Object Oriented Constructs / Systems Analysis & Design

Module Title(s):	Object Oriented Constructs / Systems Analysis & Design
Assignment Type:	Pairs Practical Project
Project Title:	**Planning and Implementing an Object-Oriented Software System**
Project Date:	**March 2021 to May 2021**
Assignment Compiler:	Michael Weiss (mweiss@cct.ie) Amilcar Aponte (amilcar@cct.ie)
Weighting:	30% SAAD / 50% OOC
Due Date:	**Saturday, 15th May 2021 @ 23:59** **Late submissions will be accepted until the 20th of May @ 23:59. All late submissions are subject to a penalty of 10% of the mark awarded.**
Method of Submission:	Moodle Submission **No e-mailed assignments accepted**

Assignment Introduction

Xtra-vision is a software system that is inside a Video rental kiosk. The Kiosk is placed in the lobby of shopping centres and grocery stores throughout Ireland. The kiosk allows customers to rent movies. Details of this system are available here at this link: **https://xtra-vision.ie/how-it-works/**

The Xtra-Vision company has signed a contract with your team to design and write the software for this Kiosk. You will be combining skills from two of the classes that you have this semester, to design and implement a Java program that will provide all of the functionality of the system described in the link above.

This an integrated assessment between the following year-2 modules: Object Oriented Constructs (OOC) and Systems Analysis and Design (SAAD). For the SAAD class you will do the planning, design, and UML modelling of your proposed system. For the OOC class you will use Object Oriented programming best practices to design and create your proposed system. In other words, you will build a working protype of the system.

Module Learning Outcomes Assessed

Module Learning Outcomes for Object Oriented Constructs:

- Apply best practices in Object Oriented Programming and Constructs in the production of software systems.
- Write robust Object-Oriented software.
- Utilise Object Oriented features and constructs such as Packages, Nested Classes, Inheritance, Polymorphism, Enums, Abstract Classes, Interfaces and Collections to provide elegant solutions to appropriate problems.
- Understand the benefits of using IDE tools in software development in terms of development, deployments, packing, code conventions and version control.

Module Learning Outcomes for Systems Analysis & Design:

- Implement UML-based planning during the software development process
- Devise clear roadmap for a software project to aid the future direction of a company based on direction and feedback
- Select the testing needs of a piece of software and design automatic and manual tests to address these
- Keep abreast of new developments in Systems Analysis and Design
- Assess a piece of software code and develop a collection of UML and design documents for the software

Specific Requirements

Your team will be writing a program that provides the user with all of the functionality that is described above regarding the Xtra-Vision video rental kiosk. Your team will do all of the planning in relation to the SAAD class and you will do all of the object-oriented Java programming in relation to the OOC class. You will be creating a report, and in your report, you should include a reflection on how you divided up all the tasks and state clearly what each team member contributed to the overall planning and prototyping of this system.

- SAAD class requirements: For the software planning part of the project each team you are required to create the requirements and functional analysis of the system, including:
    - A problem definition that describes this entire e-services type of system and details specifically what problems the Xtra-Vision software system will solve.
    - A full list of requirements regarding what the system.
    - Four Use Case diagrams that fully describe four different scenarios in which a user will use the Xtra-Vision movie kiosk. Include at least one ‘failure’ scenario along with the traditional ‘successful’ scenarios.
    - A Class diagram that will properly illustrate the relationships between classes.
    - A Sequence diagram that will properly illustrate the flow of the system functionality.
    - A State Diagram modelling the state of the system as a movie gets rented.
    - An Activity Diagram which models the entire flow of activity. Include ‘swim-lanes’.
    - One User Story, following standard Agile practices, that details a user interaction feature of the Xtra-Vision system that you would like to have included in a future version that is not currently part of the original system description.

- OOC class requirements: You are required to implement a Java program as per the requirements described on the website above, including:
    - Working prototype of the proposed software solution.
    - As you will be working in teams, you need to use a GIT repository to keep track of all the changes that you and your teammate are doing. This repository can be kept private during the development process, but it must be made public after the submission time, so the lecturers can review it a part of your assessment.
    - You should make use of abstract classes and/or interfaces when moving from the planning phase to the development part.
    - The minimum requirement is to develop a command line program. If a graphic user interface is implemented, this will be considered as distinction work.
    - As this is a prototype, all data can be stored in memory while the program runs. Data persistency would be considered as distinction work. This can be achieved through text files, JSON file, database interactions or any other mean of your choice.

