The interface is overall consistent  No new windows are opened to handle user actions; dialog boxes are limited to simple dialogs only  User mistakes and errors are gracefully handled | The overall design of the graphical user interface follows industry standards  No new windows are opened to handle user actions; dialog boxes are limited to simple dialogs only  User mistakes and errors are gracefully handled | The overall design of the graphical user interface follows industry standards  No new windows are opened to handle user actions; dialog boxes are limited to simple dialogs only  *or*  User mistakes and errors are gracefully handled | The overall design of the graphical user interface follows industry standards |
| 2 | Architecture | The application has a correct architecture as defined in assignment #2  Correct use has been made of layout managers  All user screens have been defined in XML and @FXML annotations have been applied correctly  Actions resulting from user interactions have been bound to attributes and methods  User Interface Controls have been correctly bound to classes and their attributes  Design Patterns have been applied to ensure performance and maintainability | The application has a correct architecture as defined in assignment #2  Correct use has been made of layout managers  All user screens have been defined in XML and @FXML annotations have been applied correctly  Actions resulting from user interactions have been bound to attributes and methods  User Interface Controls have been correctly bound to classes and their attributes  *or*  Design Patterns have been applied where applicable | The application has a correct architecture as defined in assignment #2  Correct use has been made of layout managers  All user screens have been defined in XML and @FXML annotations have been applied correctly  Actions resulting from user interactions have been bound to attributes and methods  *or*  Custom designs and code have been put in place to ensure correct working of the application | The application has a correct architecture as defined in assignment #2  Correct use has been made of layout managers  All user screens have been defined in XML and @FXML annotations have been applied correctly  No use has been made of binding or Design Patterns | The application has a correct architecture as defined in assignment #2  Layout managers have not been used or have been used incorrectly  All user screens have been defined in XML and @FXML annotations have been applied correctly |
| 1 | Menu Structure | A menu structure has been defined and implemented in sync with the functional design  The implementation optimally uses layout management for this  Menu options not (yet) available have been defined but are not visible to the user  Menu options available give the result or action which can be reasonably expected by the user | A menu structure has been defined and implemented in sync with the functional design  The implementation optimally uses layout management for this  Menu options not (yet) available have been disabled but are visible to the user  Menu options available give the result or action which can be reasonably expected by the user | A menu structure has been defined and implemented. The structure implemented is logical but not in sync with the functional design  The implementation optimally uses layout management for this  Menu options not (yet) available have been disabled but are visible to the user  Menu options available give the result or action which can be reasonably expected by the user | A menu structure has been defined and implemented. The structure implemented is logical but not in sync with the functional design  The implementation optimally uses layout management for this  Most but not all menu options available give the result or action which can be reasonably expected by the user | A menu structure has been defined and implemented. The structure implemented is logical but not in sync with the functional design  Most but not all menu options available give the result or action which can be reasonably expected by the user |
| 1 | Basic Controls | The basic controls (such as fields and buttons) required according to the functional design have been implemented  Actions resulting from user interactions have been implemented correctly | The basic controls (such as fields and buttons) required according to the functional design have been implemented  All trivial actions resulting from user interactions have been implemented correctly  At least one non-trivial action resulting from user interactions has been implemented correctly (for example showing details after selecting an item from a list) | The basic controls (such as fields and buttons) required according to the functional design have been implemented  At least 50% of the actions resulting from user interactions have been implemented correctly  At least one non-trivial action resulting from user interactions has been implemented correctly (for example showing details after selecting an item from a list) | The basic controls (such as fields and buttons) required according to the functional design have been implemented  All trivial actions resulting from user interactions have been implemented correctly  At least one non-trivial action resulting from user interactions has been implemented; this implementation may or may not contain non-trivial but non-blocking errors | The basic controls (such as fields and buttons) required according to the functional design have been implemented  At least one trivial action resulting from user interactions has been implemented correctly  At least one non-trivial action resulting from user interactions has been implemented; this implementation may or may not contain non-trivial but non-blocking errors |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Advanced Controls | Two or more advanced controls (such as graphs) have been implemented according to the functional design  These controls have been bound to underlying classes according to the JavaFX standards  These controls correctly reflect the status of the classes they have been bound to  Actions applicable to these controls resulting from user interactions have been implemented correctly | Two or more advanced controls (such as graphs) have been implemented according to the functional design  These controls have been bound to underlying classes using a non-standard or custom solution  These controls correctly reflect the status of the classes they have been bound to  Actions applicable to these controls resulting from user interactions have been implemented correctly | One or more advanced controls (such as graphs) have been implemented according to the functional design  These controls have been bound to underlying classes using a non-standard or custom solution  These controls correctly reflect the status of the classes they have been bound to | One or more advanced controls (such as graphs) have been implemented according to the functional design  These controls correctly reflect the data they should, but in a custom way and without binding (so no updating because the data changed.) | One or more advanced controls (such as graphs) have been implemented according to the functional design  These controls do not reflect the data they should but show the data that could be |
| 1 | Using Collections | The application uses the most suitable Collections to show multiple items on the screen  The layout of the collection on the screen is user-friendly and user-centric  Actions can be taken on the collection by the user, such as opening a detail screen  Design Patterns have been used to ensure that the application will not freeze because of a user action | The application uses the most suitable Collections to show multiple items on the screen  Actions can be taken on the collection by the user, such as opening a detail screen  Design Patterns have been used to ensure that the application will not freeze because of a user action | The application uses the most suitable Collections to show multiple items on the screen  Actions can be taken on the collection by the user, such as opening a detail screen | The application does use Collections to show multiple items on the screen, but these Collections are not the optimal choice for the application  Actions can be taken on the collection by the user, such as opening a detail screen | The application does use Collections to show multiple items on the screen, but these Collections are not the optimal choice for the application  No user actions on the collection are possible |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Dependency management and releasing | Maven has been used for dependency management  The final application has been packaged into an executable format  A professional installer has been created | Maven has been used for dependency management  The final application has been packaged into an executable format  The release package does not require the user to perform additional downloads or actions | Maven has been used for dependency management  The implemented code functions with no more than trivially correctable errors  A custom installer has been created where another solution would have been preferable | Maven has been used for dependency management  The implemented code can be executed without immediate severe errors  The final application has been packaged into an executable format | Maven has been used for dependency management  The code compiles correctly but cannot be executed without throwing exceptions |

If a criterion does not achieve the standard described under Poor a score of Absent (0) is assigned. For each topic you score between 0 (Absent) and 10 (Excellent) points. The base grade is the weighted average of points scored, with the weights as indicated in the table.

All criteria need to score at least the green level; if no level for a criterion is green there is no minimum. If one or more criteria score one level below the minimum score the maximum grade is a 4.0. If one or more criteria score two or more levels below the minimum score the maximum grade is a 2.0.

**If the work handed in cannot be imported as a Java or Maven project into Eclipse the grade will be a 1.0, and no feedback is given.**