# Assignment 2 Analyzing the Domain

### 1. Use noun/verb analyses to find the classes involved in each use case.

Title class

* Attributes: title information (e.g., title no., name, author, ISBN, and type), On-loan indicator
* Associated classes: Item
* Methods: add, remove, update

Item class

* Attributes: item information (e.g., item no.), On-loan indicator
* Associated classes: Title
* Methods: add, remove, update

Borrower class

* Attributes: borrower information (e.g., Last name, first name, address, city, postal code, province)
* Associated classes: Item, Reservation
* Methods: add, remove, update

Reservation class

* Attributes: reservation information
* Associated classes: Borrower, Item
* Methods: add, remove, update

For use cases (Actor: Librarian)

* add title: class: Title, method: add
* add item: class: Item, method: add
* remove or update title: class: Title, method: remove, update
* remove or update item: class: Item, method: remove, update

For use cases (Actor: Checkout Clerk)

* add borrower: class: Borrower, method: add
* make reservation: class: Reservation, method: add
* lend item: class: Item, method: lend
* return item: class: Item, method: return
* remove or update borrower: class: Borrower, method: remove, update
* remove reservation: class: Reservation, method: remove

### 2. Using the detailed use cases and the TestScriptTemplate provided, provide detailed test scripts for the 2 use case details.

The main difference between a use case and a test case is that a use case describes what the system does, while a test case describes how the system will be tested.

A use case is a description of a specific interaction between a user and a system, detailing the steps that are taken to accomplish a specific task or goal. It outlines the steps that the system takes to fulfill the user's request and the expected outcome. A use case is typically written from the user's perspective and is used to describe the system's functionality and requirements.

A test case, on the other hand, is a document that describes the steps that will be taken to test a specific aspect of the system. It outlines the inputs that will be provided, the expected outcome, and the steps that will be taken to verify that the outcome is achieved. A test case is typically written from the testing perspective and is used to ensure that the system meets its requirements and works as intended.

### 3. To the detailed use stories provided, add BDD scenarios

It is important for the development team, including programmers, developers, and testers, to meet with the stakeholders, such as librarians and checkout clerks, to determine the scenarios for the system.

This is because the stakeholders are the ones who will be using the system daily and have the most knowledge about the specific needs and requirements of the library management system. By involving them in the process of determining the scenarios, the development team can ensure that the system meets the needs of the users and functions in a way that is intuitive and efficient for them. Additionally, it helps in understanding the requirement and test scenario that can be used to validate the system's functionality.

A Gherkin Scenario is a way of describing a specific feature or behavior of a software system using a natural language syntax. It is made up of three parts: the Feature, the Scenario, and the Steps.

The three parts of a Gherkin Scenario are:

* Given: This describes the initial state or context of the scenario. It sets the scene for the action that will take place.
* When: This describes the action or event that is being tested. It is a single action from a user's perspective.
* Then: This describes the expected outcome or result of the action described in the "When" step. It outlines what the user should see or experience after the action has taken place.

The "Given" step sets the stage for the scenario, the "When" step describes the action being taken, and the "Then" step describes the expected outcome of the action.

BDD Scenario: Adding a new title

* Given that the librarian is logged in to the system
* When the librarian enters the information for a new title and clicks the save button
* Then the system checks if the title is new or if the save should be cancelled and prompts the librarian to confirm the save before it becomes permanent.

BDD Scenario: Adding a new item

* Given that the librarian is logged in to the system and a title has already been added
* When the librarian enters the information for a new item and associates it with the previously added title
* Then the system checks if the title exists for the item to be added and the item is added to the library's inventory.

BDD Scenario: Adding a new borrower

* Given that the checkout clerk is logged in to the system
* When the checkout clerk enters the information for a new borrower and clicks the save button
* Then the system checks if the borrower is new or if the save should be cancelled and prompts the checkout clerk to confirm the save before it becomes permanent

BDD Scenario: Removing a reservation

* Given that the checkout clerk is logged in to the system and a reservation for an item exists
* When the checkout clerk selects the option to remove the reservation and confirms the action
* Then the system verifies that the reservation exists and removes it, making the item available for loan again

BDD Scenario: Returning an item

* Given that the checkout clerk is logged in to the system and one or more items have been checked out
* When the borrower returns the items to the library and the checkout clerk scans the items
* Then the system verifies that the items are eligible for return and updates the inventory to indicate the items are available for loan again.

It is important to have a meeting on testing the feature before development starts because it allows the team to clearly understand the requirements and expected outcomes of the feature. This way, they can ensure that they are building the feature correctly and that it will meet the needs of the users. It also allows the team to identify any potential issues or challenges that may arise during development, so they can plan and address them before starting the development process. This ultimately saves time and resources and helps to ensure that the final product is of high quality and meets the needs of the users.