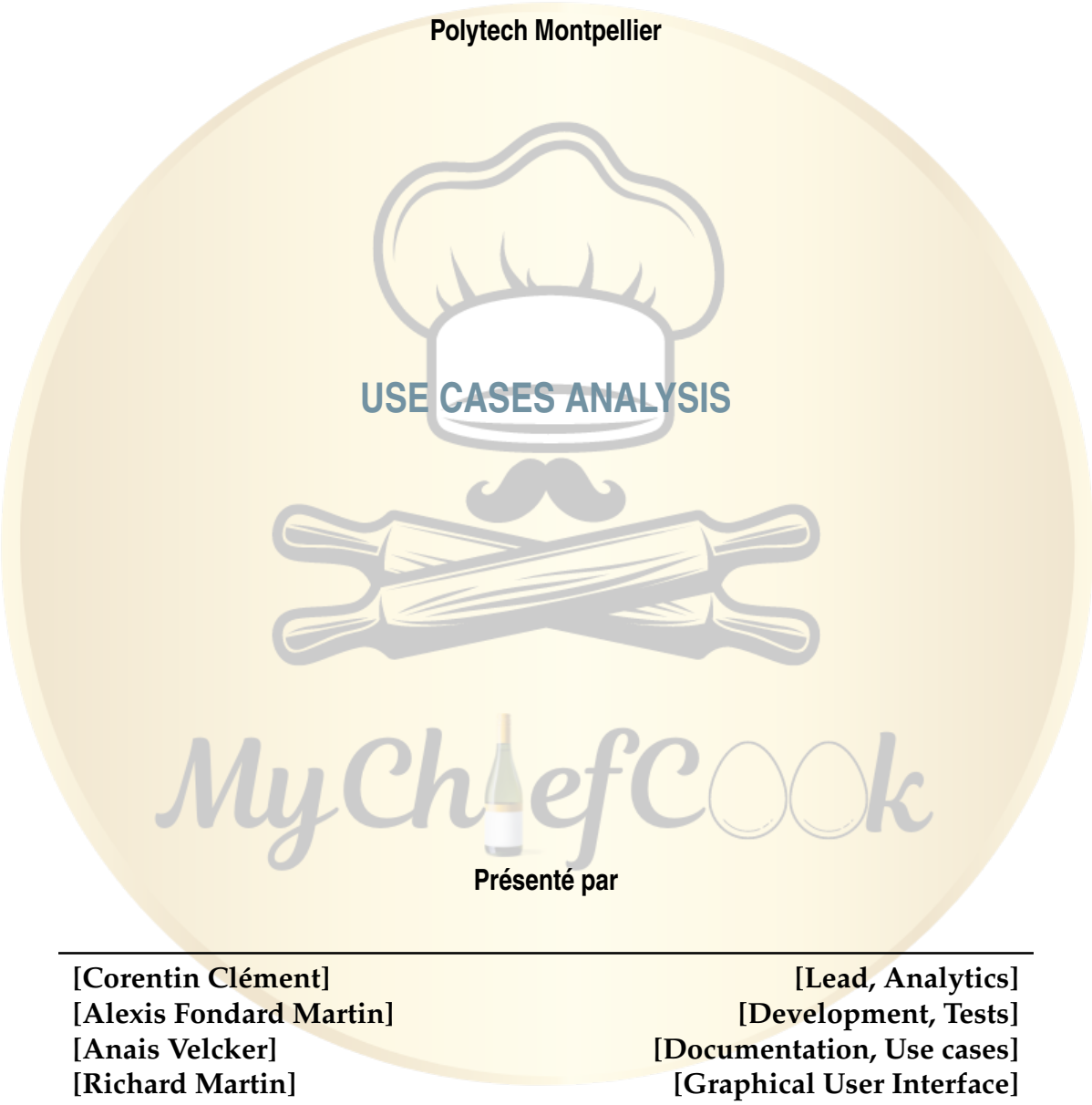
My Chief Cook

Informatique et Gestion

Polytech Montpellier



Présenté par

[Corentin Clément]	[Lead, Analytics]
[Alexis Fondard Martin]	[Development, Tests]
[Anais Velcker]	[Documentation, Use cases]
[Richard Martin]	[Graphical User Interface]
[Lucas Nouguier]	[Database, Reports]

POLYTECH MONTPELLIER

UNIVERSITÉ DE MONTPELLIER

Contents

1 Problem Statement

1.1 Context

You have been tasked by a start-up with building a proof of concept of their new cooking application. They need a prototype to show to investors and potential customers. The application will be used by Administrators to manage their recipes (and their inventory). The application will also be used by customers to search for recipes and eventually wines to come with their meal.

1.2 Problem

The final system will be an interactive web app, but for the proof of concept, you will be building a desktop application capable of running on any devices. It will allow Administrators to expose their recipes and partners while users will be able to order meals and wines. The application will also be able to suggest recipes based on the ingredients available in the user's fridge. Once they've made their choices, they must be able to pass command to either the restaurants, either to partners if they want to buy ingredients. They should as well receive wine recommandations if they chose to. Users will launch this prototype locally on their computer. For security and consistency reasons, some features will require some permissions (such as moderation).

For a more pleasant user experience, we'll need users to be able to create "favorite lists" which must be name and where they can add recipes to easily find later. They must have full ownership of these lists. There will also be a feature to allow users to manage their meal plan up until 3 weeks ahead. They will be able to choose recipes and place them as a midday or evening meal. From the calendar users must be able to see the ingredient shopping list and the calendar should be persistent and auto-update according to the current date (for instance by strike-through past meals).

Users can send feedbacks too, by rating and commenting recipes. To promote helpful comments, there will a be voting system for comments (upvote/downvote) and comments with a high score (lots of upvotes) will be among the first comment in the list, whereas comments with low score (lots of downvotes) will be among the last. In addition, They should be able to suggest new recipes to Administrators by giving a description of what they think about (and eventually a name). Administrators will then be able to accept or reject the suggestion.

Administrators should be able to add to announcements to their page and manage them (update, hide, delete…). An announcement may have some labels to facilitate the search at a latter point

1.3 Constaints

Users should not be forced to expose their private life, so the system will retain as few information as possible :

- Mandatory fields : their username, email address, password, secret question
- Optional fields : name, address, gender, birthday, phone

Yet, Administrators must provide each fields for transparency. Also, some features will required optional fields to be fulfilled, so user must be warned when some fields are missing.

