

# 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	Consultare comenzi		
Primary actor	Farmacist	Secondary actors	
Description	Farmacistii se uita peste comenzile trimise de catre personalul medical		
Trigger	Autentificarea		
Preconditions	Autentificarea		
Postconditions	Onorarea comenzii cu stergerea din baza de date sau schimbarea starii comezii in nefinalizata		
Normal flow	1.Framacistul se autentifica 2.Softul deschide o pagina cu doua liste,una cu medicamentele care trebuie trimise in functie de sectii si cealalta cu lista tuturor medicamentelor si cantitatea lor de pe storc 3.Farmacistul se uita prima verifica starea medicamentelor apoi la data 4.Utilizatorul selecteaza prima data medicamentele cu starea "Urgenta" apoi verifica daca se afla pe stoc cautand in cea de a doua lista 5.Utilizatorul apasa apoi pe butonul "Onoreaza comanda" 6.Softul va trimite "Comanda va fi trimisa pe sectie" 7.Medicamentele din lista de comezii se vor sterge automat din lista si din baza de date		
Alternative flows	4a.Utilizatorul dupa verificare starii si a datei observa ca comanda nu se afla in stoc 4a1.Utilizatorul schimba statusul medicamentului in "Nu este in stoc" apsand pe butonul "Nefinalizare comanda"		
Exceptions	Utilizatorul nu observa ca medicamentul nu este in stoc 6a.Dupa apasarea butonul de "Onoreaza comanda" se va afisa mesajul "Medicamentul nu este in stoc"		

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*<sup>1</sup>:

<b>ID and name</b>	UC-1: Order a Meal		
<b>Primary actor</b>	Patron	<b>Secondary actors</b>	Cafeteria Inventory System
<b>Description</b>	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.		
<b>Trigger</b>	A Patron indicates that he wants to order a meal.		

---

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

<b>Preconditions</b>	<p>PRE-1. Patron is logged into COS.</p> <p>PRE-2. Patron is registered for meal payments by payroll deduction.</p>
<b>Postconditions</b>	<p>POST-1. Meal order is stored in COS with a status of "Accepted."</p> <p>POST-2. Inventory of available food items is updated to reflect items in this order.</p> <p>POST-3. Remaining delivery capacity for the requested time window is updated.</p>
<b>Normal flow</b>	<p><b>1.0 Order a Single Meal</b></p> <ol style="list-style-type: none"> <li>1. Patron asks to view menu for a specific date. (see 1.0.E1, 1.0.E2)</li> <li>2. COS displays menu of available food items and the daily special.</li> <li>3. Patron selects one or more food items from menu. (see 1.1)</li> <li>4. Patron indicates that meal order is complete. (see 1.2)</li> <li>5. COS displays ordered menu items, individual prices, and total price, including taxes and delivery charge.</li> <li>6. Patron either confirms meal order (continue normal flow) or requests to modify meal order (return to step 2).</li> <li>7. COS displays available delivery times for the delivery date.</li> <li>8. Patron selects a delivery time and specifies the delivery location.</li> <li>9. Patron specifies payment method.</li> <li>10. COS confirms acceptance of the order.</li> <li>11. COS sends Patron an email message confirming order details, price, and delivery instructions.</li> <li>12. COS stores order, sends food item information to Cafeteria Inventory System, and updates available delivery times.</li> </ol>
<b>Alternative flows</b>	<p><b>1.1 Order multiple identical meals</b></p> <ol style="list-style-type: none"> <li>1. Patron requests a specified number of identical meals. (see 1.1.E1)</li> <li>2. Return to step 4 of normal flow.</li> </ol> <p><b>1.2 Order multiple meals</b></p> <ol style="list-style-type: none"> <li>1. Patron asks to order another meal.</li> <li>2. Return to step 1 of normal flow.</li> </ol>
<b>Exceptions</b>	<p><b>1.0.E1 Requested date is today and current time is after today's order cutoff time</b></p> <ol style="list-style-type: none"> <li>1. COS informs Patron that it's too late to place an order for today.</li> </ol>

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

<b>ID and name</b>	UC-5 Register for Payroll Deduction		
<b>Primary actor</b>	Patron	<b>Secondary actors</b>	Payroll System
<b>Description</b>	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.		
<b>Trigger</b>	Patron requests to register for payroll deduction, or Patron says yes when COS asks if he wants to register.		
<b>Preconditions</b>	PRE-1. Patron is logged into COS.		
<b>Postconditions</b>	POST-1. Patron is registered for payroll deduction.		

<b>Normal flow</b>	<b>5.0 Register for Payroll Deduction</b> <ol style="list-style-type: none"> <li>1. COS asks Payroll System if Patron is eligible to register for payroll deduction.</li> <li>2. Payroll System confirms that Patron is eligible to register for payroll deduction.</li> <li>3. COS asks Patron to confirm his desire to register for payroll deduction.</li> <li>4. If so, COS asks Payroll System to establish payroll deduction for Patron.</li> <li>5. Payroll System confirms that payroll deduction is established.</li> <li>6. COS informs Patron that payroll deduction is established.</li> </ol>
<b>Alternative flows</b>	None
<b>Exceptions</b>	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 ...
---------------------	--

