Food Pants

ID UC 1.1 Manage Pantry
Scope Pantry system
Level User goal
Stakeholders and interests

- **User:** Person interested in managing a digital pantry. Wants the ability to view food items currently in the pantry. Wants the ability to add food items to their pantry. Wants the ability to modify food items in their pantry. Wants the ability to remove food items from their pantry.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: User has gone through setup process. User wants to manage pantry.
Postcondition: Pantry is updated with any user modifications.

Main success scenario:

1. User navigates to the "Pantry" page.
2. System displays a list of all food items in the pantry.
3. User presses the "Add Item" button.
4. User enters the name of the food type to add to pantry in a search box.
5. System displays food type search results in a drop-down menu.
6. User selects the food type to add.
7. User enters the quantity/amount of the food item to add to pantry.
8. System adds food item to pantry.

User repeats steps 3-8 until satisfied with the pantry

Extensions:

a.* Anytime the user decides to stop managing the pantry
    1. *User presses the back button*
        a. User has unsaved changes
            i. *System prompts user to save their changes*
            ii. *User presses "OK"*
            iii. *System redirects user to the a pantry overview page where all food items in the pantry are displayed*

4.a User wants to completely remove a food item from the pantry
    1. *User selects trash icon next to food item*
    2. *System prompts the user for confirmation*
    3. *a. User presses the "Yes" button*
        i. *System deletes the food item from the pantry*
        b. User presses the "No" button

4.b User wants to change the quantity/amount of an existing food item

1. *User selects the pencil button next to a food item to edit it*
2. *User selects the quantity/amount text box*
3. *User enters a new quantity/amount*

4.c User wants to consume an item in the pantry

1. *User selects the "Consume" option next to a food item*
2. *extend <consumePantryItem>*

4.d User wants to search for an item

1. *extend <searchPantry>*

6.a No results meaning food type is not registered in the food type database

1. *User selects the "Add food type" option from the end of the drop-down menu*
2. *extend <addFoodType>*

9.a Food item already exists in pantry

1. *System adds quantity of food item user entered to quantity of food item already in the pantry*

ID UC 1.2 Search Pantry
Scope Pantry system
Level User goal
Stakeholders and interests

- **User:** Person interested in searching the pantry. Wants the ability to find a food item in the pantry.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: User has gone through setup process. User wants to search the pantry.
Postcondition: Food items in the pantry are displayed based on user search criteria.

Main success scenario:

1. User navigates to the "Pantry" page to view the pantry.
2. System displays pantry
3. User navigates to the search bar along the top.
4. System permits input into search bar
5. User types in the search bar.
6. System displays food items in the pantry in a list with names that contain the user's entered search keyword.

User repeats steps 4-5 until done searching the pantry

Extensions:

a.* Anytime the user decides to stop searching the pantry
    1. *User presses the back button*
    2. *System returns full list of food items in the pantry*

5.a Pantry is empty
    1. *System tells users that there are no food items in the pantry*
    2. *System prompts user if they would like to add a food item to the pantry*
    3. *User clicks "Add Item" button*
    4. *extend <managePantry>*

5.b No food items in pantry contain user search criteria in name
    1. *System tells the user no results were found*
    2. *System prompts user if they would like to add a food item to the pantry*
    3. *User clicks "Add Item" button*
    4. *extend <managePantry>*

ID UC 1.3 Consume Pantry Item
Scope Pantry system
Level User goal
Stakeholders and interests

- **User:** Person interested in consuming an item from the pantry. Wants the ability to decrement the quantity of a food item in the pantry.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: Food types to purchase are registered in the food type database. User wants to consume a pantry item.
Postcondition: Shopping list modifications are saved.

Main success scenario:

1. User navigates to the “Pantry” page to view the pantry.
2. System displays the pantry to the user
3. User navigates and selects the food item they want to consume.
4. System displays the particular food item information
5. User clicks “Consume” option.
6. System decrements quantity of food item on pantry.

User repeats steps 3-5 until consumed all food items they wish to

Extensions:

a.* Anytime the user decides to stop consuming items from the pantry
    1. *User presses the back button*
    2. *System returns full list of food items in the pantry*

5.a Food item is decremented to zero
    1. *System removes food item from the pantry*

ID UC 2.1 Manage Nutritional log
Scope Nutrition Log system
Level User goal
Stakeholders and Interests

- **User:** A person utilizing the Nutrition Log system within the FoodPants application to view, add, remove, or edit an item.
- **System maintainer:** Person responsible for application execution.

Precondition: User has undergone setup process. User wants to manage their nutritional log.
Postcondition: Nutritional log updated with modifications.

Main success scenario:

1. User clicks navigates to "Nutrition" page
2. System displays nutrition page and nutrition timeline to the user
3. User views their day, week, and month timeline
4. System displays each corresponding nutritional timeline depending on the day, week, month time frame
5. User selects specific nutritional item in timeline
6. System displays nutritional information about specific item
7. User clicks "Edit" button on specific item
8. System provides a form with the current item values which can be changed
9. User alters the desired values within the form which include the timestamp for consumption and nutritional information (calories, carbs, protein, fat, cholesterol, sodium)
10. System updates the form fields
11. User submits the form
12. System updates specific nutritional item

User repeats steps 3-9 until satisfied with the nutritional log.

Extensions:

a.a User exits page abruptly
    1. *Nutrition page and information is no longer displayed to the user*

a.b System encounters an unexpected error
    1. *System will exit nutritional page*
    2. *System will log the error internally*

2.a If the user wants to manually add an item to the nutrition log
    1. *User presses the "+" Add Item button*
    2. *User enters the name of the Item to add in a search box.*
    3. *System displays search results in a drop-down menu.*
    4. *a. User selects the item they want to add.*

b. If the food type is not registered in the food type database

i. User selects the "Add food type" option from the end of the drop-down menu

ii. extend <addFoodType>

5. *User enters the quantity they consumed.*

   a. The user can change the timestamp for consumption if it is not the current time.

6. *User selects create item*

7. a If the User wants to remove the existing item from the nutrition log

1. *User clicks on the 'Delete Item' button within the item details interface*
2. *System asks the user to confirm that they want to delete the item*
3. *User clicks 'Confirm'*
4. *System removes item from the nutrition log*

11. a User hits 'X' to exit the nutritional modification form

1. *System exits the form without saving the modifications*

ID UC 2.2 Manage Nutrition Goals
Scope Nutrition Log system
Level User goal
Stakeholders and Interests

- **User:** A person utilizing the Nutrition Log system within the FoodPants application to set and manage nutritional goals.
- **System maintainer:** Person responsible for application execution.

Precondition: User has undergone setup process. User is viewing the Nutrition page. User wants to manage their nutrition goals.
Postcondition: A nutrition goal has been added, edited, or removed.

Main success scenario:

1. User selects the goals button to retrieve nutritional goals menu
2. System displays nutritional goals menu
3. User selects the "+" button
4. System displays nutrition goal creation form
5. User picks the nutrient type, goal value, and goal type (minimum or maximum)
6. System updates form with modifications
7. User submits the new nutritional goal
8. System updates nutritional goals menu with the new goal

User repeats steps 3-8 until satisfied with the management of their nutrition goals

Extensions:

a.a User exits page abruptly

2. *Nutrition page and information is no longer displayed to the user*

a.b System encounters an unexpected error

3. *System will exit nutritional page*
4. *System will log the error internally*

User repeats this step until they have gotten all the way out of the specified process

3. a If the User wants to modify a pre-existing nutrition goal

1. *User clicks on the edit goal button within the nutritional goals menu*
2. *System displays form where current goal values can be changed*
3. *User alters the desired values within the form*
   a. User hits the 'Confirm' button
   b. User hits an 'X' which exits without saving changes
4. *System updates goal information*

3. b If the User wants to remove a pre-existing goal from the nutrition log

1. *User clicks on the edit goal button within the nutritional goals interface*
2. *System displays form with the current goal values*

3. *User clicks on the delete goal button within the edit goal interface*
4. *System asks the user to confirm that they want to delete the goal*
5. *User clicks 'Confirm'*
6. *System removes goal from the nutrition goals menu*

ID UC 2.3 Manage Nutrition Report
Scope Nutrition Log system
Level User goal
Stakeholders and Interests

- **User:** A person utilizing the Nutrition Log system within the FoodPants application to view, create, modify, or remove a nutrition report.
- **System maintainer:** Person responsible for application execution.

Precondition: User has undergone setup process. User is viewing the Nutrition page. User wants to manage nutrition reports.
Postcondition: Nutrition reports are updated with any user modifications.

Main success scenario:

1. User selects the nutrition reports button to navigate to the nutrition reports menu
2. System displays nutrition reports menu
3. User views nutritional reports via scroll along the page
4. System displays a list of all previous nutritional reports
5. User selects the "+" button
6. System displays nutrition report creation form
7. User enters the desired time range in which they would like the report
8. System updates form with desired time range
9. User submits the creation form
10. System updates nutritional reports menu with new nutritional report

User repeats steps 5-8 until satisfied with the management of their nutrition goals

Extensions:

a.a User exits page abruptly
    3. *Nutrition page and information is no longer displayed to the user*

a.b System encounters an unexpected error
    5. *System will exit nutritional page*
    6. *System will log the error internally*

4. a Nutritional reports list is empty
    1. *System displays a message saying no nutritional reports exist yet*

5. a If the User wants to modify a pre-existing nutrition report
    1. *User clicks on the edit report button within a particular report*
    2. *System displays form with the current report specifications which can be changed*
    3. *User alters the desired values within the form*
        a. User hits the 'Confirm' button
        b. User hits an 'X' which exits without saving changes
    4. *System updates report information*

5. b If the User wants to remove a pre-existing report from the nutrition report menu

1. *User clicks on the edit report button within the specified report*
2. *System displays form with current report specifications*
3. *User clicks on the delete report button within the edit report interface*
4. *System asks the user to confirm that they want to delete the report*
5. *User clicks 'Confirm'*
6. *System removes report from the nutrition reports menu*

ID UC 3.1
Scope Recipe creation system.
Level User goal
Stakeholders and interests

- **Project Leader:** Person responsible for overseeing system goals.
- **Project Member:** Person responsible for implementing and maintaining the system.
- **System:** Person responsible for recipe creation execution.

Precondition: The user can successfully execute the application.
Postcondition: Recipe successfully created.

Main success scenario:

1. User clicks on “Recipes” from sidebar menu
2. System displays recipe page with stored recipes
3. User creates a recipe manually
4. System provides a recipe creation form
5. User enters recipe information into form
6. System updates form fields with user input
7. User submits recipe form
8. System verifies form has required information
9. New recipe is added into recipes list

Extensions:

a.a User exits form abruptly
    1. *recipe creation will be canceled.*

a.b User exits page abruptly
    1. *Recipe creation will be canceled*

a.c User exits application abruptly
    1. *Recipe creation will be canceled*

b.* System encounters an unexpected error
    1. *Recipe creation will be canceled*
    2. *Page will be restarted*

2.a If there are no recipes yet created
    1. *System will only displayed recommended recipes*

5.a User does not enter a recipe name into form
    1. *System will ask user for recipe name*

5.b If user does not enter at least one ingredient into form
    1. *System will ask user for at least one ingredient*

5.c If user does not enter number of servings into form
    1. *System will ask user for number of servings*

ID UC 3.2
Scope View recipe system
Level User goal
Stakeholders and interests

- **Project Leader:** Person responsible for overseeing system goals.
- **Project Member:** Person responsible for implementing and maintaining the system.
- **System:** Person responsible for recipe viewing execution.