1.4 Core Features

- Chose a recipe from ingredients and eventually auto-ajust proportions to match
  - the number of people
  - the number of meals
  - the current quantity of ingredients in the fridge
- Optionally add wine to the meal
- Make a meal plan for at most the next 3 weeks
- Redirect to partners' website to buy ingredients and/or wines

2 Use Cases Model

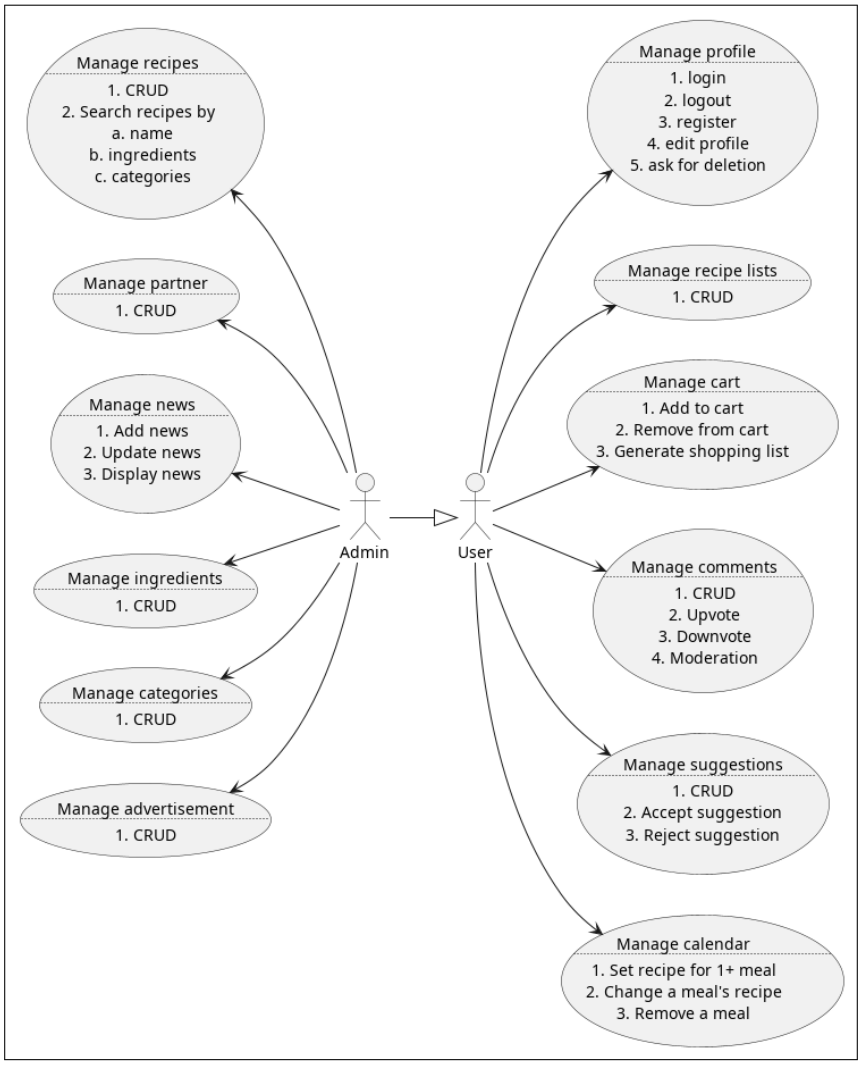


Figure 1: *Use Cases Model*

3 Use Cases Estimated Complexity

Manage profile *Assignee : Anaïs V.* *Complexity : 5/5*	**Manage recipe lists** *Assignee : Richard M.* *Complexity : 2/5*	**Manage cart** *Assignee : Richard M.* *Complexity : 3/5*
Manage recipes *Assignee : Lucas N.* *Complexity : 4/5*	**Manage ingredients** *Assignee : Lucas N.* *Complexity : 2/5*	**Manage categories** *Assignee : Corentin C.* *Complexity : 1/5*
Manage comments *Assignee : Alexis F.* *Complexity : 3/5*	**Manage suggestions** *Assignee : Corentin C.* *Complexity : 3/5*	**Manage calendar** *Assignee : Alexis F.* *Complexity : 3/5*
Manage news *Assignee : Anaïs V.* *Complexity : 2/5*	**Manage partners** *Assignee : Richard M.* *Complexity : 1/5*	**Manage advertisement** *Assignee : Corentin C.* *Complexity : 2/5*

4 Use Cases Description

4.1 Ingredient management

4.1.1 Brief description

Ingredients are the base of recipes. As for recipes, they can be modified (created, updated, deleted) by Administrators only. They will be able to specify a name, a list of tags (categories) and wether it's allergenic or not. If it's allergenic, it must be emphasized. Name, and allergen status are mandatory fields.

Administrators should be able to search ingredients by name and the result should be sorted by name by default. They should be able to sort by tags, and allergenic status. Results should be paginated and the number of results per page should be configurable. They should also be able to filter by tags and allergenic status.

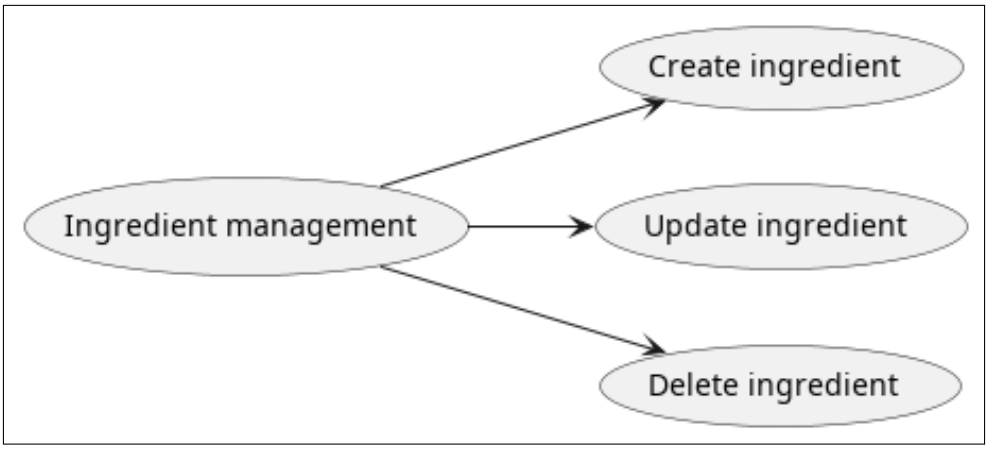


Figure 2: *Ingredient Management*

4.1.2 Flow of events

This Use Case starts whenever a Administrator wants to add, update or delete an ingredient to the database

Basic Flows

- Create an ingredient : The system asks the Administrator to enter the information about the ingredient : name, tags (categories), allergen
    1. Once the Administrator provided at least the required information, the system adds the ingredient to the database.
    2. The system provide a success message, returns the newly created ingredient, and goes back to ingredient management menu.
- Update an ingredient : The system ask the Administrator to enter the new information about the ingredient
    1. Once the Administrator provided at least the required information, the system updates the ingredient in the database.
    2. The system provide a success message, updates the ingredient on the application, and goes back to ingredient management menu.
- Delete an ingredient : The system ask the Administrator which ingredient he wants to delete
    1. Once the Administrator chosed the ingredient, the system checks if it is used by any recipe. If it is not, the system deletes the ingredient from the database.
    2. The system provide a success message, deletes the ingredient on the application, and goes back to ingredient management menu.

