Requirement guide for class project

Use cases will be used as requirements for this project.

Template

An adaptation of the standard Cockburn template will be used. The template and examples follow:

ID and name	Vizualizare detalii comanda		
Primary actor	Personalul medical	Secondary actors	
Description	Vizualizarea detaliilor unei comenzi realizate de un anumit peronal medical		
Trigger	Apasare butonul din pagina de start “Vizualizare detalii comenzi”		
Preconditions	Utilizatorul trebuie sa fie autentificat		
Postconditions	Modificarea sau trimiterea comenzii la farmacie cu mesajul “Comanda va fi trimisa la farmacie.O zi buna!” si salvarea in baza de date		
Normal flow	1.Utlizatorul face click pe “Vizualizare detalii comanda” 2.Softul afiseaza o lista cu medicamentele adaugate in comanda,cantitatea si starea lor 3.Utilizatorul verifica datele 4.Utilizatorul apasa pe butonul “Trimite comanda” 5.Softul afis mesajul “Comanda va fi trimisa la farmacie.O zi buna! si se intoarce la pagina de start si comanda este salvata in baza de date cu comenzi		
Alternative flows	3a.Utilizatorul verifica datele dar realizaza ca trebuie modificate 3a1.Selecteaza un medicament si apasa pe butonul de modificare 3a2.Modifica comanda si apoi apsa pe “Salveaza” 3a3.Softul o sa l intoarca la pagina anterioara		
Exceptions	Utilizatorul poate sa nu apese pe butonul de “Trimitere comanda” 4a.Utilizatorul verifica comanda si apoi se intoarce la “Comanda noua” unde mai poate adauga medicamente 4a1.Utlizatorul va trimite comanda din acea pagina		

Descriptions of template fields:

- **ID and name:** Title should be descriptive and should usually begin with a verb, e.g. order, calculate, input, etc. ID can have any format but must be unique among all use cases.
- **Primary actor:** Person that wishes to accomplish a goal through the use of the system. Only a single primary actor per use case.
- **Secondary actors:** Actors that have an interest in the completion of the goal but that do not directly interact with the system.
- **Description:** Concise description of the purpose of the use case.
- **Trigger:** Condition internal or external to the system that prompts the use case to start.
- **Preconditions:** Conditions that must be true before the use case starts. Each should be labeled with an ID unique to the use case.
- **Postconditions:** Conditions that must be true after the use case ends normally. Each should be labeled with an ID unique to the use case.
- **Normal flow:** Detailed step-by-step description of the logical flow of the use case. It should describe an explicit two way interaction, with the system prompting for input and the actor responding accordingly. Each step should be numbered.
- **Alternative flows:** Flows that achieve the same goal as the normal flow but are expected to be less common or lower priority.
- **Exceptions:** Conditions that result in the normal flow ending prematurely due to an unrecoverable condition in the system. The condition that causes the flow should be clearly stated, as should be any other decisions that the actor must make in this situation.

Examples

For a hypothetical *Cafeteria Ordering System*[1]:

ID and name	UC-1: Order a Meal		
Primary actor	Patron	**Secondary actors**	Cafeteria Inventory System
Description	A Patron accesses the Cafeteria Ordering System from either the corporate intranet or external Internet, views the menu for a specific date, selects food items, and places an order for a meal to be picked up in the cafeteria or delivered to a specified location within a specified 15-minute time window.		
Trigger	A Patron indicates that he wants to order a meal.		
Preconditions	PRE-1. Patron is logged into COS. PRE-2. Patron is registered for meal payments by payroll deduction.		

[1] Examples adapted from Wiegers, K. E. & Beatty, J. (2013) Software requirements . 3rd ed. Redmond, WA: Microsoft Press.