Precondition: User can successfully execute the application. User has created at least one recipe.
Postcondition: Recipes successfully viewed.

Main success scenario:

1. User selects “Recipes” from sidebar menu
2. System displays list of recipes to user
3. User searches for specific recipe by name
4. System displays recipe search results
5. User clicks on a specific recipe
6. System displays information from recipe

Extensions:

a.a User exits page abruptly
    1. *Recipes are no longer displayed to the user*

a.b System encounters an unexpected error
    1. *System will exit recipe page*
    2. *System will store the error internally*

2.a User has no recipes
    1. *System will only display recommended recipes in list*

3.a User searches for recipe based on ingredients
    1. *System displays recipes based on matching ingredients*

4.a If no matching recipes from search
    1. *System will display a blank page to the user*

ID UC 3.3
Scope Recipe to shopping list system
Level User goal
Stakeholders and interests

- **Project Leader:** Person responsible for overseeing system goals.
- **Project Member:** Person responsible for implementing and maintaining the system.
- **System:** Person responsible for recipe viewing execution.

Precondition: A user can successfully view recipes list.
Postcondition: Recipe successfully added to shopping list

Main success scenario:

1. User clicks on “Recipes” from sidebar menu
2. System displays stored recipes to user
3. User views recipes
4. System displays list of recipes
5. User selects specific recipe
6. System displays specific recipe information
7. User adds recipe to shopping list
8. System adds recipe ingredients to shopping list

Extensions:

a.a User exits page abruptly
    4. *Recipes are no longer displayed to the user*

a.b System encounters an unexpected error
    7. *System will exit recipe page*
    8. *System will store the error internally*

4.a If user has no recipes
    1. *System will only display recommended recipes in recipe list*
    2. *System will ask user to create a recipe or choose from recommended*

7.a Recipe ingredients already exist in shopping list
    1. *System will add the ingredients again*

ID UC 3.4 Modify Recipe
Scope Recipe System
Level User goal
Stakeholders and Interests

- **User:** A person utilizing the Recipe System within the FoodPants application to modify existing recipes.
- **System maintainer:** Person responsible for application execution.

Precondition: Recipes exist within the Recipe System so there is something to modify. User wants to modify a recipe.
Postcondition: A recipe has been edited.

Main success scenario:

1. User navigates to the recipe page.
2. System displays all recipes.
3. User selects the "Edit Recipe" button.
4. System displays the recipe specifications.
5. User edits food type and amount fields and/or the recipe instruction field, and selects the "Save Changes" button to save edited fields.
6. System confirms that changes were saved, and replaces the old recipe with the user-edited recipe.
7. User is returned to the recipe view page with the newly modified recipe listed.

Extensions:

3.a User has no recipes
    1. *User selected the "+" button.*
    2. *Recipe creation page is displayed to the user.*

5.a The user wants to add a new food type
    1. *User selects the "+" button in the food type field.*
    2. *New food type page is displayed*
    3. *User fills required fields.*
    4. *User selects the "Create Item" button.*
    5. *Recipe modification page is displayed with newly created food type added to the food type field.*

5.a The user wants to exit the recipe modification page without saving modifications
    1. *The user selects the "Discard Changes" button.*
    2. *The user is returned to the recipe view page.*

5.b The user wants to delete the recipe.
    1. *The user selects the "Delete Recipe" button.*

2. *The user is returned to the recipe view page and the recipe is no longer listed.*

ID UC 3.5 Recommend Recipes
Scope Recipe System and Digital Pantry
Level User goal
Stakeholders and Interests

- **User:** A person utilizing the Recipe System within the FoodPants application to view recommended recipes.
- **System maintainer:** Person responsible for application execution.

Precondition: Recipes exist within the Recipe System and the user has items in their pantry. User wants to view recommended recipes based on what they already have in their pantry.
Postcondition: Recommended recipes are displayed to the user for further action to be taken.

Main success scenario:

1. User navigates to the recipe page.
2. System displays all recipes.
3. User selects the "View Recommendations" button.
4. System displays recommended recipes.
5. User selects a recipe to view

Extensions:

3.a User exits the recipe viewing page
    1. *User selects the "Back" button.*
    2. *Home page is displayed to the user.*

5.a The user wants to stop viewing recommended recipes
    1. *User selects the "View Recommendations" button, removing its highlight*
    2. *Default recipe page is displayed*

ID UC 3.6 Make Recipe
Scope Recipe system
Level User goal
Stakeholders and Interests

- **User:** Person who has stored recipes and items within the FoodPants digital pantry, wants to use stored items to make a recipe and log the results to their nutrition log.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: The user has created the recipe within the recipe system, and has decided how much of the recipe they will eat (they can perform this step after cooking if necessary)
Postcondition: Used items will have been consumed from the user's pantry, and the recipe and nutritional info will be stored in the user's nutrition log.

Main success scenario:

1. User selects to make the recipe.
2. System removes items from the digital pantry that were used to make the recipe.
3. User is prompted for how much of the recipe they consumed.
4. System adds the amount of the recipe consumed is added to the nutrition log.
5. include (manageNutritionLog)
6. User is prompted to enter how much of the recipe is leftover.
7. System stores leftovers as part of the User's digital pantry.
8. include (managePantry)
9. User is returned to the recipe view menu.

Extensions:

a. * The desired recipe does not exist
    1. *The System will not allow the user to cook the recipe.*
    2. *The user can add the recipe to the system.*

4. a The user does not have all the required items to cook the recipe.
    1. *The system prompts the user if they want to cook the recipe anyways.*
    2. *a. User confirms to cook the recipe anyways.*
        i. *User selects to consume only existing items from the pantry.*
        ii. *User selects to consume no items from the pantry.*
    b. User requests desired items to be added to the shopping list.
        i. *System adds items to the shopping list (include addRecipeItemsToShoppingList).*
        ii. *User is redirected to the shopping list menu, preparing the recipe is aborted.*

6. a The user has not set up serving sizes for this recipe

    1. *The system prompts the user to dictate how many servings the recipe makes*
    2. *The user enters how many servings were consumed.*

9. a The user has no food remaining or does not wish to record leftovers
    1. *The user can press an option to skip this step.*

Special Requirements

- Users can utilize this function even if they are missing ingredients.
- This will interface nicely with the shopping list feature so that users can automatically fill in missing items.

ID UC 4.1 Manage Shopping List
Scope Shopping List system
Level User goal
Stakeholders and interests

- **User:** Person interested in managing a shopping list. Wants the ability to modify their shopping list. Wants the ability to add items to their shopping list based on a recipe. Wants an easily visible display of their shopping list.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: Food types to purchase are registered in the food database. User wants to manage a shopping list.
Postcondition: Shopping list modifications are saved.

Main success scenario:

1. User navigates to the "Shopping List" page to view the list.
2. System displays the list of shopping items.
3. User opens the "Add Item" form and enters the name of the food to purchase.
4. System displays food search results.

User repeats steps 3-4 until the food with the given name is found

5. User submits the form with the selected food to purchase.
6. System adds the item to the shopping list and displays the updated list.

User repeats steps 3-6 until satisfied with the shopping list

Extensions:

a.* Anytime the user decides to stop managing the shopping list
    1. *User exits the active window*

3.a User wants to add items to their list from a recipe instead of manually
    1. *User navigates to the "Add Items From Recipe" option*
    2. *extend <addIngredientsToCart>*
    3. *System adds the recipe items to the shopping list*

3.b User presses the "Delete Item" button to delete an existing item
    1. *System prompts the user for confirmation*
    2. *a. User presses the "Yes" button*
        i. *System deletes the item*
    b. User presses the "No" button

3.c User wants to change the quantity/amount to purchase for an existing item
    1. *User selects the quantity/amount text box*
    2. *User enters a new quantity/amount*

4.a Food is not registered in the food database

1. *User selects the "Add food" option from the end of the drop-down menu*
2. *extend <addFood>*

6.a User wants to export the shopping list

1. *extend <exportShoppingList>*

ID UC 4.2 Export Shopping List
Scope Shopping List system
Level User goal
Stakeholders and Interests

- **User:** Person that is interested in creating a grocery list of their items and exporting it from the application.
- **System maintainer:** Person responsible for application execution.

Precondition: Shopping list is created, desired items have been added
Postcondition: A shopping list is added to shopping lists and can be viewed from that menu.

Main success scenario:

1. User is viewing their shopping list and selects to export the shopping list.
2. System prompts the user on how they would like to export the list.
3. User selects that they would like to print the list.
4. System displays list as a print preview.
5. User prints the list.
6. System saves the list for view in the list menu.

Extensions:

a. * If the list is empty
    1. *The system will deny the user request, telling them to add items*

3. a If the user changes their mind,
    1. *The system will return the user to the main shopping list window.*

5. a The user wants to save the list as a PDF.
    1. *The system will export the list to a PDF.*
    2. *The user can download the file.*

5. b The user wants to save the list as a text file.
    1. *The system will export the list to a text file.*
    2. *The user can download the file.*

5. c The user wants to save the list as a csv file.
    3. *The system will export the list to a csv file.*
    4. *The user can download the file.*

5. d User does not want to export the list
    1. *The system will proceed to save the list for view in the list section.*

ID UC 4.3 Mark Purchased Items
Scope Shopping List system
Level User goal
Stakeholders and interests

- **User:** Person interested in shopping list. Wants the ability to cross out purchased items from their shopping list (one by one or all items at once).
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: User had previously added items to the shopping list. User purchased all items from their shopping list or a subset of the items from their shopping list. User wants to mark items from their shopping list as purchased.
Postcondition: Items marked as purchased are removed from the shopping list. Items marked as purchased are added to the pantry.

Main success scenario:

1. User navigates to the “Shopping List” page to view the list.
2. System displays all items in the shopping list.
3. User marks a subset of the items as purchased.
4. System moves the selected items from the shopping list to the pantry.

User repeats steps 3-4 until the shopping list and pantry accurately reflect the user’s purchases

Extensions:

a.* Anytime the user decides to stop managing the shopping list
    1. *User exits the active window*

3.a. The list is empty
    1. *The system will deny the user request, telling them to add items*

ID UC 5.1 Startup
Scope Startup
Level Summary
Stakeholders and Interests

- **New User:** New person utilizing the FoodPants application for the first time, wants to become familiarized with the application.
- **System maintainer:** Person responsible for application execution.

Precondition: FoodPants is installed as an executable on the user's machine and running.
Postcondition: New User has a profile created, and has been familiarized with the system.

Main success scenario:

1. The system greets the new user
2. The system prompts the new user for their details.
3. The user enters details such as name, gender, height, and weight for calorie info.
4. The system saves the user data and configures the new profile.
5. The user is shown the digital pantry.
6. The user is prompted to enter items they may already have in their pantry.
7. include (addFoodToPantry)

Continue step 7 until the user decides they are satisfied with their pantry.

8. User is shown the recipe menu and suggested a recipe to add
9. include (getNewReccomendedRecipies)
10. User is shown the shopping list and prompted to add an item of their choice
11. include (manageShoppingList)
12. System thanks the user for using the walkthrough, displays documentation for further questions
13. User finishes tutorial and redirects to the home window.