Alternative Flows

- Create an ingredient :
    1. If the Administrator does not provide every mandatory fields and tries to validate, the system display an error message as a "toast" and does nothing else.
    2. If the Administrator cancels, the system goes back to ingredient management menu.
- Update an ingredient :
    1. If the Administrator does not provide every mandatory fields and tries to validate, the system display an error message as a "toast" and does nothing else.
    2. If the Administrator tries to update an ingredient already use by a recipe, the system display a warning message and ask for confirmation. If the Administrator confirms, the system updates the ingredient in the database and in the app and the normal flow continues. If the Administrator does not confirm, the system does nothing else.
    3. If the Administrator cancels, the system goes back to ingredient management menu.

- Delete an ingredient :
  1. If the Administrator tries to delete an ingredient already use by a recipe, the system display a error message. Otherwise, the system deletes the ingredient in the database and in the app and the normal flow continues.
  2. If the Administrator cancels, the system goes back to ingredient management menu.

4.1.3 Special requirements

Must be thread-safe in case multiple Administrators are modifying the same ingredient.

4.1.4 Preconditions

The Administrator must be logged onto the system before this use case begins.

4.1.5 Postconditions

If it's successful, the ingredient will be added/updated/deleted from the database. Otherwise, the state remains unchanged.

4.1.6 Extension points

None

4.2 Recipe management

4.2.1 Brief description

Recipes can be modified (created, updated, deleted) by Administrators only. They will be able to specify a name, a summary, a list of ingredients, some categories, a description, and maybe a picture of it. Recipes can be search by name, tags, ingredients, whether it contains allergens or not, and may be sorted by name or popularity.

If a recipe contains allergens, it must be specified and allergenic ingredients must appear in emphasized text. A recipe can also receive many comments and ratings from the users.

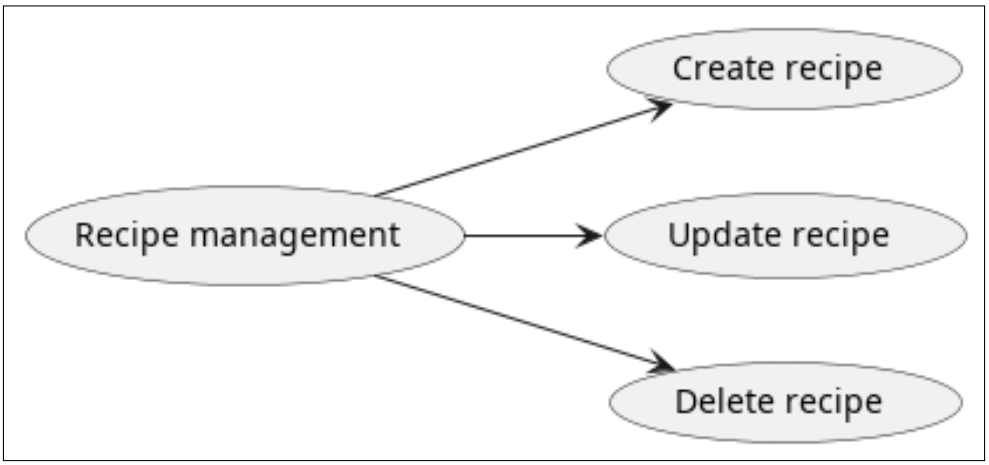


Figure 3: *Recipe Management*

4.2.2 Flow of events

This Use Case starts when a Administrator wants to add, update or delete a recipe to the database.

Basic Flows

- Create a recipe : The system asks the Administrator to enter the information about the recipe : name, summary, tags (categories), ingredients, description, picture
    1. Once the Administrator provided at least the required information, the system adds the recipe to the database.
    2. The system provide a success message, returns the newly created recipe, and goes back to recipe management menu.
- Update a recipe : The system ask the Administrator to enter the new information about the recipe
    1. Once the Administrator provided at least the required information, the system updates the recipe in the database.
    2. The system provide a success message, updates the recipe on the application, and goes back to recipe management menu.
- Delete a recipe : The system ask the Administrator which recipe he wants to delete
    1. Once the Administrator chosed the recipe, the system deletes the recipe from the database.
    2. The system provide a success message, deletes the recipe on the application, and goes back to recipe management menu.

Alternative Flows

- Create a recipe :
    1. If the Administrator does not provide all the mandatory information and tries to validate, the system display an error message as a "toast" and does nothing else.
    2. If the Administrator cancels, the system goes back to recipe management menu.
- Update a recipe :
    1. If the Administrator does not provide all the mandatory information and tries to validate, the system display an error message as a "toast" and does nothing else.
    2. If the Administrator cancels, the system goes back to recipe management menu.
- Delete a recipe :
    1. If the Administrator cancels, the system goes back to recipe management menu.

4.2.3 Special requirements

Must be thread-safe in case multiple Administrators are modifying the same recipe.

4.2.4 Preconditions

The Administrator must be logged onto the system before this use case begins.

4.2.5 Postconditions

If it's successful, the recipe will be added/updated/deleted from the database. Otherwise, the state remains unchanged.

4.2.6 Extension points

None

4.3 Category Management

4.3.1 Brief description

A recipe belongs to one or more categories. The goal is to give simple information about the recipe (for example, the recipe is a "vegetarian" "dish"), but also to be able to apply filters on the recipe search.

The different categories can be created, modified and deleted only by administrators connected to their account.

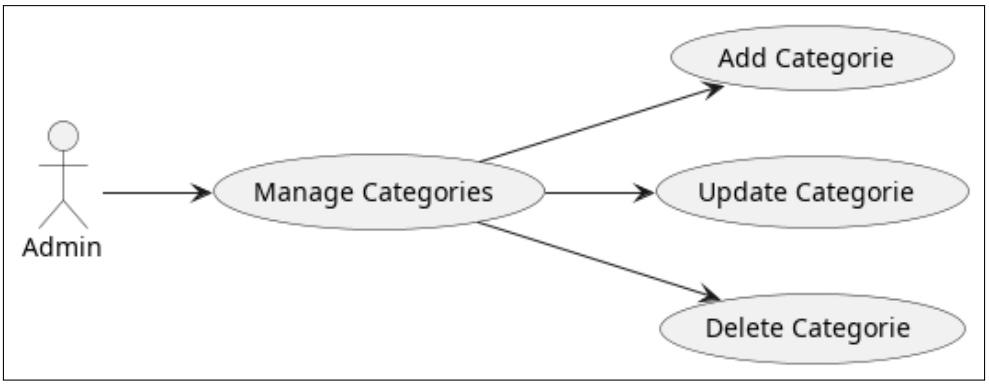


Figure 4: *Category Management*

4.3.2 Flow of events

The system requests that the Administrator specify the function he/she would like to perform : either Add a Category, Update a Category, or Delete a Category (see Figure 4). Once the Administrator provides the requested information, one of the sub flow is executed.