Postconditions	POST-1. Meal order is stored in COS with a status of “Accepted.” POST-2. Inventory of available food items is updated to reflect items in this order. POST-3. Remaining delivery capacity for the requested time window is updated.
Normal flow	**1.0 Order a Single Meal** 1. Patron asks to view menu for a specific date. (see 1.0.E1, 1.0.E2) 2. COS displays menu of available food items and the daily special. 3. Patron selects one or more food items from menu. (see 1.1) 4. Patron indicates that meal order is complete. (see 1.2) 5. COS displays ordered menu items, individual prices, and total price, including taxes and delivery charge. 6. Patron either confirms meal order (continue normal flow) or requests to modify meal order (return to step 2). 7. COS displays available delivery times for the delivery date. 8. Patron selects a delivery time and specifies the delivery location. 9. Patron specifies payment method. 10. COS confirms acceptance of the order. 11. COS sends Patron an email message confirming order details, price, and delivery instructions. 12. COS stores order, sends food item information to Cafeteria Inventory System, and updates available delivery times.
Alternative flows	**1.1 Order multiple identical meals** 1. Patron requests a specified number of identical meals. (see 1.1.E1) 2. Return to step 4 of normal flow. **1.2 Order multiple meals** 1. Patron asks to order another meal. 2. Return to step 1 of normal flow.
Exceptions	**1.0.E1 Requested date is today and current time is after today’s order cutoff time** 1. COS informs Patron that it’s too late to place an order for today. 2a. If Patron cancels the meal ordering process, then COS terminates use case.

	2b. Else if Patron requests another date, then COS restarts use case. **1.0.E2 No delivery times left** 1. COS informs Patron that no delivery times are available for the meal date. 2a. If Patron cancels the meal ordering process, then COS terminates use case. 2b. Else if Patron requests to pick the order up at the cafeteria, then continue with normal flow, but skip steps 7 and 8. **1.1.E1 Insufficient inventory to fulfill multiple meal order** 1. COS informs Patron of the maximum number of identical meals he can order, based on current available inventory. 2a. If Patron modifies number of meals ordered, then return to step 4 of normal flow. 2b. Else if Patron cancels the meal ordering process, then COS terminates use case.

ID and name	UC-5 Register for Payroll Deduction		
Primary actor	Patron	**Secondary actors**	Payroll System
Description	Cafeteria patrons who use the COS and have meals delivered must be registered for payroll deduction. For noncash purchases made through the COS, the cafeteria will issue a payment request to the Payroll System, which will deduct the meal costs from the next scheduled employee payday direct deposit.		
Trigger	Patron requests to register for payroll deduction, or Patron says yes when COS asks if he wants to register.		
Preconditions	PRE-1. Patron is logged into COS.		
Postconditions	POST-1. Patron is registered for payroll deduction.		
Normal flow	**5.0 Register for Payroll Deduction** 1. COS asks Payroll System if Patron is eligible to register for payroll deduction. 2. Payroll System confirms that Patron is eligible to register for		

	payroll deduction. 3. COS asks Patron to confirm his desire to register for payroll deduction. 4. If so, COS asks Payroll System to establish payroll deduction for Patron. 5. Payroll System confirms that payroll deduction is established. 6. COS informs Patron that payroll deduction is established.
Alternative flows	None
Exceptions	5.0.E1 Patron is not a full time employee. 5.0.E2 Patron is already enrolled for payroll deduction.

Extra credit step: Traceability

For this extra step, you will add traceability information for each use case by adding a new field to the template:

Method-level traces	<fully.qualified.ClassName>#<methodName> ...

Any method that implements the functionality described in the normal flow, alternative flow or exceptions should be included in this field. This means that the method that is initially executed and any methods of any classes that the work is delegated to should be included.

Examples for previous use cases:

UC-1:

Method-level traces	my.company.ordering.MenuWidget#dateClicked my.company.ordering.MenuWidget#completeOrder my.company.ordering.InventoryInterface#checkInventory ...

UC-5:

Method-level traces	my.company.payroll.PayrollInterface#checkEligibility my.company.payroll.RegistrationForm#confirm ...