



2019-2020 Semester 2

CT548 Object Oriented Design and Development

## Individual Assignment 4

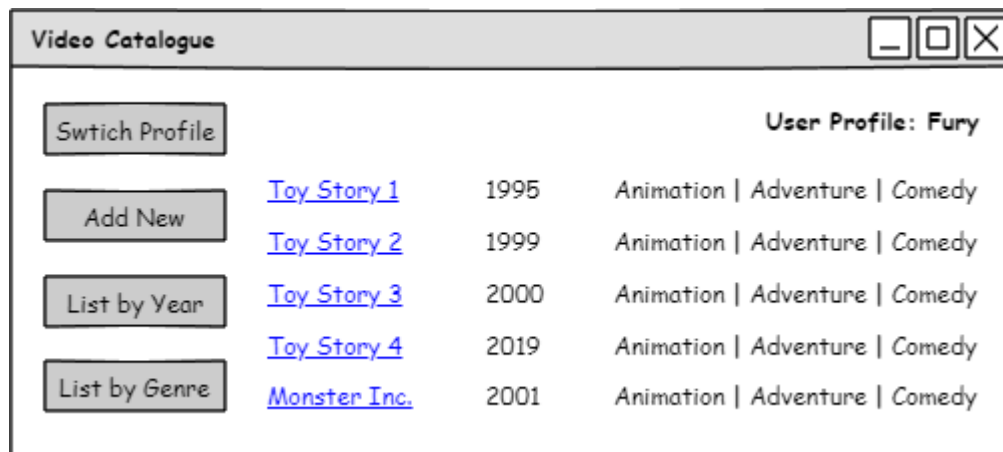
<b>Objective</b>	<p>Given the requirements of a software system</p> <ol style="list-style-type: none"> <li>1) analyse the requirements and design the system using UML,</li> <li>2) implement the system using Java language and tools,</li> <li>3) use a test-driven development approach,</li> <li>4) document the system</li> </ol>
<b>Lecturer(s)</b>	<p>Dr Umair ul Hassan                      umair.ulhassan@nuigalway.ie</p> <p>Dr Alessandro Adamou                      alessandro.adamou@nuigalway.ie</p> <p>Office: Data Science Institute</p>
<b>Marks Awarded</b>	<p>The marks awarded for this assignment are worth <b>50%</b> of the overall marks for the CT548 module.</p>
<b>Submission</b>	<p>Submission deadline is <b>11:59am</b> on <b>Thursday, April 23<sup>rd</sup>, 2020</b>.</p> <p>The solution must be <b>submitted through Blackboard</b>.</p> <p>Provide a <b>single archive file</b> of your diagrams, code and documentation as detailed in the submission guidelines on Blackboard.</p>
<b>Late Submission or No Submission Policy</b>	<p>Late assignments will incur a <b>penalty</b> depending on the delay.</p> <p>After <b>48 hours</b> past the deadline, submissions will no longer be accepted.</p> <p>Note that only the <b>last submission will be graded</b>, whether submitted before or after the deadline. The timestamp of the last submitted attempt counts towards submission punctuality.</p>
<b>Academic Integrity</b>	<p>You are expected to adhere to the highest standards of integrity and honesty when completing assessments at NUI Galway. We reserve the right to follow up with a student by interview if there is any concern in relation to the integrity of the assessment. Any irregularities of conduct, including plagiarism, may be reported to the appropriate forums in NUI Galway. <a href="http://www.nuigalway.ie/plagiarism/">http://www.nuigalway.ie/plagiarism/</a></p>

*Please ensure that your submitted files follow the submission guidelines and package structure.*

### Problem

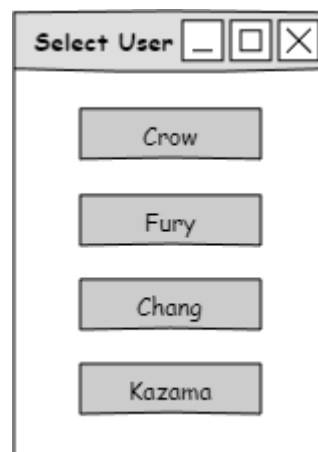
Design and implement a media catalogue that allows a user to browse through films and TV series and to create their own media library from the catalogue. For each item, the catalogue contains title, description, year, genre, director, and cast. When the application is started, a default user profile is shown to the main screen as depicted in Figure 1. The figure shows four buttons on the left, five catalogue items based on user's preferred genre, and the current active profile. Each catalogue item is listed in terms of title, year, and genre. The title for each item links to a new screen to show its details. The buttons link to four screens.