Basic Flows

- If the Administrator selected "Add a Category", the Add a Category sub flow is executed :
    1. The system request that the administrator enter new information. This include the name's category.
    2. the system generates a id number for the new category, register it in the database, the category is available for recipes and the administrator is informed that the creation worked with the message "The category has been created".
- If the Administrator selected "Update a Category", the Update a Category sub flow is executed :
    1. The administrator selects a category from the list of categories (obtained from the system). The administrator can only change the name's category.
    2. The system update the category's name in the database. The administrator is informed that the update worked with the message "The category has been updated".
- If the Administrator selected "Delete a Category", the Delete a Category sub flow is executed :

1. The administrator selects a category from the list of category (obtained from the system). The administrator can delete the chosen category and need to confirm to delete.
2. The system update the database therefore deletes the category. The administrator is informed that the delete has worked with the message "The category has been deleting".

Alternate Flows

- Add Category or Update category with an existing name : the administrator can't create a category or update a category with an existing name. In this case, the state will not change and the administrator is notified that he is not allowed to do the change/creation with a message "This name is already use".
- Action cancelling : If the Administrator decides to cancel the current action, then the action aborts and the system is unchanged.

4.3.3 Special requirements

None

4.3.4 Preconditions

The Administrator must be logged onto the system before this use case begins.

4.3.5 Postconditions

If the use case was successful, the category information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

4.3.6 Extension points

None

4.4 Ad management

4.4.1 Brief description

We have advertisements in the application, to suggest to buy some ingredients with a promotion with our partners. So we need to manage the ads : create an ad, update an ad and delete an ad.

The different adverts can be created, modified and deleted only by administrators connected to their account.

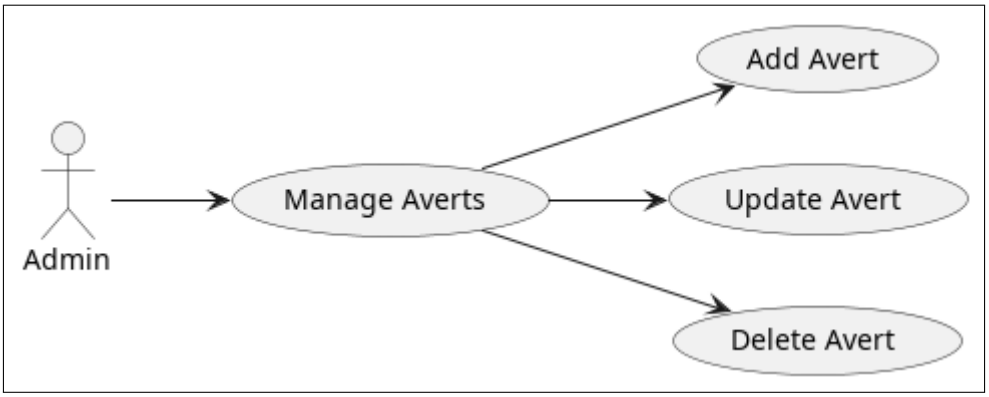


Figure 5: *Ad Management*

4.4.2 Flow of events

The system requests that the Administrator specify the function he/she would like to perform : Add an Advert, Update an Advert, or Delete an Advert (see Figure 5). Once the Administrator provides the requested information, one of the sub flows is executed.

Basic Flows

- If the Administrator selected "Add an Advert", the Add an Advert sub flow is executed :
  1. The system request that the administrator enter new information. This include the partner, the product, the promotion, the price.
  2. The system generates a id number for the new advert, register it in the database and the ad is available for the ingredient. The administrator is informed that the creation worked with the message "The advert has been created".
- If the Administrator selected "Update a Category", the Update an Advert sub flow is executed :
  1. The administrator selects an ad from the list of adverts (obtained from the system). The administrator can change the partner, the product, the promotion, the price.
  2. The system updates the data in the database and the administrator is informed that the update worked with the message "The advert has been updated".
- If the Administrator selected "Delete a Category", the Delete an Advert sub flow is executed :
  1. The administrator selects an ad from the list of adverts (obtained from the system). The administrator can delete the chosen advert and need to confirm to delete.
  2. The system updates the database therefore deletes the ad. The administrator is informed that the delete worked with the message "The advert has been deleted".

Alternate Flows

- Action cancelling : If the Administrator decides to cancel the current action, then the action aborts and the system is unchanged.

4.4.3 Special requirements

None

4.4.4 Preconditions

The Administrator must be logged onto the system before this use case begins.

4.4.5 Postconditions

If the use case was successful, the advertising information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

4.4.6 Extension points

None

4.5 Suggestion Management

4.5.1 Brief description

When the customer uses the application, he can sometimes be disappointed because he can't find a recipe or an ingredient or want to suggest a new feature for the application. The manage suggestions is the CRUD to this. The users will create suggestions and the admins will read, update and delete or implement the suggestion.

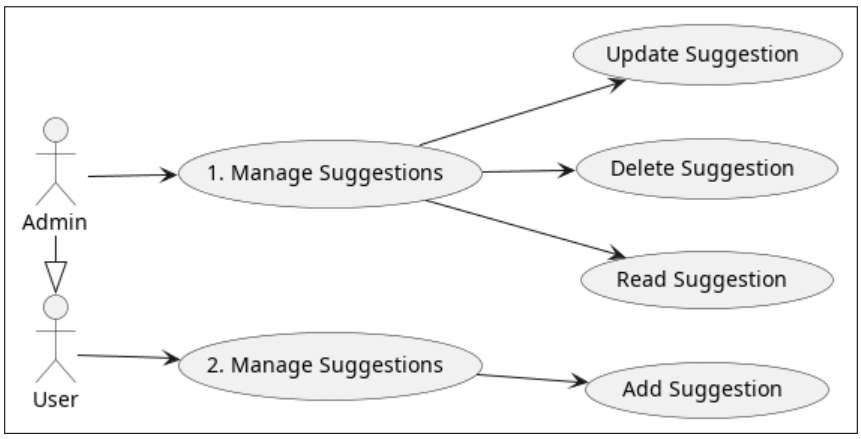


Figure 6: *Suggestion Management*

4.5.2 Flow of events

The functionality is available through a button placed in different places in the application. The button to access the suggestion form page is visible in the drop down menu and when searching for an ingredient or recipe the button is directly available on the page if no results match the user's search.