Extensions:

a. * The user does not wish to continue the tutorial
    1. *The user can click a button to exit the tutorial*
    2. *The system will resume processes from the home menu*

3. a If the user does not wish to enter gender, height, or weight.
    1. *These fields are optional, some nutrition features will be disabled.*

6. b User does not wish to set up the digital pantry
    1. *The user can press an option to skip this step for now.*

8. a The user does not want to add the suggested recipe
    1. *The user can press an option to skip this step.*

10. a The user does not want to add an item to their list
    1. *The user can press an option to skip this step.*
13. a The user does not want to view the documentation
    1. *The user can deny this option to finish the tutorial.*

Special Requirements

- Provide startup as an embedded walkthrough within the application.

ID UC 6.1 Manage Food Types
Scope Food system
Level User goal
Stakeholders and interests

- **User:** Person interested in managing the food types. Wants to customize their available food types for tracking within their pantry, shopping list, recipes, and nutrition log.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: Basic foods are predefined (but still modifiable). User wants to manage their foods.
Postcondition: Food modifications are saved.

Main success scenario:

1. User navigates to the "Foods" page to view the foods.
2. System displays the defined foods.
3. User opens, fills out, and submits the "Add Food" form.
4. extend <addFood>
5. System adds the food to the list of foods and displays the updated list.

User repeats steps 3-5 until satisfied with the list of foods

Extensions:

a.* Anytime the user decides to stop managing the list of foods
    1. *User closes the active window*

3.a User presses the "Delete Food" button to delete an existing food type
    1. *System prompts the user for confirmation*
    2. *a. User presses the "Yes" button*
        i. *System deletes items in pantry of the given food type*
        ii. *System deletes items in shopping list of the given food type*
        iii. *System removes the given food type from the list of food types displayed*
    b. User presses the "No" button

3.b User wants to edit an existing food details
    1. *User opens, fills out, and submits the "Edit Food" form*
    2. *System updates the food in the list of foods and displays the updated list*

ID UC 6.2 Create New Food Type
Scope Food system
Level User goal
Stakeholders and Interests

- **User:** Person interested in managing the foods. Wants to customize their available foods for tracking within their pantry, shopping list, recipes, and nutrition log.
- **System maintainer:** Person responsible for application execution. Wants to satisfy customer interests.

Precondition: User wishes to create a new food or manage existing foods.
Postcondition: New food is created and is selectable by the user.

Main success scenario:

1. User navigates to the "Food Types" page to view the food types.
2. System displays food types.
3. User navigates to the "Modify Food Types" option and presses the "Add Food Type" button.
4. System displays new food type specifications.
5. User enters required information to complete a new food type, and selects the "Create Food Type" button.
6. System confirms new food type was successfully created.
7. The Food Types page is displayed to the user.

User repeats steps 4-5 until satisfied with the list of food types

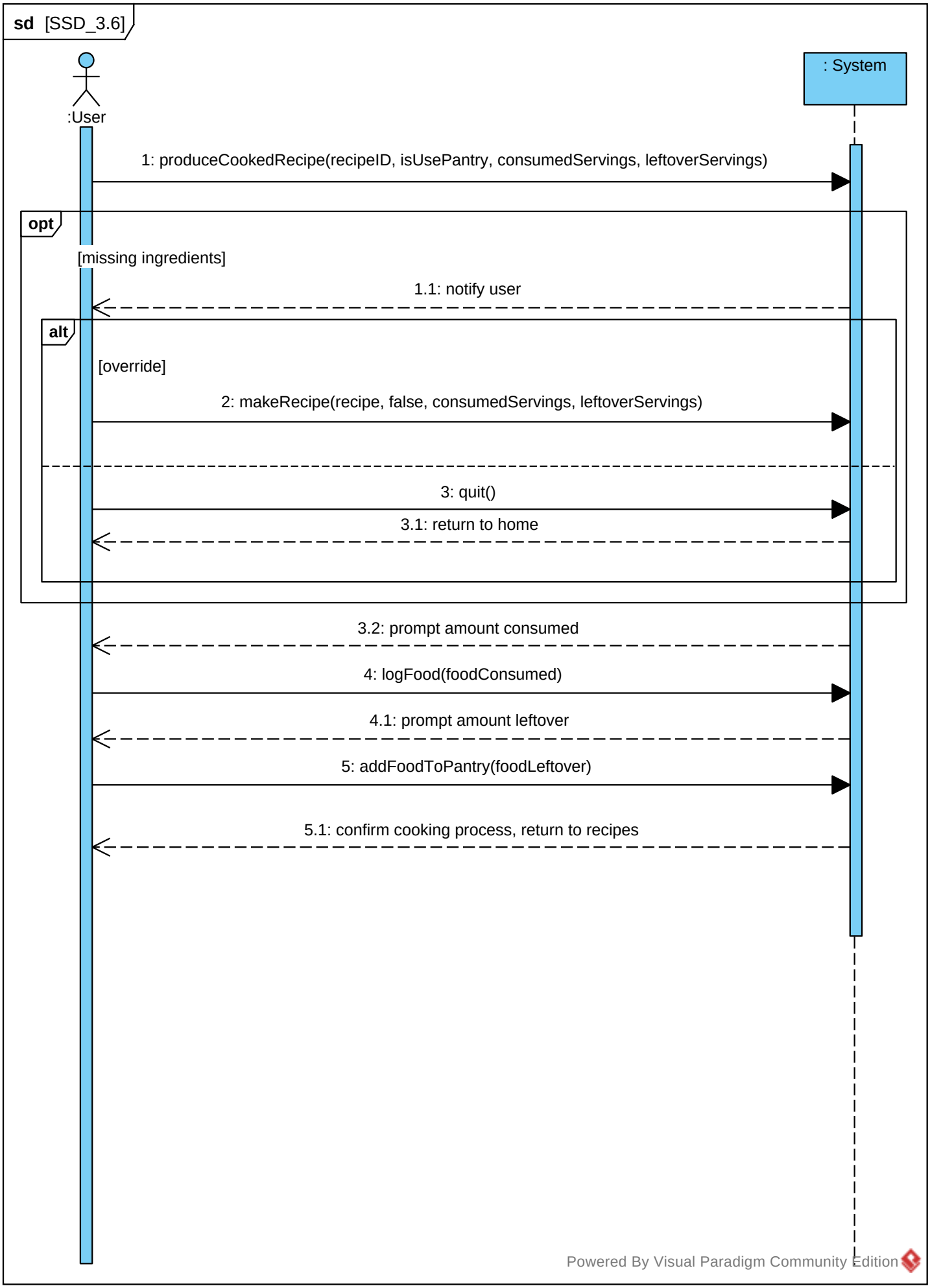
8. User presses the "Save" button.

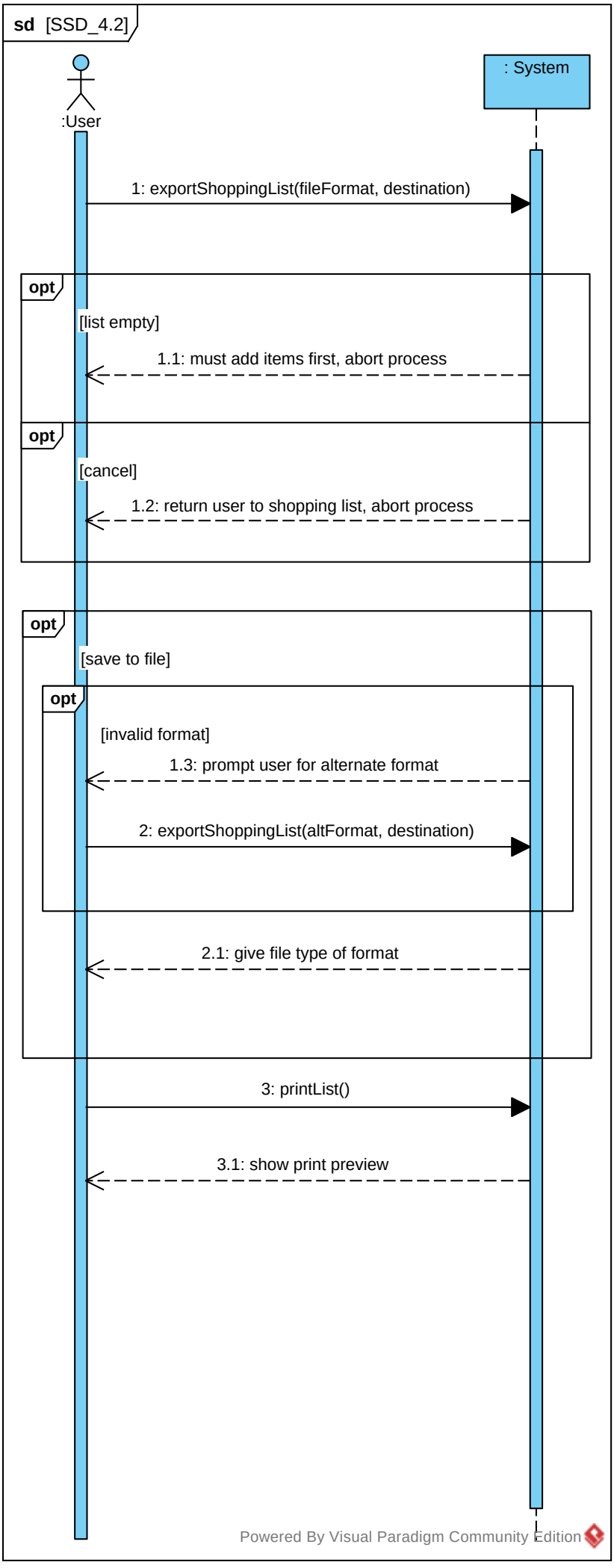
Extensions:

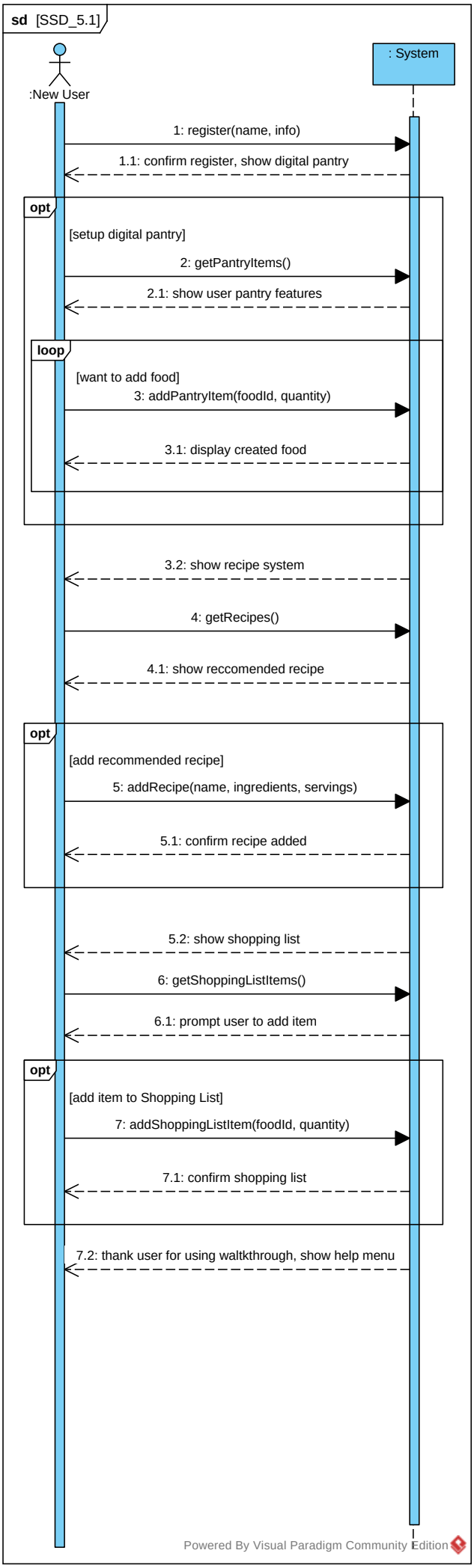
3a.* The user no longer wants to manage food types.
    1. *User presses the "Back" button*
        a. *User has unsaved changes*
            i. *System prompts user to save their changes*
            ii. *User presses "OK"*
            iii. *System redirects user to the "Modify Shopping List" page*