1. The "Switch profile" button is used to select another user profile.
2. The "Add New" button is used to create a item entry for a video in catalogue.
3. The "List by Year" button is used to browse all items in the catalogue according to the year.
4. The "List by Genre" button is used to browse all items in the catalogue according to genre.



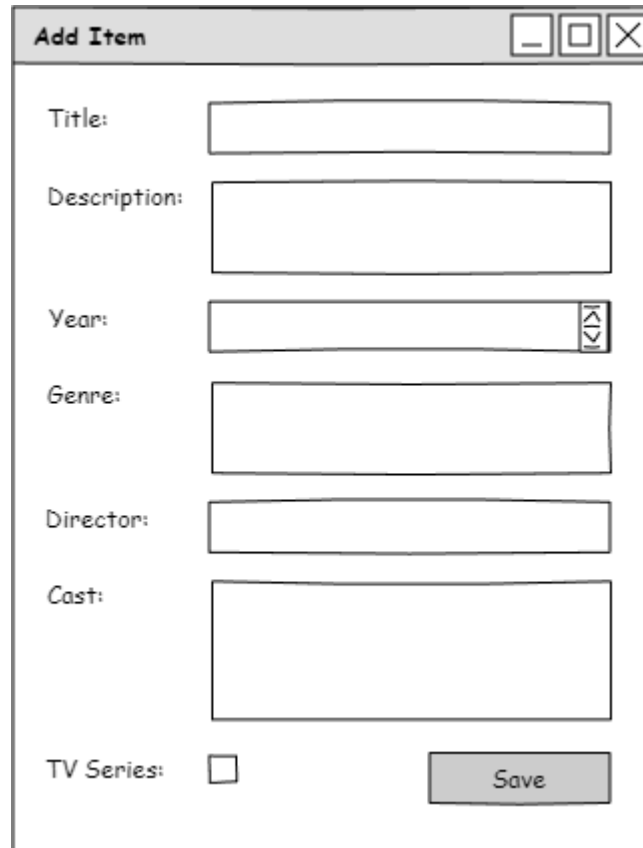
*Figure 1: Main screen with active user profile called "Fury"*

When the user clicks on the "Switch Profile" button in the main screen, the profile selection screen is displayed, which contains a button for each user profile. Figure 2 shows an example screen with four user profiles. Once the user clicks on a profile button, the profile selection screen is closed, and the main screen is updated with the chosen profile.



*Figure 2: Profile selection screen with four user profiles.*

When user clicks the “Add New” button in the main screen, the “Add Item” screen is displayed that contains seven input fields to collect information about a new catalogue item. The user enters the details of the new item and presses the “Save” button that saves the data and closes the “Add Item” screen.



The image shows a window titled "Add Item" with standard window controls (minimize, maximize, close) in the top right corner. Inside the window, there are seven input fields arranged vertically, each with a label to its left: "Title:", "Description:", "Year:", "Genre:", "Director:", "Cast:", and "TV Series:". The "Year:" field has a small spinner control on its right side. The "TV Series:" label is followed by a checkbox. At the bottom right of the window is a "Save" button.

***Figure 3: Add item screen with seven input fields***

When user clicks the link on the title of an item, the “Item details” screen is displayed that contains seven fields to display information about the linked catalogue item. User views the details of the linked item and presses the “Back” button to close the screen.

**Item Details**

Title: Toy Story 4

Description: When a new toy called "Forky" joins Woody and the gang, a

Year: 2019

Genre: Animation | Adventure | Comedy | Family | Fantasy

Director: Josh Cooley

Cast: Tom Hanks  
Tim Allen  
Annie Potts  
Madeleine McGraw

Back

**Figure 4: Item details screen with six item fields**

*Hint: The UI of the “Add Item” screen can be reused for “Item details” screen with disabled fields.*

When user clicks the “List by Year” button in the main screen, the “List by Year details” screen is displayed that contains the list of all items in catalogue grouped and ordered according to year. The items are ordered according to their title in each year group. The title of each item is linked to its details screen like the main screen.