Basic Flows

- If the Payroll User selected "Propose a Suggestion", the Add a Suggestion sub flow is executed :
    1. The customer enters the information on the system. This includes the category of the suggestion (ingredient, recipe, feature) and a text field to detail the request (the name of the ingredient or the new feature requested...).
    2. The system generates a id number for the new suggestion, register it in the database and the suggestion is available for admins. The customer is informed that the creation worked with the message "The suggestion has been sent".
- If the Administrator selected "Update a Suggestion", the Update a Suggestion sub flow is executed :
    1. The administrator selects a suggestion from the list of suggestion (obtained from the system). The admin can change the suggestion, that is change the text field and the category of the suggestion if the customer is mistaken.
    2. The system updates changes in the database. The administrator is informed that the update worked with the message "The suggestion has been updated".

- If the Administrator selected “Delete a Suggestion“, the Delete a Suggestion sub flow is executed :
    1. The administrator selects a suggestion from the list of suggestion (obtained from the system). The administrator can delete the chosen suggestion and need to confirm to delete.
    2. The system updates the database therefore deletes the ad. The administrator is informed that the delete worked with the message “The advert has been deleted”.

Alternate Flows

- Action cancelling : If the Administrator decides to cancel the current action, then the action aborts and the system is unchanged.

4.5.3 Special requirements

None

4.5.4 Preconditions

The Administrator must be logged onto the system before this use case begins.

4.5.5 Postconditions

If the use case was successful, the advert information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

4.5.6 Extension points

None

4.6 Comment management

4.6.1 Brief description

This use case describes how a User can see comments and rates, post a comment under a recipe and/or rate a recipe. An Admin can also manage all comments (a kind of moderation, he can delete a comment if he thinks there is an abuse).

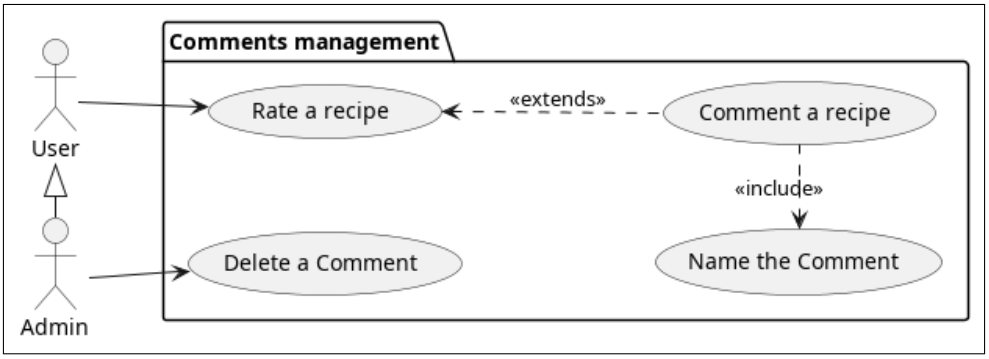


Figure 7: *Comment management*

4.6.2 Flow of events

This use case starts when the User wishes to comment or rate a recipe.

- The User opens a recipe and scrolls down to the Comment section below.
- There he can see the average rate and read all comments, that contain each a rate of the recipe.
- If the User clicks on the stars button (to rate the recipe), the Add Comment sub flow is executed.

This use case can also start when the Admin is on his moderation section.

- The Admin opens the list of comments, or goes to a recipe and scrolls down to the Comment section.
- If the Admin clicks on the Delete Comment button, the Delete Comment sub flow is executed.

Basic Flows

- Add Comment :
  1. The User has clicked on the stars button (he selected a note between 1-5 stars), then he can possibly fill the form bellow. This form includes:
     - Name of the comment (optional)
     - Content of the comment (optional but need a name)
  2. The User needs to click on the Submit button to send the comment.
- Delete comment :
  1. The Admin clicks on the Delete Comment button. A new popup appears asking the Admin if he is sure of his decision.
  2. The Admin clicks on the Validate button, then the selectionned button is deleted.

Alternative Flows

- Delete canceled : if in the Delete Comment sub flow the Admin decides not to delete the comment, the delete is canceled and the Basic Flow is restarted at the beginning.

4.6.3 Special requirements

None

4.6.4 Preconditions

The User/Admin needs to be logged in and a recipe needs to be created before this use case begins.

4.6.5 Postconditions

If the use case was successful, the comment is added/modified and the other users can now see the new/modified comment. If the Admin has removed the comment, it's now no longer visible.

4.6.6 Extension points

None

4.7 Calendar management

4.7.1 Brief description

This use case describes how a User can save his recipes in a calendar and plan when to make them. He can see his calendar whenever he needs to, and delete a recipe from the calendar if he needs to.

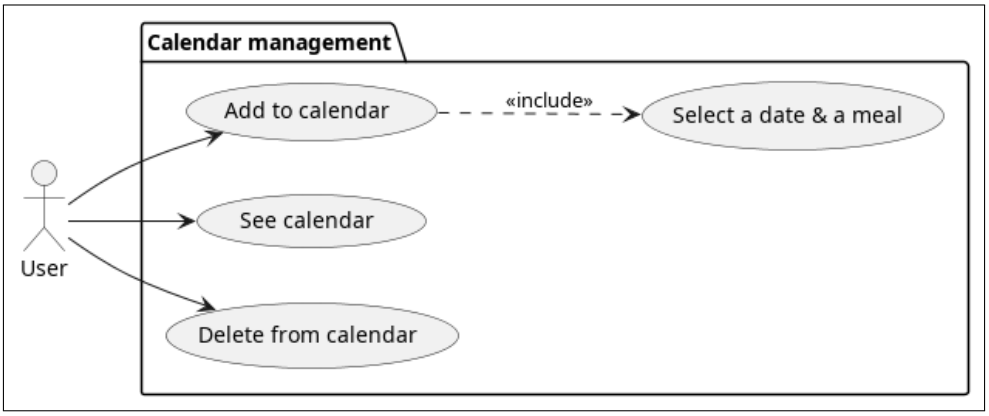


Figure 8: *Calendar management*

4.7.2 Flow of events

This use case starts when the User wishes to add a recipe to his calendar or see it :

1. The User opens a recipe or the list of recipes.
2. If the User clicks on the Add to Calendar button (which can be seen on a recipe or on each recipe of the list), the Add to Calendar sub flow is executed.
3. If the User clicks on the calendar button/icon, he can see his calendar with all his recipes saved in it.
4. If the User clicks on the cross (X) of a recipe saved in his calendar, the Delete from Calendar sub flow is executed.

Basic Flows

- Add to calendar :
  1. The User clicks on the Add to Calendar button, then a calendar popup is shown and ask for the User to select a date:
     - Year
     - Month
     - Day
     - Which meal: breakfast, lunch, snack, dinner, other
  2. The User needs to click on the Validate button to save the recipe to this date.
- Delete from calendar : the User has clicked on the cross (X) of a recipe saved in his calendar, a Validate popup is shown to check if he really needs to delete the recipe saved for this day.

Alternative Flows

- Add canceled : if in the Add to Calendar sub flow the User decides not to add the recipe, the saving is canceled and the Basic Flow is restarted at the beginning.
- Delete canceled : if in the Delete from Calendar sub flow the User decides not to delete the recipe, the delete is canceled and the Basic Flow is restarted at the beginning.

