**OOP2 Assignment (25%)**

The purpose of this assignment is to demonstrate your ability to apply the learning from the module to build a Java application demonstrating features from the latest Java LTS (Java 21). The application is expected to cover the following language features from the module:

**Fundamentals**:

* lambdas: for example: *Consumer, Predicate, Supplier, Functio* etc..
* streams 
  + terminal operations
    - *min(), max(), count(), findAny(), findFirst(), allMatch(), anyMatch(), noneMatch(), forEach()*
    - *collect()* - *Collectors.toMap()*, *Collectors.groupingBy()* and *Collectors.partitioningBy()*
  + intermediate operations e.g. *filter(), distinct(), limit(), map()* and *sorted()*
* *switch* expressions and pattern matching
* sealed classes and interfaces
* Date/Time API
* records

**Advanced:**

* collections/generics - for example: use of *Comparator.comparing()* for sorting
* concurrency e.g. using *ExecutorService* to process a list of *Callable*’s
* NIO2
* Localisation

**Extra marks:**

* any topics from Java 22 and/or 23 - be sure to clearly explain these topics and what you had to do to get the newer code to compile/run
  + **specifically, only unnamed variables and patterns required here (permanent in Java 22).** 
    - [**https://docs.oracle.com/en/java/javase/22/language/unnamed-variables-and-patterns.html#GUID-D54E1CF1-BDFD-4B57-8A6E-5B4C87F4D58A**](https://docs.oracle.com/en/java/javase/22/language/unnamed-variables-and-patterns.html#GUID-D54E1CF1-BDFD-4B57-8A6E-5B4C87F4D58A)
* consistent updates to a public repository

**Application Domain**

You are free to select your own domain. Please select a domain that makes sense e.g. a car park/restaurant application. You are free to reuse your OOP1 project if you wish and re-factor it for OOP2. **However, if you had an exemption for OOP1 then you need to design one for OOP2**.

**Deadline**

1700 Friday 4th April. If you cannot submit on that date due to illness or other extenuating circumstances, you will have to request a deferral by emailing [eng@tus.ie](mailto:eng@tus.ie). A live Q&A via zoom may be scheduled after the submission date if seemed necessary - you will be notified by student email if this is the case. The zip to upload should contain your report, code and screencast.

**What to submit**

|  |  |
| --- | --- |
| Report | A brief introduction to the application. |
| List of user stories completed. Bear this in mind:  **as your Java Lecturers, we want to see features from the language as user stories, as opposed to functionality of the application**. |
| Evaluation - evaluation (1-2 pages) of how well you adhered to the project brief and any problems encountered. |
| Code | Your source code |
| Video screencast | Brief overview of your architecture (Optional UML diagram ) |
| Demo of all your user stories. Please refer to your code when explaining these (and not the UML). **From our perspective, we need to know that you understand the code**. |
| Max. time is 10 mins |
| Ensure you annotate your video accordingly; the better explained it is, the easier it is to give marks. |
| Audio quality is very important. |

**Marking rubric**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Elements | Excellent (75+) | Good (55%-75%) | Satisfactory  (40%-55%) | Fail (0-39%) |
| Presentation quality  (20%) | Excellent audio.  Presentation demonstrates adequate preparation. Content is presented in a coherent and logical fashion. Appropriate use of visual annotations. | Good audio.  Presentation demonstrates adequate preparation. Content is presented in a coherent and logical fashion. | Good audio.  Presentation demonstrates adequate preparation. | Poor audio.  Presenter does not adhere to the maximum time limit. |
| Application demo (user stories/requirements)  (50%) | Presenter demonstrated a complete application (both fundamentals and advanced language features).  Excellent understanding of the application and language features used. | Presenter demonstrated a more complete application (not all language features implemented).  Good understanding of the application and language features used. | Presenter demonstrated an application with a good level of functionality (all fundamental language features implemented).  Good understanding of the application and language features used. | Presenter demonstrated minimal or no functionality at all. Very few fundamental language features implemented.  Weak understanding of the application and language features used. |
| Code quality (10%) | Code is of a very high standard and is very easy to follow. | Code is of a high standard and easy to follow. | Code is of a good standard and relatively straightforward. | Code is of a low standard and hard to follow. |
| Evaluation (20%) | Application evaluated against the project brief.  **Java 22/23 used**.  Consistent repo updates. | Application evaluated against the project brief. | Minimal or no evaluation. | No evaluation done. |