**List by Year**

Year	Item Title	Genre
2019	<a href="#">Jumanji: The Next Level</a>	Action   Adventure   Comedy
	<a href="#">Toy Story 4</a>	Animation   Adventure   Comedy
2001	<a href="#">Monster Inc.</a>	Animation   Adventure   Comedy
2000	<a href="#">Toy Story 3</a>	Animation   Adventure   Comedy
1999	<a href="#">Toy Story 2</a>	Animation   Adventure   Comedy
1995	<a href="#">Toy Story 1</a>	Animation   Adventure   Comedy

**Figure 5: List by Year screen showing catalogue items according to years**

When user clicks the “List by Genre” button in the main screen, the “List by Genre” screen is displayed that contains the list of all items in catalogue grouped and ordered by genre. The items are ordered according to their title in each genre group. The title of each item is again linked to its details screen like the main screen.



Figure 6: Item details screen with six item fields

### Modelling requirements

1. The system must be designed and implemented using model-view-controller (MVC): user interface code shall never directly access the data.
2. In each view, the items must be sorted first by the selected dimension (genre if in the “List by genre” view etc.), then by title.
3. People (e.g. directors, cast members) must also be modelled as objects in the code. For simplicity, equality is by name alone: two people with the same name are the same person.
4. The generation of new objects for films and TV series must be implemented using a factory pattern (Abstract Factory or Factory Method).
5. Each view uses different criteria for sorting and grouping media items: implement a Strategy pattern to assign them at runtime.
6. It is enough that created catalogue items and user profiles remain in memory: it is not required to store the data to a database or file.
7. The initial data in the catalogue must be loaded from the JSON file provided with the assignment in Blackboard. The JSON will contain initialization data about users, catalogue items, people, and genres.

### Part 1: DESIGN

(15 marks)

Analyse the above requirements and provide the detailed design of the system using following UML diagrams.

1. Use case diagram.
2. Complete and detailed class diagrams of the system.
3. Detailed sequence diagrams that show the interaction between actors and objects for each use case.
4. Package diagram that details the components and their relationships.

*Note: submit all diagrams at a separate folder as part of a single archived file (zip or gzip).*

**Part 2: IMPLEMENTATION**

**(20 marks)**

Implement the system according to your design to ensure that it fulfils the functional requirements. Following the MVC pattern, separate your code in following packages.

1. Model to contain class(es) that define business logic
2. View to contain the class(es) for definition of the user interface using Swing
3. Controller that contains class(es) that translate the events in View to messages in Model
4. Runtime that contains the code to setup and start the application.

*Note: submit all packages according to the Maven project structure as part of a single archived file.*

**Part 3: TESTS**

**(7 marks)**

Create the following tests for your system:

1. Units tests for each class in the Model
2. Integration tests that test the combined functionalities of two or more classes in Model.

*Note: submit all tests in a separate package in the same project archive (src/test).*

**Part 4: DOCUMENTATION**

**(8 marks)**

Create detailed documentation for the developed system. It should contain the following sections.

1. Describe the reasoning behind your design choices in terms of design patterns, classes, and user interaction.
2. Describe the test scenarios that you have created and their objectives.
3. Provide a small guide for the using the system (i.e. help for the user)
4. Discuss how the media item creation subsystem can be extended to create movies and series that are physical (e.g. on DVD or Blu-ray) separate from digital ones (e.g. streaming or downloadable files).
5. Discuss at least one more way in which the system can be extended.

*Note: submit the documentation as a README.md markdown file. Use headings to separate each section appropriately in this file.*

**Useful links:**

- MVC application with Java 8 and Swing <https://medium.com/@ssaurel/learn-to-make-a-mvc-application-with-swing-and-java-8-3cd24cf7cb10>
- Markdown Guide <https://www.markdownguide.org/>
- StarUML <https://en.wikipedia.org/wiki/StarUML>
- PlantUML <https://en.wikipedia.org/wiki/PlantUML>
- MODELIO <https://en.wikipedia.org/wiki/Modelio>