4.7.3 Special requirements

None

4.7.4 Preconditions

The User needs to be logged and a recipe needs to be created before this use case begins.

4.7.5 Postconditions

If the use case was successful, the recipe is added to the date in the calendar and the User can now see it in his calendar. Otherwise, the system state is unchanged.

4.7.6 Extension points

None

4.8 Recipe List

4.8.1 Brief description

If the user whises to save a recipe, he can use the recipe lists. He can have multiple lists to better sort them (for instance seasonal recipes, or recipes for a specific occasion). There will be a default list named "Favorites". One user can't have two lists with the same name.

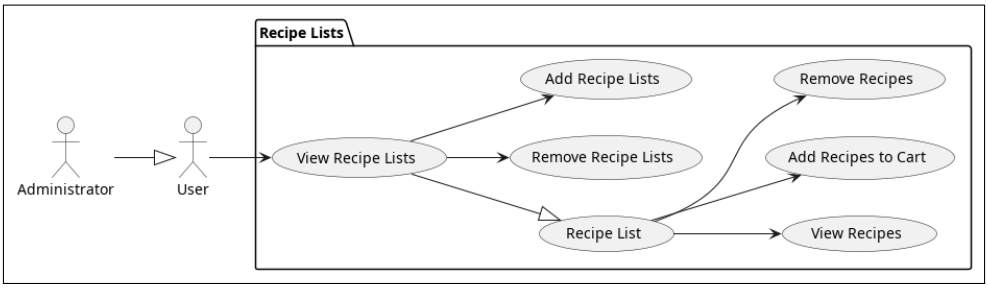


Figure 9: *Recipe list management*

4.8.2 Flow of events

This use case begins when the user wants to access to his lists, either to modify them or read them

- If the user clicks on the "create a list", then the Create list sub flow is executed
- If the user clicks on a list, then the Display recipes sub flow is executed
- If the user clicks on the "delete" button of a list, then the Delete list sub flow is executed
- If the user clicks on the "rename" button of a list, then the Rename list sub flow is executed
- If the user clicks on the "add to list" button of a recipe, then the Add to list sub flow is executed

- If the user clicks on the "remove from list" button of a recipe, then the Remove from list sub flow is executed

Basic Flows

- Create lists :
  1. The system ask the user to enter a name for the new list
  2. The system creates the list
  3. The system returns the newly created list to the user and displays it
- Display recipes : the system displays the recipes of the list. It may be displayed as a list or as a grid.
- Delete lists :
  1. The system asks the user to confirm the deletion
  2. The system deletes the list and removes it from the application
- Rename list :
  1. The system asks the user to enter a new name for the list
  2. The system renames the list in the database and in the application
- Add to list :
  1. The system asks the user to select the list to which he wants to add the recipe
  2. The system adds the recipe to the list
- Remove from list : the system removes the recipe from the list

Alternative Flows

- Remove recipe lists : if the user tries to delete a list that is not empty, the system asks him to confirm the deletion
- Create recipe list : if the user tries to create a list with a name that already exists, the system asks him to enter a new name
- Updating a list : if the user tries to update a list that doesn't exist, the system warns him with a popup

4.8.3 Special requirements

None

4.8.4 Preconditions

The user must be logged in to access to his lists.

4.8.5 Postconditions

If it's successful, the lists will be added/updated from the database. Otherwise, the state remains unchanged.

4.8.6 Extension points

None

4.9 Cart management

4.9.1 Brief description

The cart is a list of the recipe and ingredients an user ordered. There is only one cart by user and it cannot be deleted, only emptied. On the main page of the cart we'll be able to see the list of ordered recipes and ingredients. "Standalone" ingredients will appear separately from recipes' ingredients.

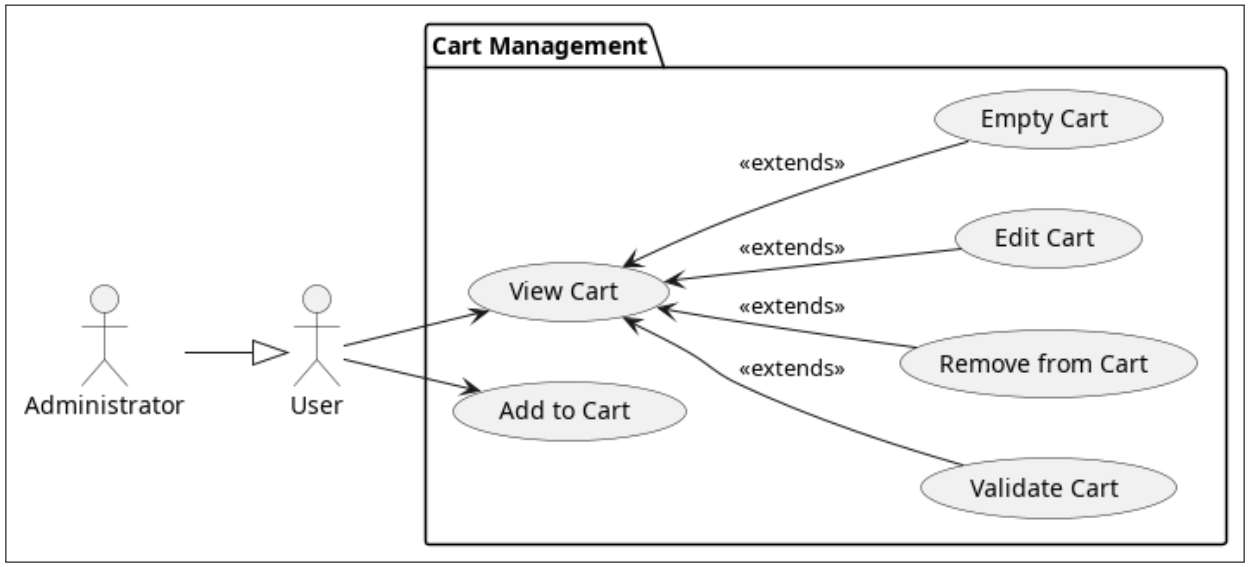


Figure 10: *Cart management*

4.9.2 Flow of events

This use case begins when a user wants to either add an element to his cart, see his cart or empty it.