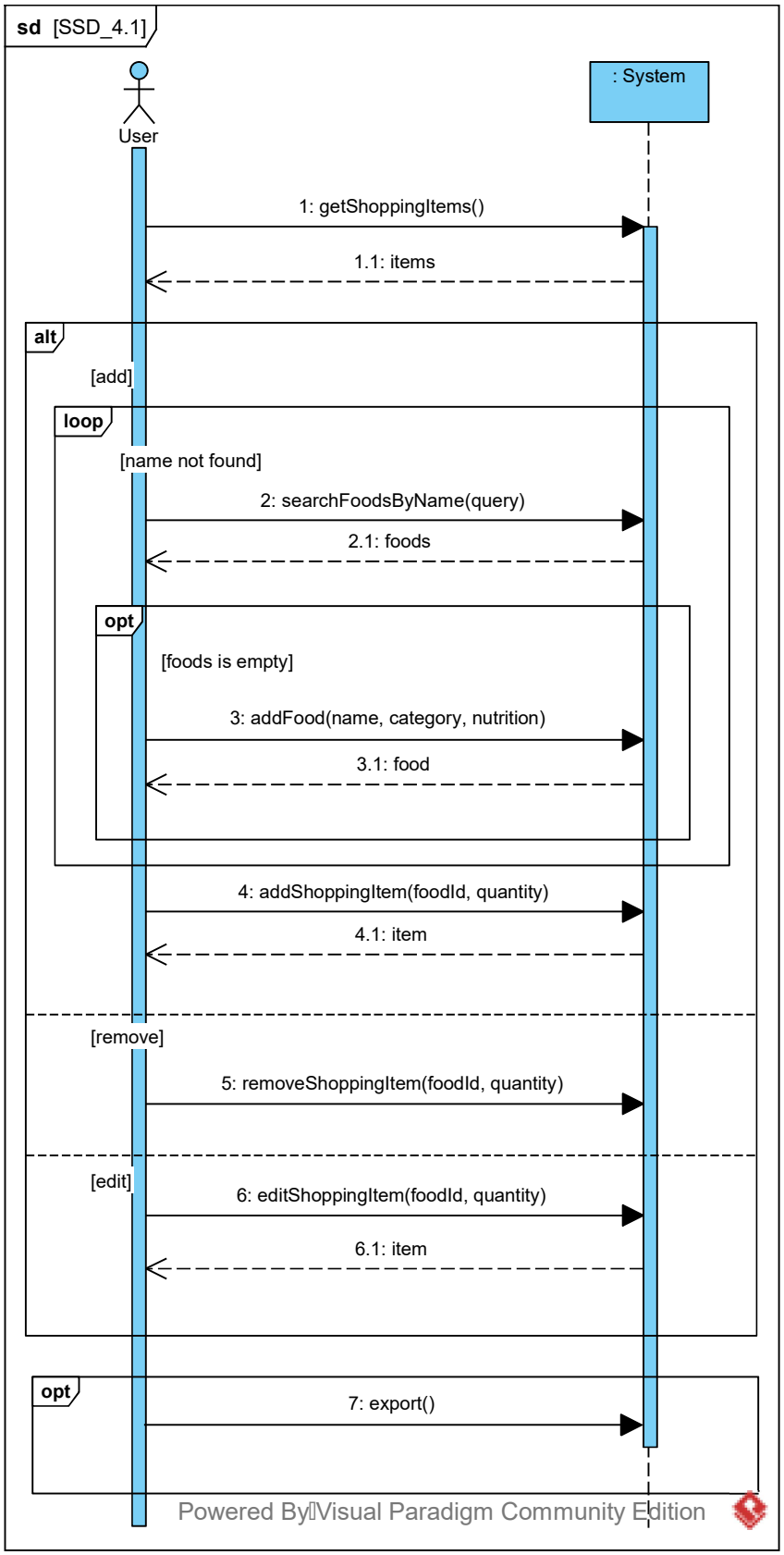
5.a User enters an existing name in the "Food Name" text box
    1. *The system prompts the user to change the name and prevents the user from exiting the text box.*
    2. *Return to step 3 of the main success scenario.*

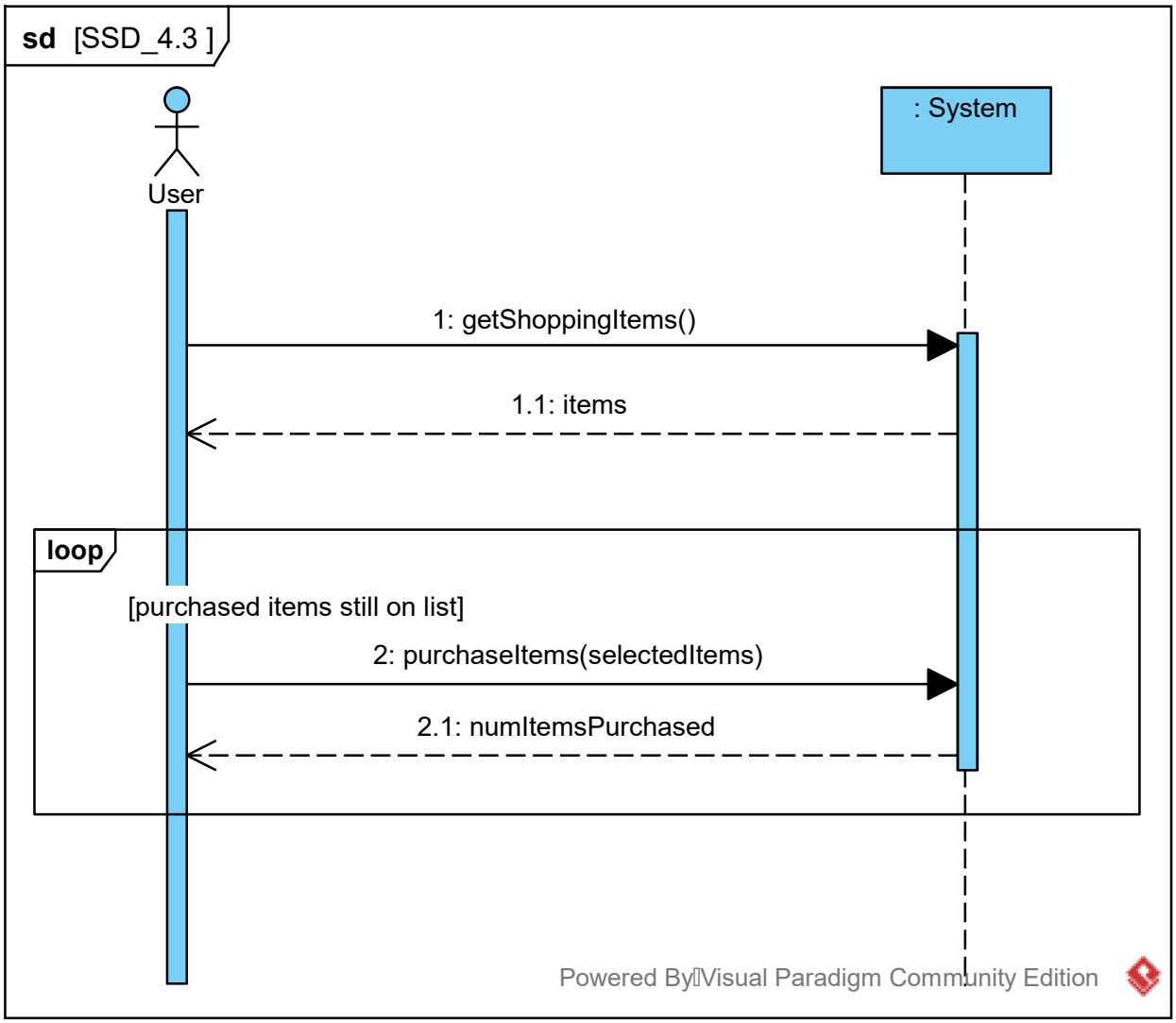
5.b User leaves a required text box empty
    1. *User prompts user to fill out required text fields.*
    2. *Return to step 7 of main success scenario.*

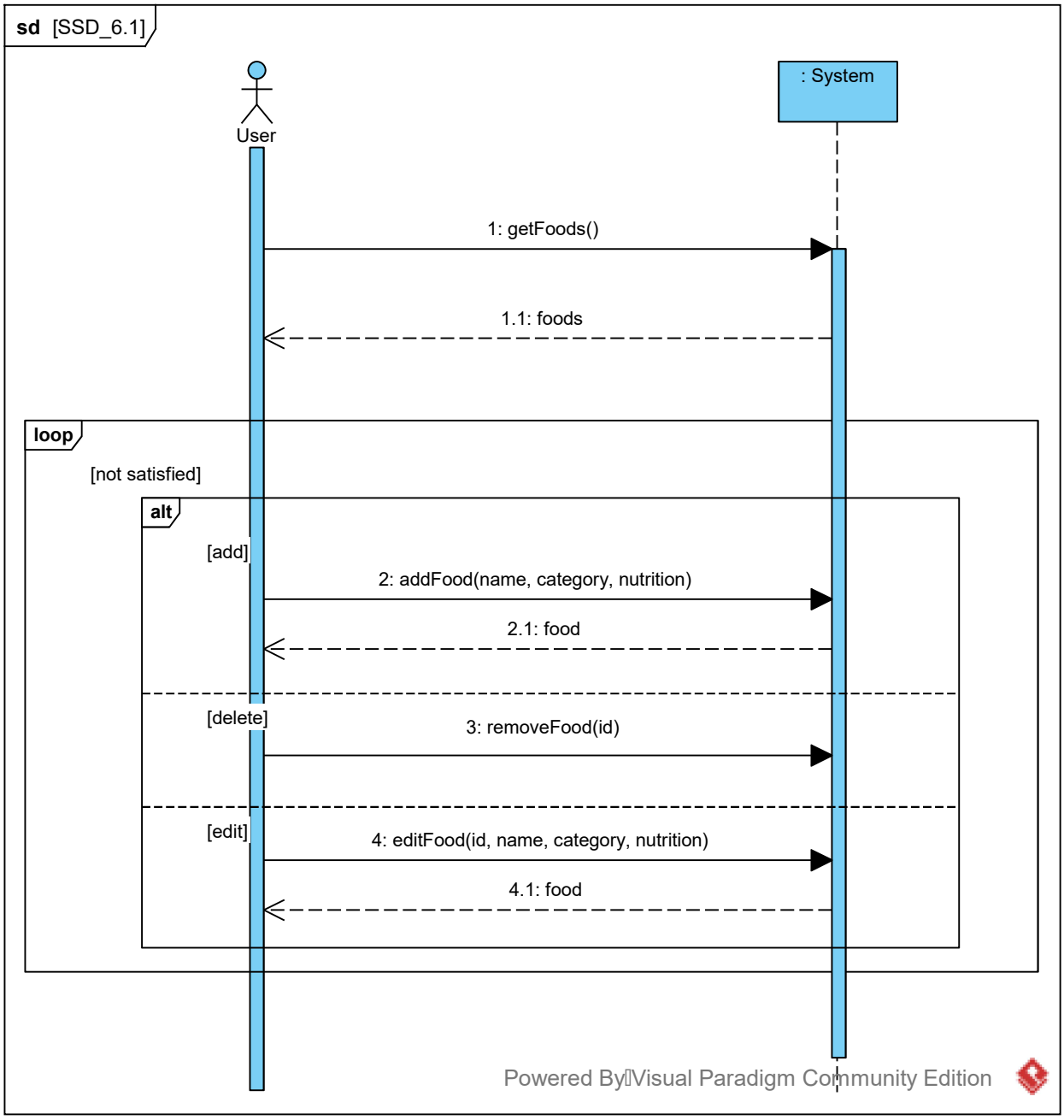

sd [SSD_3.6]
:User
: System
1: produceCookedRecipe(recipeID, isUsePantry, consumedServings, leftoverServings)
opt
[missing ingredients]
1.1: notify user
alt
[override]
2: makeRecipe(recipe, false, consumedServings, leftoverServings)
3: quit()
3.1: return to home
3.2: prompt amount consumed
4: logFood(foodConsumed)
4.1: prompt amount leftover
5: addFoodToPantry(foodLeftover)
5.1: confirm cooking process, return to recipes
Powered By Visual Paradigm Community Edition