Distinction work:

Object Oriented Constructs

If you would like to achieve a distinction, consider adding some extra layers of functionality, such as, but not limited to the following items:

- Create dynamically some testing data (brand new random data in every run of the program).
- Implementing data persistency through a txt file, JSON file or an external database.
- Implement a Graphic User Interface for you program.
- Sending a real confirmation email.

In all of the cases, make sure to adhere to the principles of object orientation and good practice. To achieve these distinction marks, at least two extra layers of functionalities should be implemented.

System Analysis & Design (for additional marks). Attempt these only if you have completed the design and coding of the system and you have time to dedicate to doing some additional research.

- Research item 1: Explore the testing needs of the finished Xtra-Vision software system and explain how automatic and manual tests are created to address the testing needs of a Java based object-oriented software project (research and references required). Discuss any testing that you performed while implementing this system.
- Research item 2: Dealing with design complexity:

  Software systems can start off simple and quickly grow to be quite complex and therefore when developing a computer system, the developer will often be dealing with a certain amount of complexity. Explain (in your own words) how modelling techniques manage complexity in the development of computer-based systems. Your answer should include a description of how models can support software developers who are striving to understand the complexity of the systems that they are developing. Be sure to include factors that are typically used to deal with complexity when analysing a system during the UML modelling process (research and references required).

Deliverables

You and your teammate must submit **one compressed folder** containing:

- Full NetBeans project containing the source code of the working program and any other external dependency that you used.
- One PDF document, including your analysis and diagram submitted in report format.

This must be uploaded to the Moodle uploader links that are posted on **both the OOC and SAAD classes** (upload it to both Moodle links) by both members of the team (each team member makes their own submission).

Notes

- Your project must make use of the four principles of object orientation: Encapsulation, Abstraction, Inheritance and polymorphism.
- The use of other structures such as enums, packages, wrappers classes, among others, are also recommended.
- Comment your code!!
- You will be working in teams (groups of two, no more and no less) so pick someone who is good to work with. But don't just pick someone because they are your Friend/Amigo. Choose a team partner who will be just as focused and hard working as you are… Someone who is willing to put in the effort to create a very high standard of work that you will both be proud of.
- In any situation, the lecturer is entitled to call you in for further explanation of your code.
- **Your code must run as no debugging will be done.**
- **Plagiarism will not be tolerated. All code must be your own. If you used some snippet of code from an external source, make sure that you reference it correctly inside your code. The same applies to all of your diagrams.**

Marking Scheme Summary

Object Oriented Constructs

Description	Weighting
Minimum requirements	
GIT Repository: - Correctly set up - Includes both members of the team - All development process has been well documented	5
Renting Movie(s) - Movie Selection has been correctly implemented as per specification, including limitation of number of movies. - Checkout and payment have been correctly implemented - Validations have been implemented - Relevant error messages are displayed if needed - Appropriate data structures have been used - Principles of object orientation have been applied - Code is well commented	45
Returning Movie(s) - Returning of the movie(s) - Optional email registration for confirmation - Validations have been implemented - Relevant error messages are displayed if needed - Appropriate data structures have been used - Principles of object orientation have been applied - Code is well commented	20
Additional Functionalities	
Implementation of extra layers of functionalities such as: Dynamic data creation, GUI implementations, data persistency, email sending, among others. - At least two of the above (or others) have been implemented - Functionality works correctly - Validations have been implemented - Relevant error messages are displayed if needed - Appropriate data structures have been used - Principles of object orientation have been applied - Code is well commented	30
TOTAL	**100**

ASSESSMENT OF INDIVIDUAL CONTRIBUTION

Each group member inherits the overall group project mark. This mark is then adjusted by their individual contribution. Individual contribution will be assessed by reference to your GIT repository.

For example:

- Final group result: 76%
- Student with perfect contribution receives: 100% * 76% = 76%
- Student with just sufficient contribution receives: 50% * 76% = 38%

Therefore, it is important that both of you ensure that you are contributing to the coding work!

System Analysis & Design

Description	Weighting
An accurate problem definition, an accurate list of system requirements, full description of this type of system	30
UML Modelling of the Xtra-Vision system using UML diagrams	40
Research regarding software testing for this type of software project Research regarding dealing with complexity	20
Report content, presentation, and referencing	10
TOTAL	**100**