- If the user clicks on the "add to cart" button of a recipe or ingredient, then the Add element to cart sub flow is executed
- If the user clicks on the "remove from cart" button of a recipe or ingredient, then the Remove element from cart sub flow is executed
- If the user clicks on the "change quantity" button of a recipe or ingredient, then the Change quantity sub flow is executed
- If the user clicks on the "empty cart" button, then the Empty cart sub flow is executed
- If the user clicks on the "order" button, then the Order sub flow is executed

Basic Flows

- Add element to cart :
  1. The system asks the user for the quantity he wants
  2. The system adds the element to the cart
- Remove element from cart : the system removes the element from the cart

- Change quantity :
  1. The system asks the user for the new quantity
  2. The system updates the quantity of the element in the cart
- Empty cart :
  1. The system asks the user to confirm the emptying
  2. The system empties the cart
- Order :
  1. The system asks the user to confirm the order. It displays the list of ordered recipes and ingredients and the lists of total quantities of ingredients needed.
  2. The system ask for the delivery address, phone and payement method. If the user has already entered this information, the system autocomplete the form.
  3. Once the user has provided the requested information, the system sends the order. For this proof of concept, it will juste be a message with the order information.
  4. The cart is emptied

Alternative Flows

- Add element to cart :
  1. The user wants to add an element that is already in his cart
  2. The system displays a message to inform him that the element is already in his cart, and launch the "Change quantity" sub flow.
- Empty cart : if the user can't clear his cart, for unknown reasons, a message is displayed to inform him that the cart can't be cleared.

4.9.3 Special requirements

None

4.9.4 Preconditions

The user must be logged in to access to his cart.

4.9.5 Postconditions

If it's successful, the cart will be updated in the database. Otherwise, the state remains unchanged.

4.9.6 Extension points

None

4.10 Partner management

4.10.1 Brief description

MyChiefCook has a list of Partner companies that are displayed to the users. They may offer discounts or special offers to the users. Users can access to the list of partners and their offers.

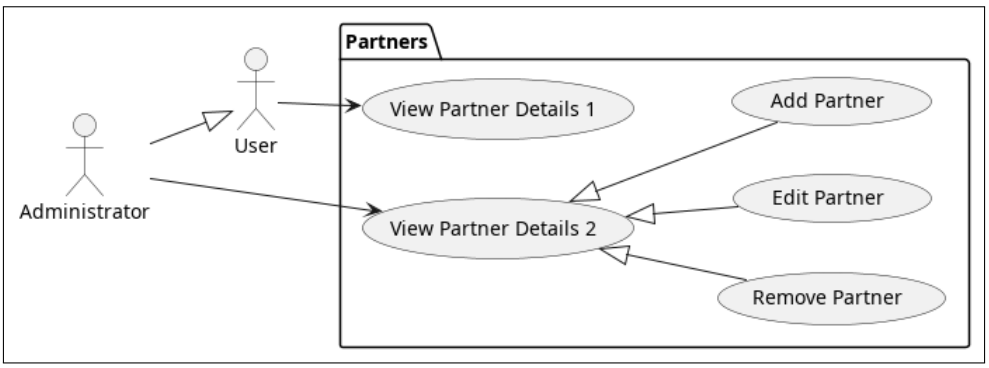


Figure 11: *Partner management*

4.10.2 Flow of events

This use case begins when an admin wants to manage the list of partners.

- If the admin clicks on the "add partner" button, then the Add partner sub flow is executed
- If the admin clicks on the "delete partner" button, then the Delete partner sub flow is executed
- If the admin clicks on the "update partner" button, then the Update partner sub flow is executed

Basic Flows

- Add partner :
  1. The system asks the admin to enter the information about a partner (name, description, website) and their offers (description, discount)
  2. The system adds the partner to the database
- Edit partner :
  1. The system asks the admin to enter the new information about the partner
  2. The system updates the partner in the database
- Delete partner :
  1. The system asks the admin to confirm the deletion
  2. The system deletes the partner from the database

Alternative Flows

- Add/Edit partner : if an admin tries to add a partner's offer on an ingredient on which there is already a discount, they system asks him to confirm the deletion of the previous offer and replace it with the new one.
- Operation canceled : if the admin cancels the operation, the modification are not saved and the admin is redirected to the partner management page.

4.10.3 Special requirements

None

4.10.4 Preconditions

The user must be logged in and be an admin

4.10.5 Postconditions

If it's successful, the partner will be added/updated from the database. Otherwise, the state remains unchanged.

4.10.6 Extension points

None

4.11 News management

4.11.1 Brief description

The news will show the user what is new (change in the application, new recipes, new partner). It can be modified (created, updated) by administrators only. They will be able to specify a title, a description, and a picture if they want. They also will be able to display or not the news to the users.

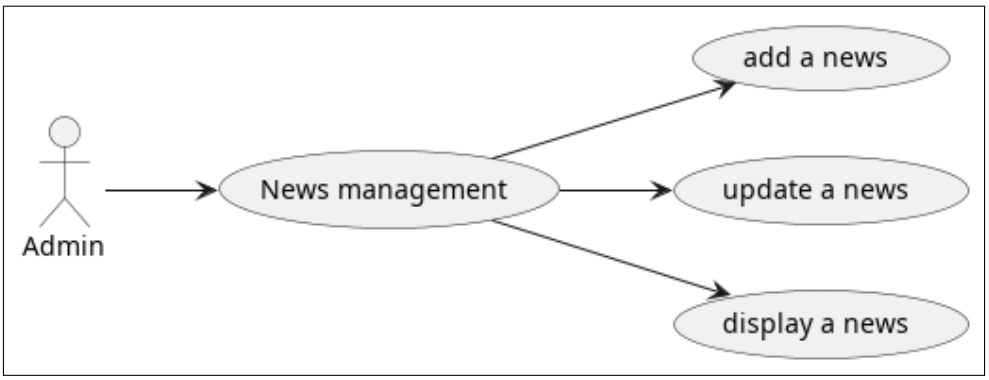


Figure 12: *News management*

4.11.2 Flow of events

This use case starts when the administrator wishes to add, change, and/or display a news from the system.

The system requests that the administrator specify the function he/she would like to perform (either add a news, update a news or display a news). Once the administrator has provided the requested information, one of the sub flows is executed.

Basic Flows

- If the administrator selected "Add news", the Add news sub flow is executed :
  1. The system requests that the administrator enter the news information. This includes : title, description, pictures (optional)
  2. Once the administrator has provided the requested information, the system generates and assigns a unique news id to the new news. The news is added to the system and the news display is set to "not displayed" by default.
- If the administrator selected "Update news", the Update news sub flow is executed :

1. The system requests that the administrator makes the desired changes to the news information. This includes any of the information specified in the Add a news sub-flow.
2. Once the administrator has updated the necessary information, the system updates the news record with the updated information.

- If the administrator selected "Display news", the Display news sub flow is executed : the system requests that the administrator selects the option he wants. He can display the news to the users or hide it.

4.11.3 Special requirements

Must be thread-safe in case multiple administrators are modifying the news.

4.11.4 Preconditions

The administrator must be logged onto the system before this use case begins.