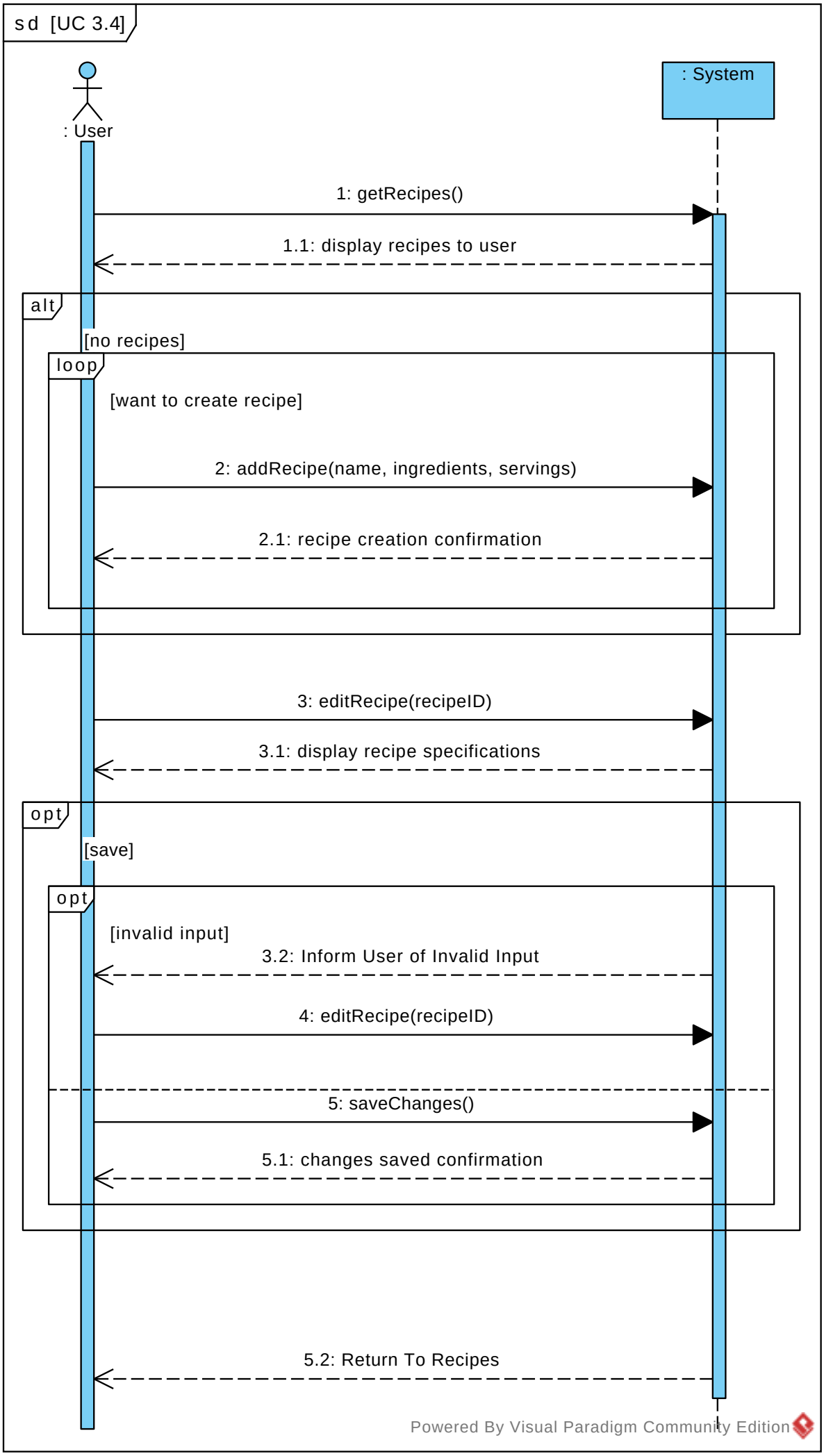

sd [SSD_4.2]
:User
: System
1: exportShoppingList(fileFormat, destination)
opt
[list empty]
1.1: must add items first, abort process
opt
[cancel]
1.2: return user to shopping list, abort process
opt
[save to file]
opt
[invalid format]
1.3: prompt user for alternate format
2: exportShoppingList(altFormat, destination)
2.1: give file type of format
3: printList()
3.1: show print preview
Powered By Visual Paradigm Community Edition

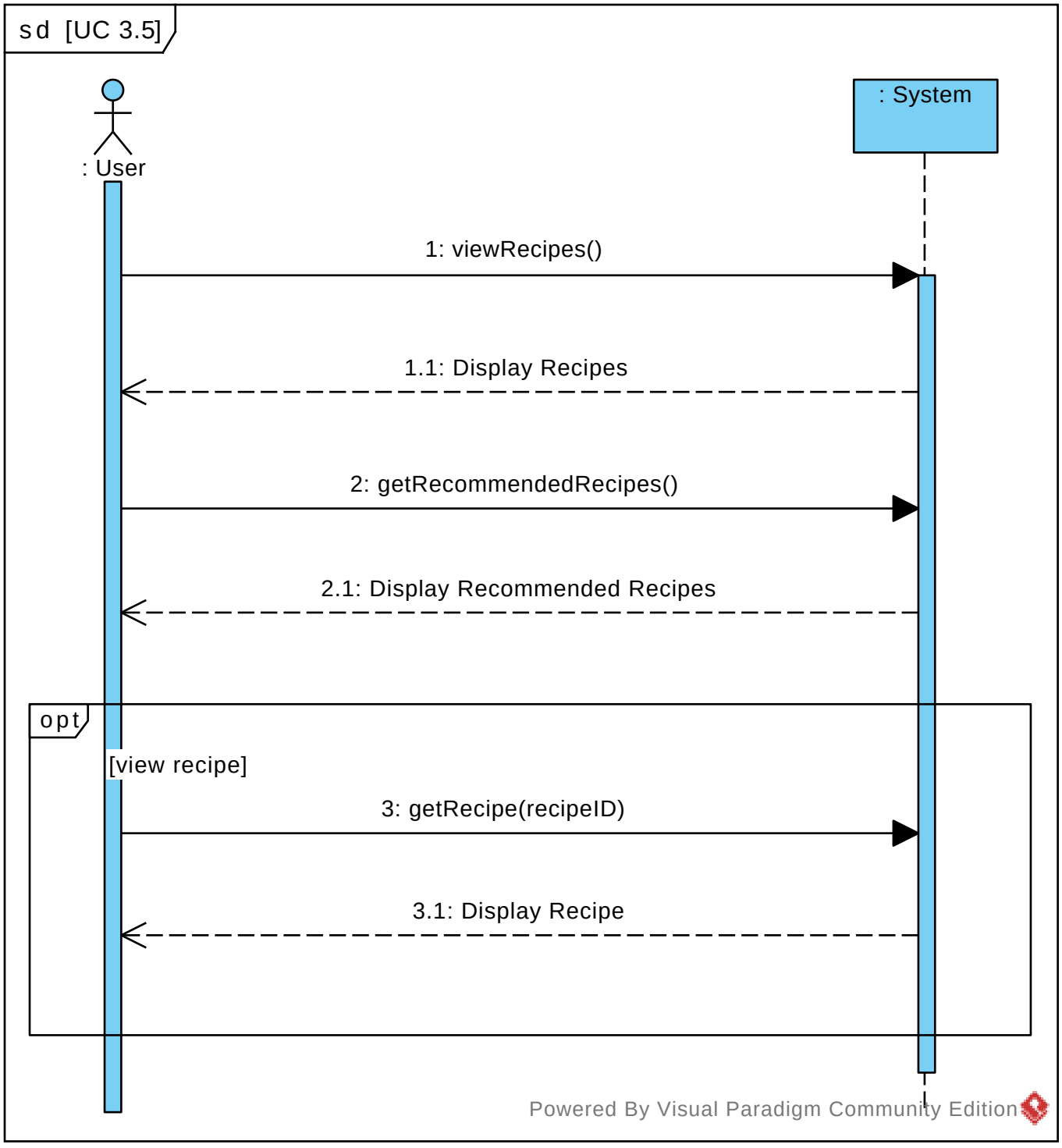

sd [SSD_5.1]
:New User
: System
1: register(name, info)
1.1: confirm register, show digital pantry
opt
[setup digital pantry]
2: getPantryItems()
2.1: show user pantry features
loop
[want to add food]
3: addPantryItem(foodId, quantity)
3.1: display created food
3.2: show recipe system
4: getRecipes()
4.1: show reccomended recipe
opt
[add recommended recipe]
5: addRecipe(name, ingredients, servings)
5.1: confirm recipe added
5.2: show shopping list
6: getShoppingListItems()
6.1: prompt user to add item
opt
[add item to Shopping List]
7: addShoppingListItem(foodId, quantity)
7.1: confirm shopping list
7.2: thank user for using waltkthrough, show help menu
Powered By Visual Paradigm Community Edition

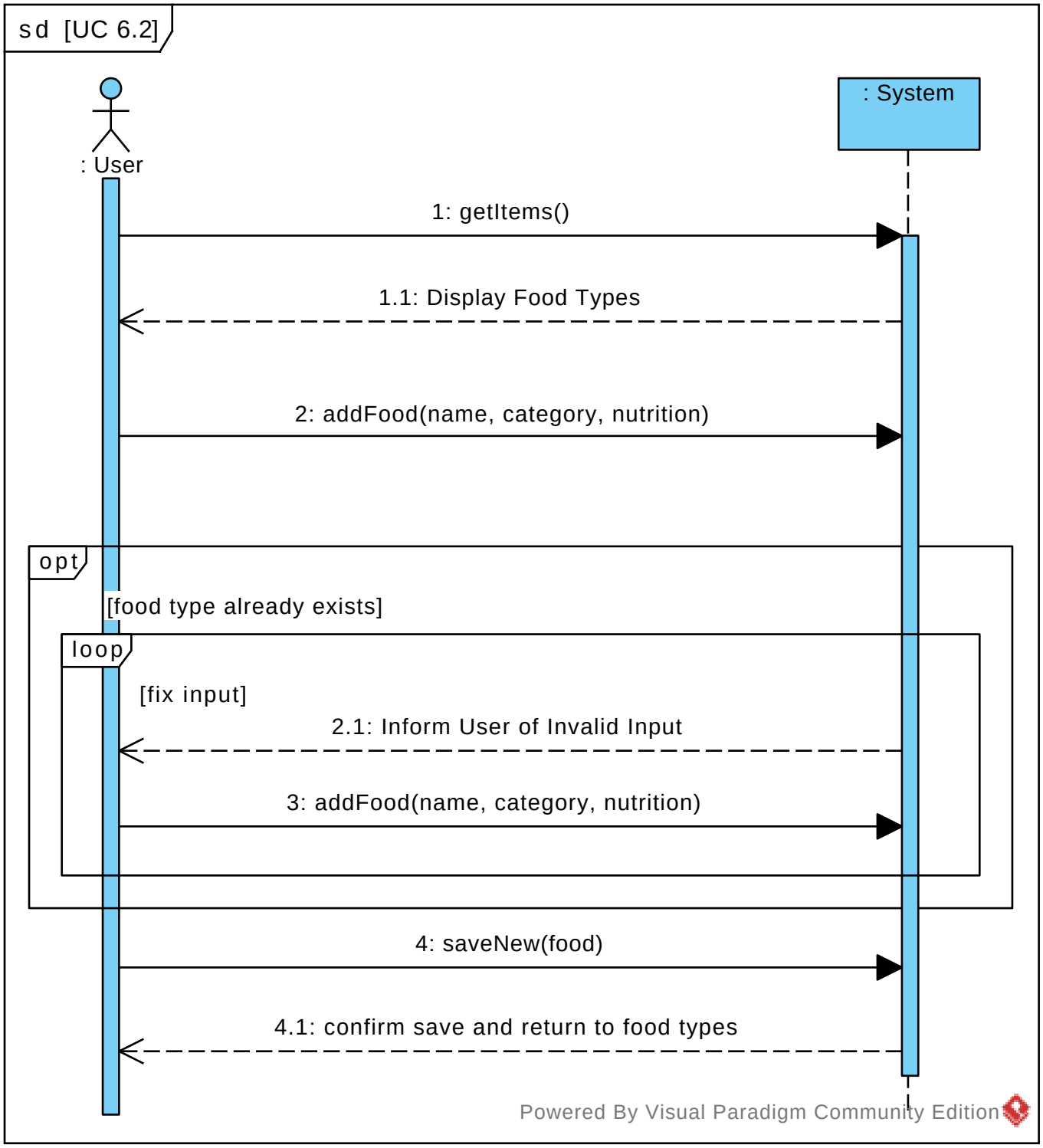

sd [SSD_4.1]
User
: System
1: getShoppingItems()
1.1: items
alt
[add]
loop
[name not found]
2: searchFoodsByName(query)
2.1: foods
opt
[foods is empty]
3: addFood(name, category, nutrition)
3.1: food
4: addShoppingItem(foodId, quantity)
4.1: item
[remove]
5: removeShoppingItem(foodId, quantity)
[edit]
6: editShoppingItem(foodId, quantity)
6.1: item
opt
7: export()
Powered By Visual Paradigm Community Edition

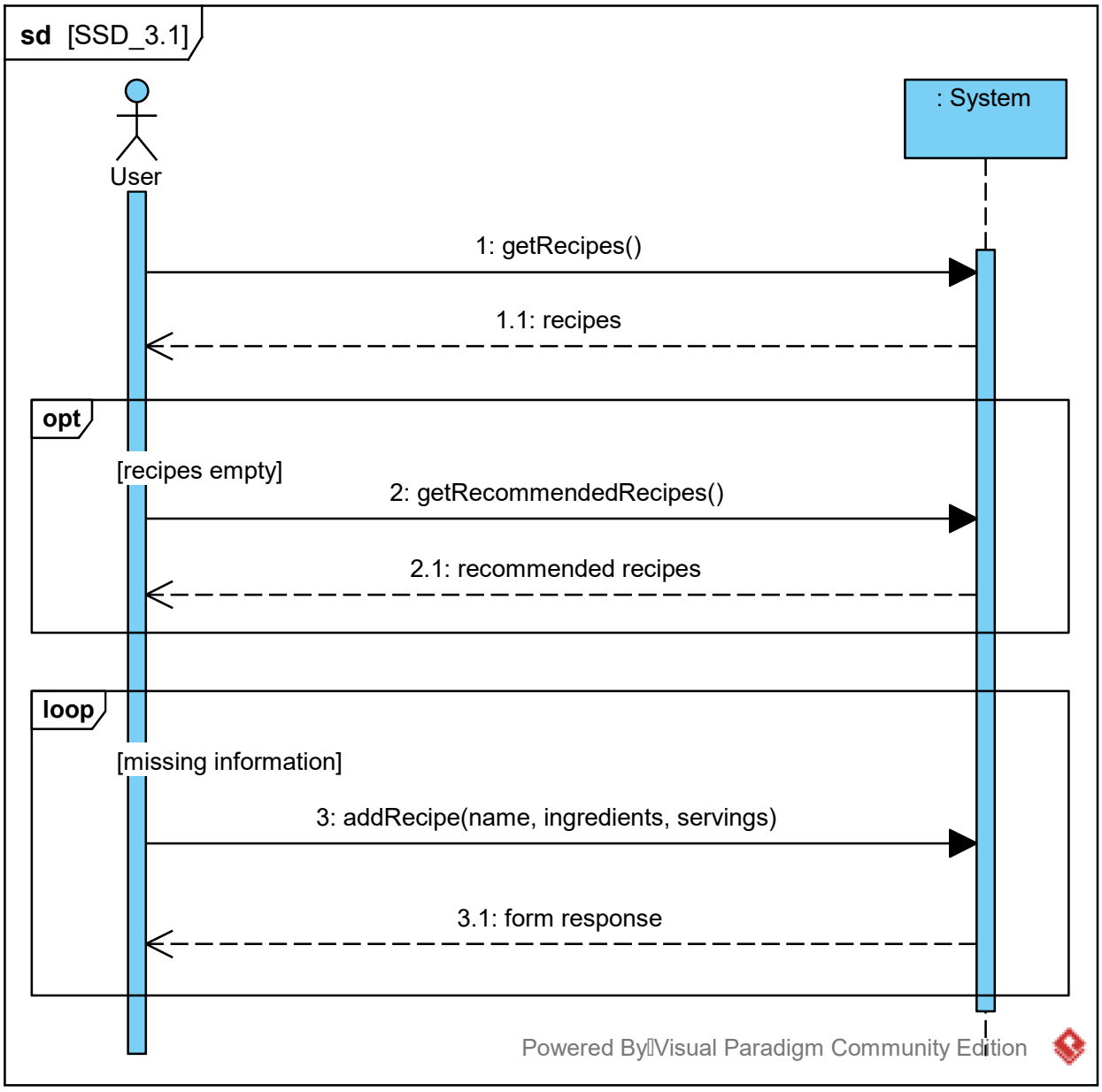

sd [SSD_4.3]
User
: System
1: getShoppingItems()
1.1: items
loop
[purchased items still on list]
2: purchaseItems(selectedItems)
2.1: numItemsPurchased
Powered By Visual Paradigm Community Edition

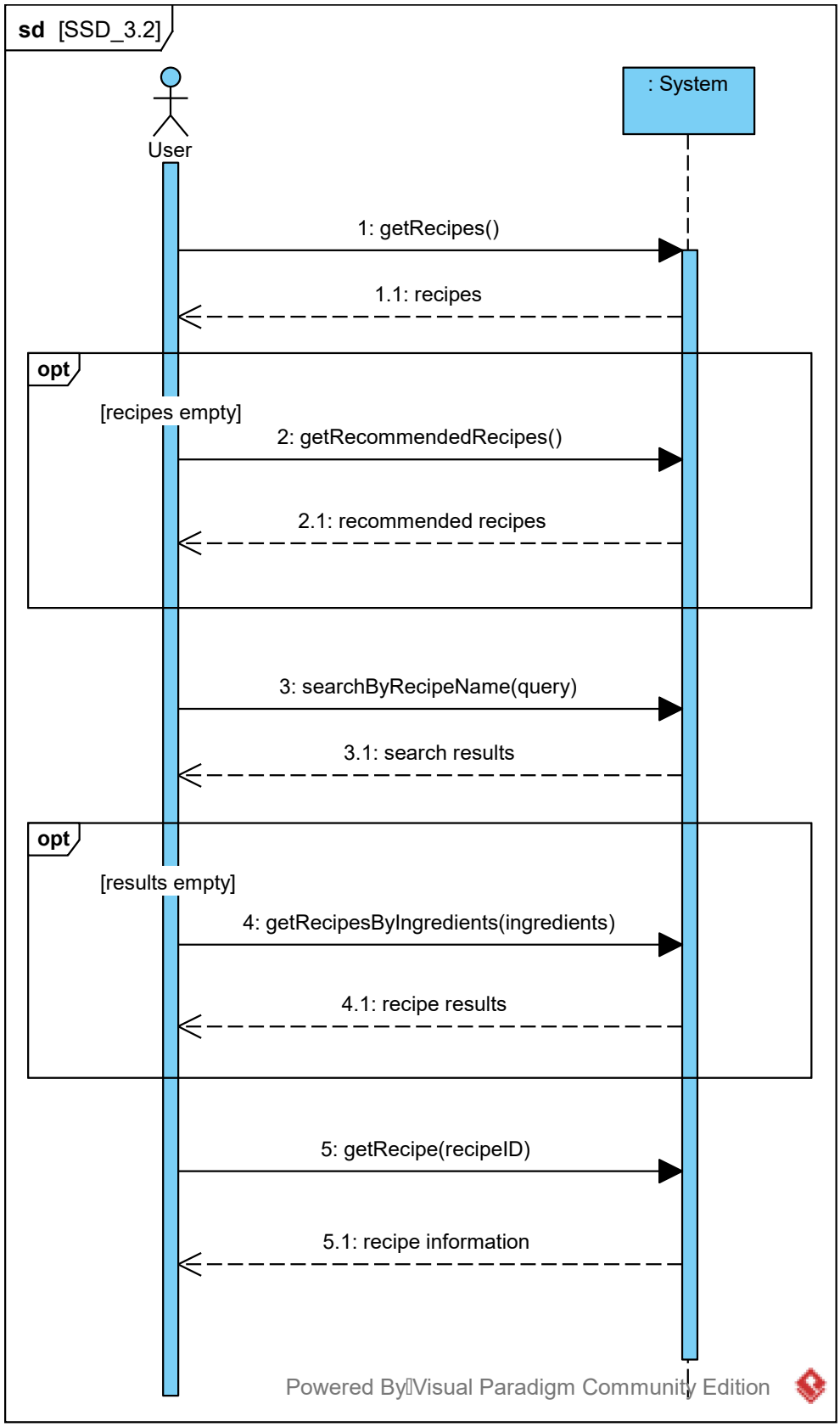

sd [SSD_6.1]
User
: System
1: getFoods()
1.1: foods
loop
[not satisfied]
alt
[add]
2: addFood(name, category, nutrition)
2.1: food
[delete]
3: removeFood(id)
[edit]
4: editFood(id, name, category, nutrition)
4.1: food
Powered By Visual Paradigm Community Edition