4.11.5 Postconditions

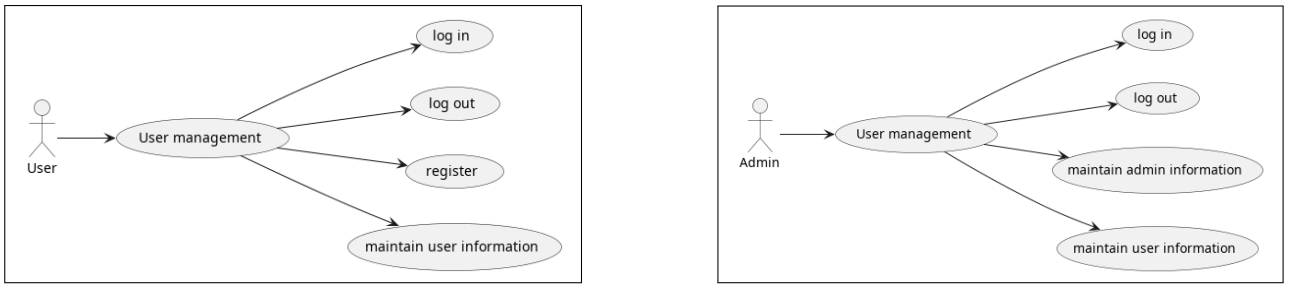
If it's successful, the news will be added/updated from the database and display if the administrator wishes. Otherwise, the state remains unchanged.

4.11.6 Extension points

None

5 User management

This use case is a very big one and particularly important because it is the base of the application, which is the reason why it has been placed in a separate section.



(a) *Standard user* ***(b)*** *Administrator*

Figure 13: *User management*

5.1 Logging in

5.1.1 Brief description

Users can signin / signup to the application by providing their username, email, password, and optionally their name, address, social network accounts, gender, birthday and a picture.

5.1.2 Flow of events

This starts when a user wishes to log into the application.

Basic Flows

- Signin : if the user selected "Sign in" the Sign in sub flow is executed. The system requests that the user enter the login information (email & password). Then :
  1. If the user provides the requested information, the system validates the entered information and logs the user into the system.
  2. If the user selects "forgot password" the system will ask the secret question and if the system validates, the user is logged in and he can modify his password else, he is redirected to the login page.
  3. If the user selects "Register", the user will be redirect to the Register flow.

Alternative Flows

- Invalid email/password : if in the Have an account flow, the user enters an invalid email and/or password, the system displays an error message. The user can choose to either return to the beginning of the Have an account flow or cancel the login, at which point the use case ends.
- Login canceled : if in the sign in sub flow the user decides not to login, the signin is canceled and the user is redirected to the log in sub flow.

5.1.3 Special requirements

None

5.1.4 Preconditions

None

5.1.5 Postconditions

If the log in was successful, the user is now logged into the system. If not, the system state is unchanged.

5.1.6 Extension points

None

5.2 Logging out

5.2.1 Brief description

Users can also log out of the application

5.2.2 Flow of events

This starts when a user wishes to log out the application : the system logs out the user from the system.

5.2.3 Special requirements

User must be logged in to log out

5.2.4 Preconditions

If the log out was successful, the user is now disconnected from the system and he will be redirected to the login page.

5.2.5 Postconditions

None

5.2.6 Extension points

None

5.3 Register

5.3.1 Brief description

When a new user wants to use the application

5.3.2 Flow of events

The system requests that the user enter the register information. This includes :

- Name
- Birthdate
- Phone
- Email
- Password
- Secret question
- Allergies

Basic Flows Once the user has provided the requested information, the system validates the entered information, generates and assigns a unique user id to the user. The user is added to the system and logs into the system.

Alternative Flows

- Invalid form : if the user enters an email already used or if another field of the form is wrong, the system displays an error message ("Something went wrong" for instance).
- Registration canceled : if in the Register sub flow the user decides not to register, the register is canceled and the user is redirected to the login sub flow.

5.3.3 Special requirements

None

5.3.4 Preconditions

User should not already have an account.

5.3.5 Postconditions

If the register was successful, the user is redirected to the home page.

5.3.6 Extension points

None

5.4 Change information

5.4.1 Brief description

This use case allows the user to maintain his personal information. This includes changing user information from the system. He can also ask for deleting his account.

5.4.2 Flow of events

This use case begins when a user wants to change his information.

The system requests that the user specify the function he/she would like to perform (either modify personal information or ask for deleting). Once the user has provided the requested information, one of the sub flows is executed

Basic Flows

- Change personal information : if the user selected "modify personal information", the modify personal information sub flow is executed :
  1. The system retrieves his personal information and displays the user information. The user makes the desired changes to his information. This includes any of the information specified in the Register sub flow.
  2. Once the user has updated the necessary information, the system updates the user record with the updated information.
- Account deletion : If the user selected "Ask for deleting", the Ask for deleting sub flow is executed :
  1. The system sends a notification to the administrator informing him that the user "X" wants to delete his account.
  2. The administrator can decide if he delete it or not. But as a GDPR compliant application, the administrator must delete the account if the user asks for it.

Alternative Flows

- Modify personal information canceled : if in the modify personal information sub flow and the user decides not to modify his information, the modification is canceled and the user is redirected to the user page.
- Ask for deleting canceled : if in the ask for deleting sub flow and the user decides not to delete his account, the demand is canceled he is redirected to the user page.

5.4.3 Special requirements

None

5.4.4 Preconditions

User must be loggged in

5.4.5 Postconditions

If the use case was successful, the user information is updated from the system. Otherwise, the system state is unchanged.

5.4.6 Extension points

None

5.5 Administrator

5.5.1 Brief description

This use case allows the administrator to maintain his personal information and maintain users. This includes changing his personal information from the system. He can also view the list of users and delete user account.

5.5.2 Flow of events

The system requests that the administrator specify the function he/she would like to perform (either modify personal information or deleting).

Basic Flows

- If the administrator selected "modify personal information", the modify personal information subflow is executed :
  1. The system retrieves his personal information and displays the administrator information.
  2. The administrator makes the desired changes to his information. This includes any of the information specified in the Register sub flow.
  3. Once the administrator has updated the necessary information, the system updates the user record with the updated information.
- If the administrator selected "view all users", the view user sub flow is executed :
  1. The system retrieves the list of all users and displays it. The administrator has the list of all users.
  2. He can click and see all the information of a user.
- If the administrator selected "Delete a user ", the Delete a user sub flow is executed :
  1. The administrator receive a notification that a user wants to delete his account.
  2. He decide to delete the account permanently or not (GDPR).
  3. If he decides to delete the account, an email will be sent to the user and the system updates the user record.

Alternative Flows

- Modify personal information canceled : if in the modify personal information sub flow the administrator decides not to modify his information, the modification is canceled and the administrator is redirected to the user page.

5.5.3 Special requirements

None

5.5.4 Preconditions

The administrator must be logged in.

5.5.5 Postconditions

If the use case was successful, the administrator information is updated from the system and the user information is updated or deleted from the system. Otherwise, the system state is unchanged.

5.5.6 Extension points

None

6 Annexes