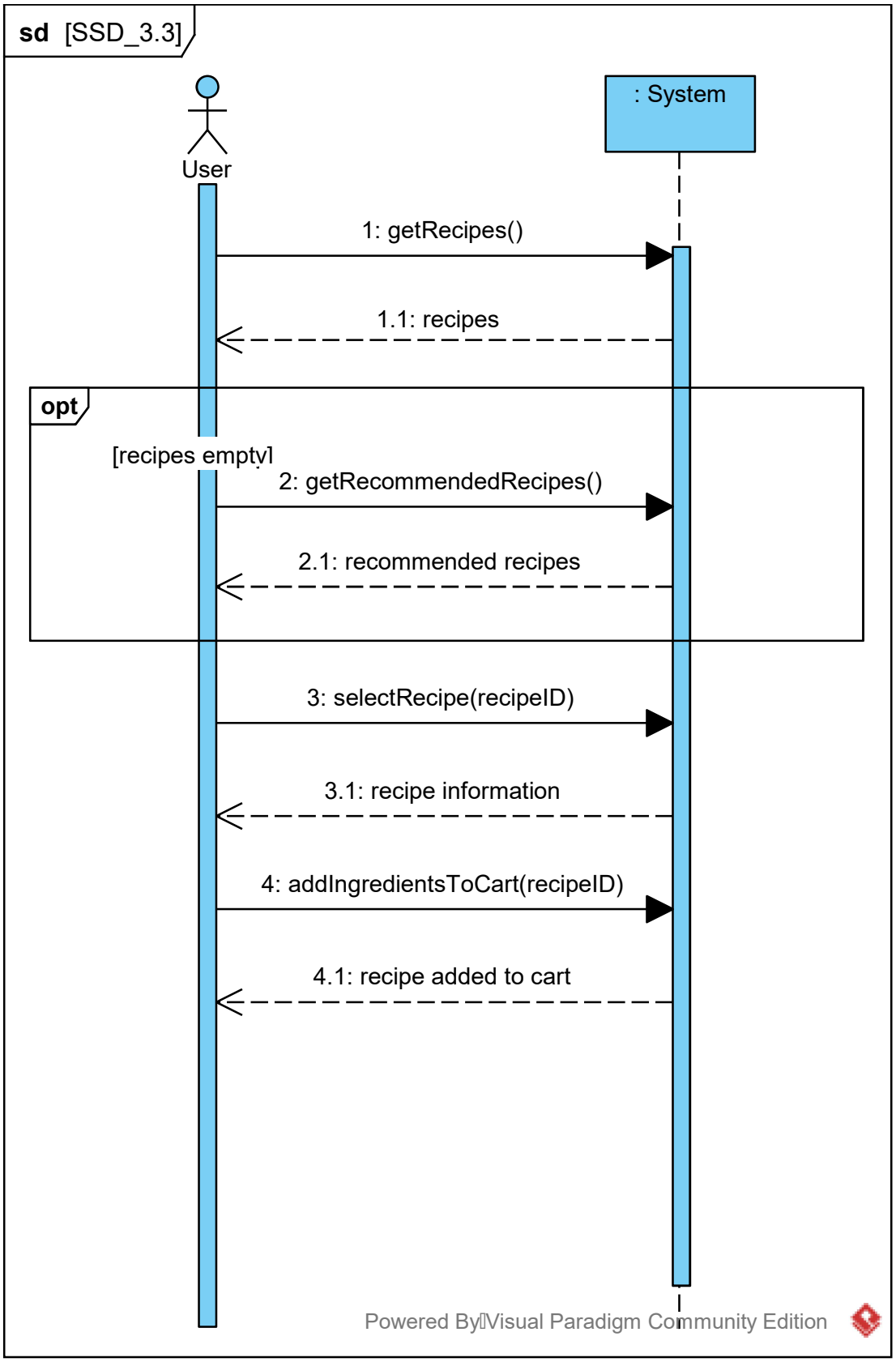

sd [UC 3.4]
: User
: System
1: getRecipes()
1.1: display recipes to user
alt
[no recipes]
loop
[want to create recipe]
2: addRecipe(name, ingredients, servings)
2.1: recipe creation confirmation
3: editRecipe(recipeID)
3.1: display recipe specifications
opt
[save]
opt
[invalid input]
3.2: Inform User of Invalid Input
4: editRecipe(recipeID)
5: saveChanges()
5.1: changes saved confirmation
5.2: Return To Recipes
Powered By Visual Paradigm Community Edition

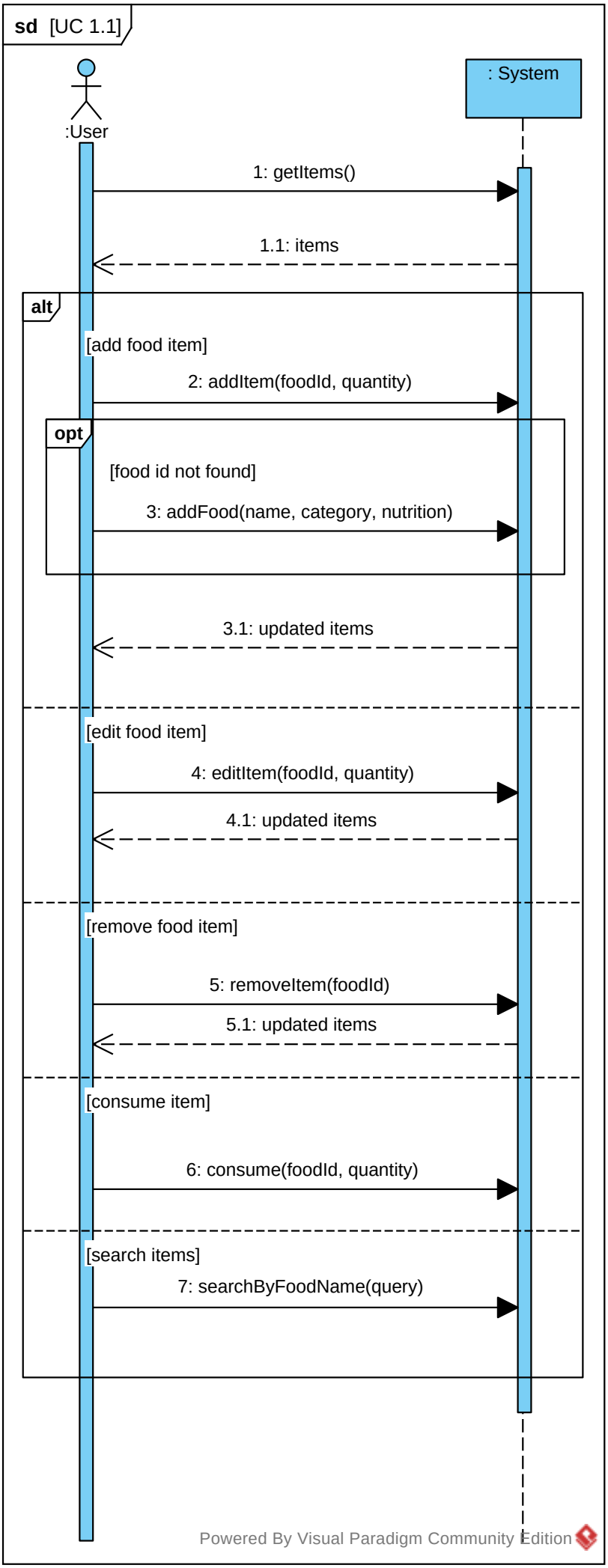

sd [UC 3.5]
: User
: System
1: viewRecipes()
1.1: Display Recipes
2: getRecommendedRecipes()
2.1: Display Recommended Recipes
opt
[view recipe]
3: getRecipe(recipeID)
3.1: Display Recipe
Powered By Visual Paradigm Community Edition

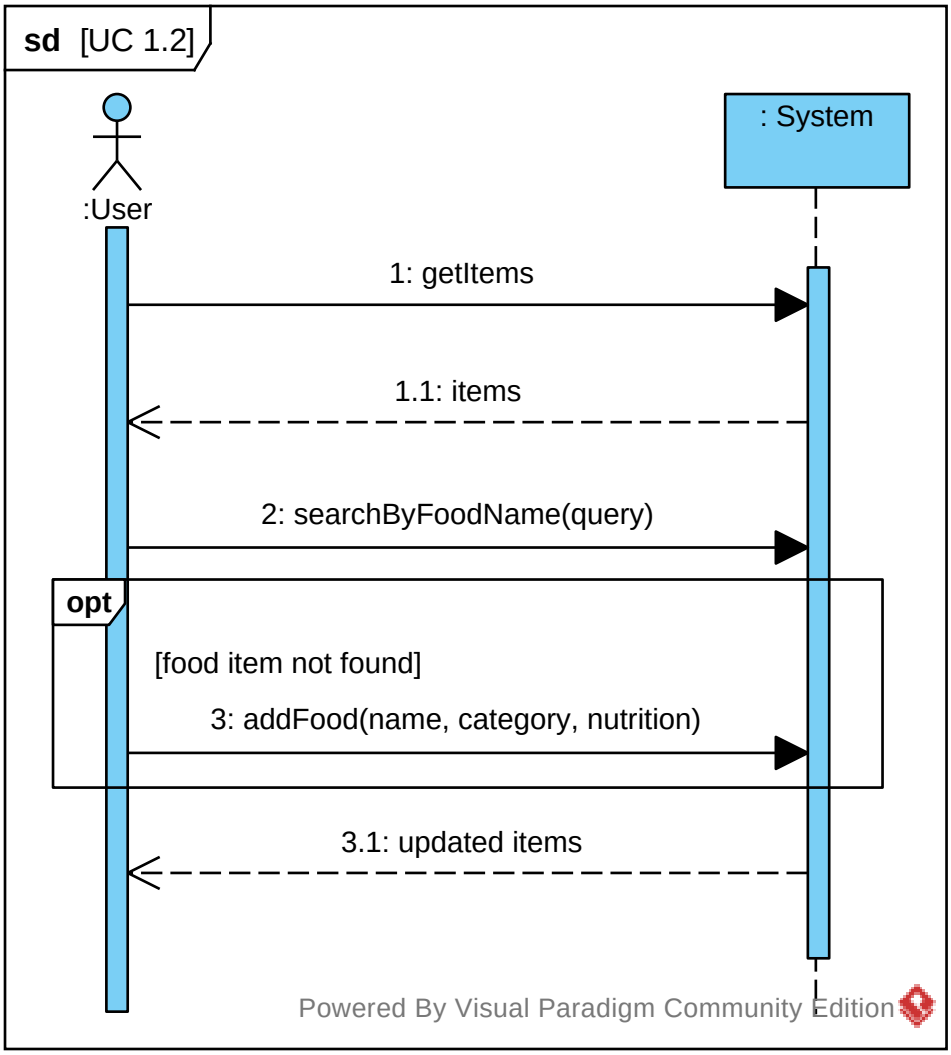

sd [UC 6.2]
: User
: System
1: getItems()
1.1: Display Food Types
2: addFood(name, category, nutrition)
opt
[food type already exists]
loop
[fix input]
2.1: Inform User of Invalid Input
3: addFood(name, category, nutrition)
4: saveNew(food)
4.1: confirm save and return to food types
Powered By Visual Paradigm Community Edition

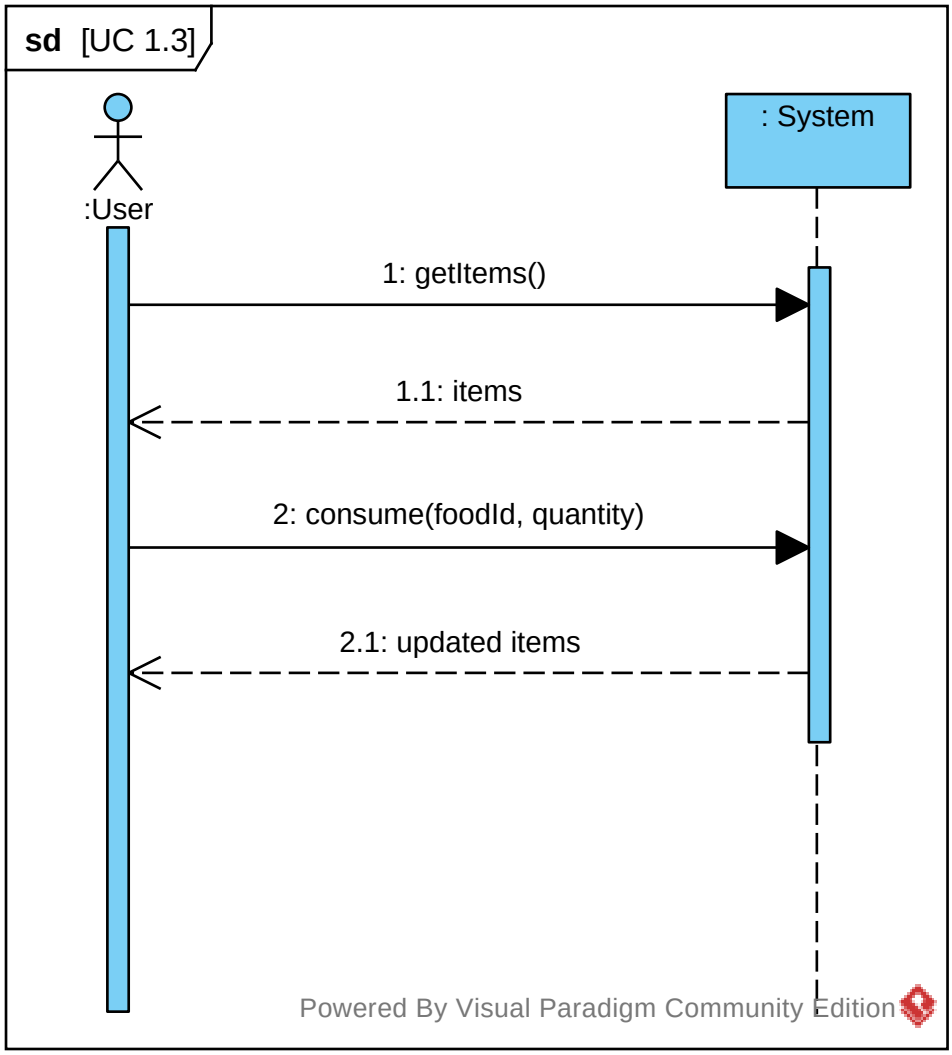

sd [SSD_3.1]
User
: System
1: getRecipes()
1.1: recipes
opt
[recipes empty]
2: getRecommendedRecipes()
2.1: recommended recipes
loop
[missing information]
3: addRecipe(name, ingredients, servings)
3.1: form response
Powered By Visual Paradigm Community Edition

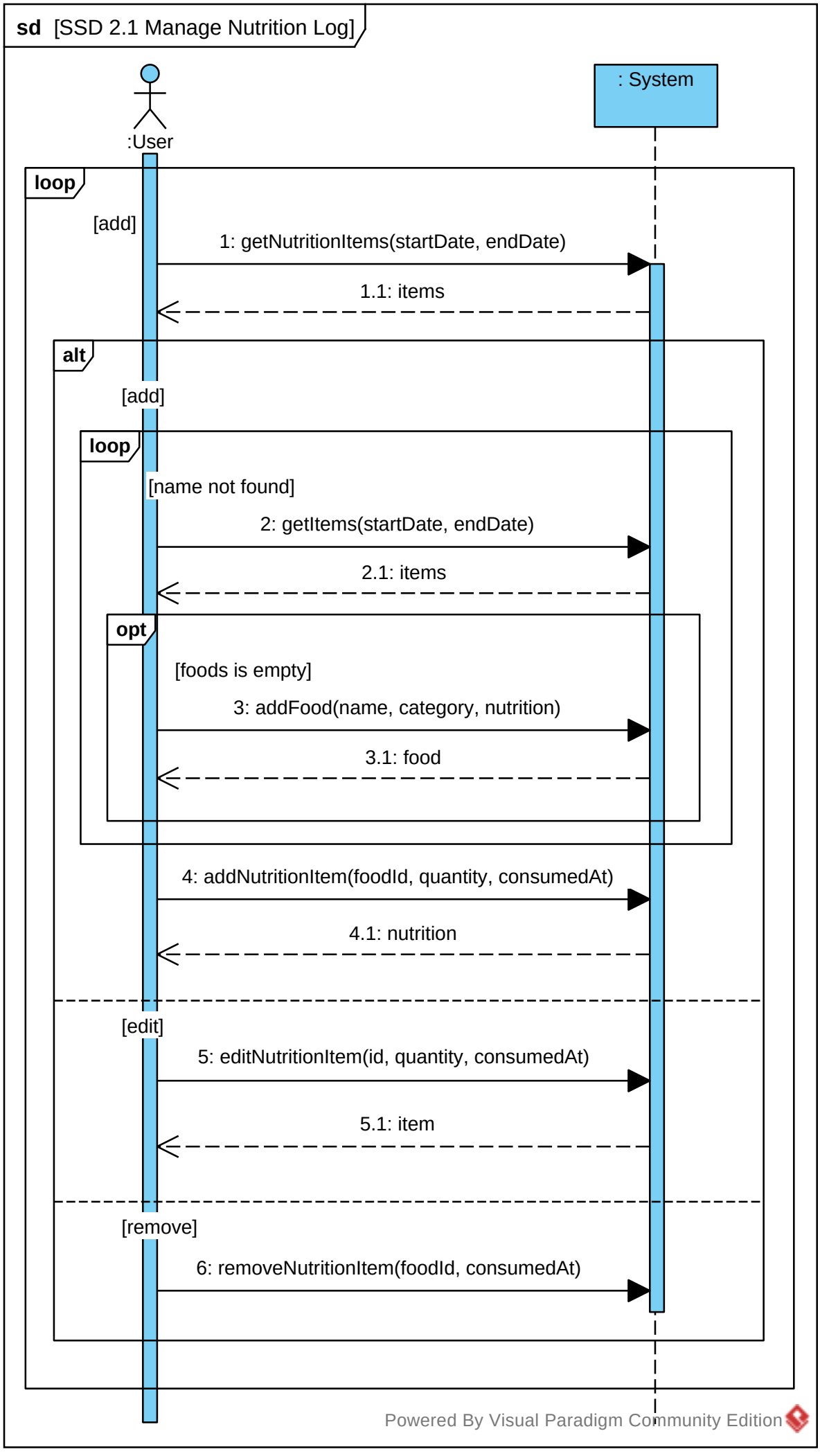

sd [SSD_3.2]
User
: System
1: getRecipes()
1.1: recipes
opt
[recipes empty]
2: getRecommendedRecipes()
2.1: recommended recipes
3: searchByRecipeName(query)
3.1: search results
opt
[results empty]
4: getRecipesByIngredients(ingredients)
4.1: recipe results
5: getRecipe(recipeID)
5.1: recipe information
Powered By Visual Paradigm Community Edition

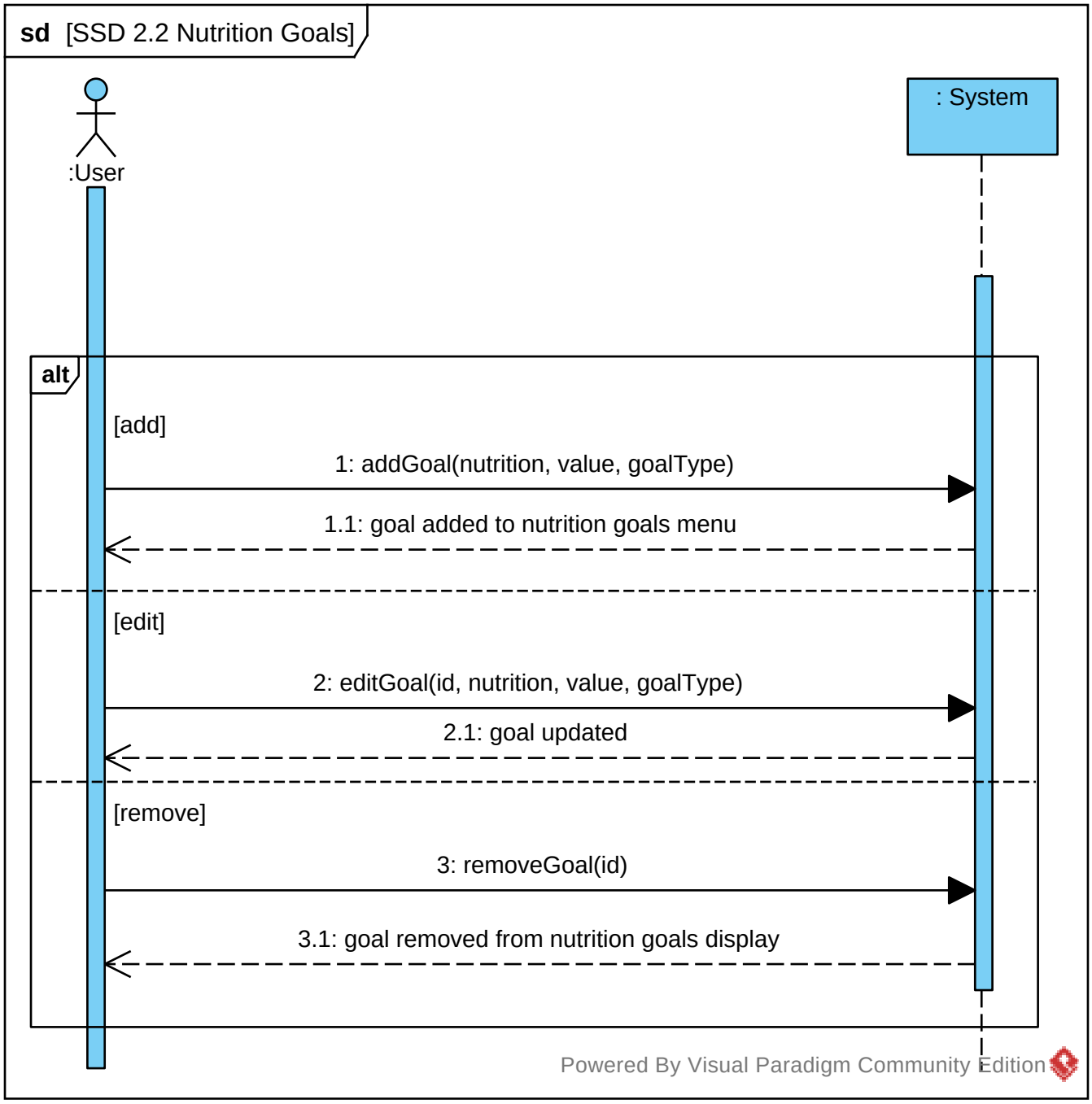

sd [SSD_3.3]
User
: System
1: getRecipes()
1.1: recipes
opt
[recipes empty]
2: getRecommendedRecipes()
2.1: recommended recipes
3: selectRecipe(recipeID)
3.1: recipe information
4: addIngredientsToCart(recipeID)
4.1: recipe added to cart
Powered By Visual Paradigm Community Edition

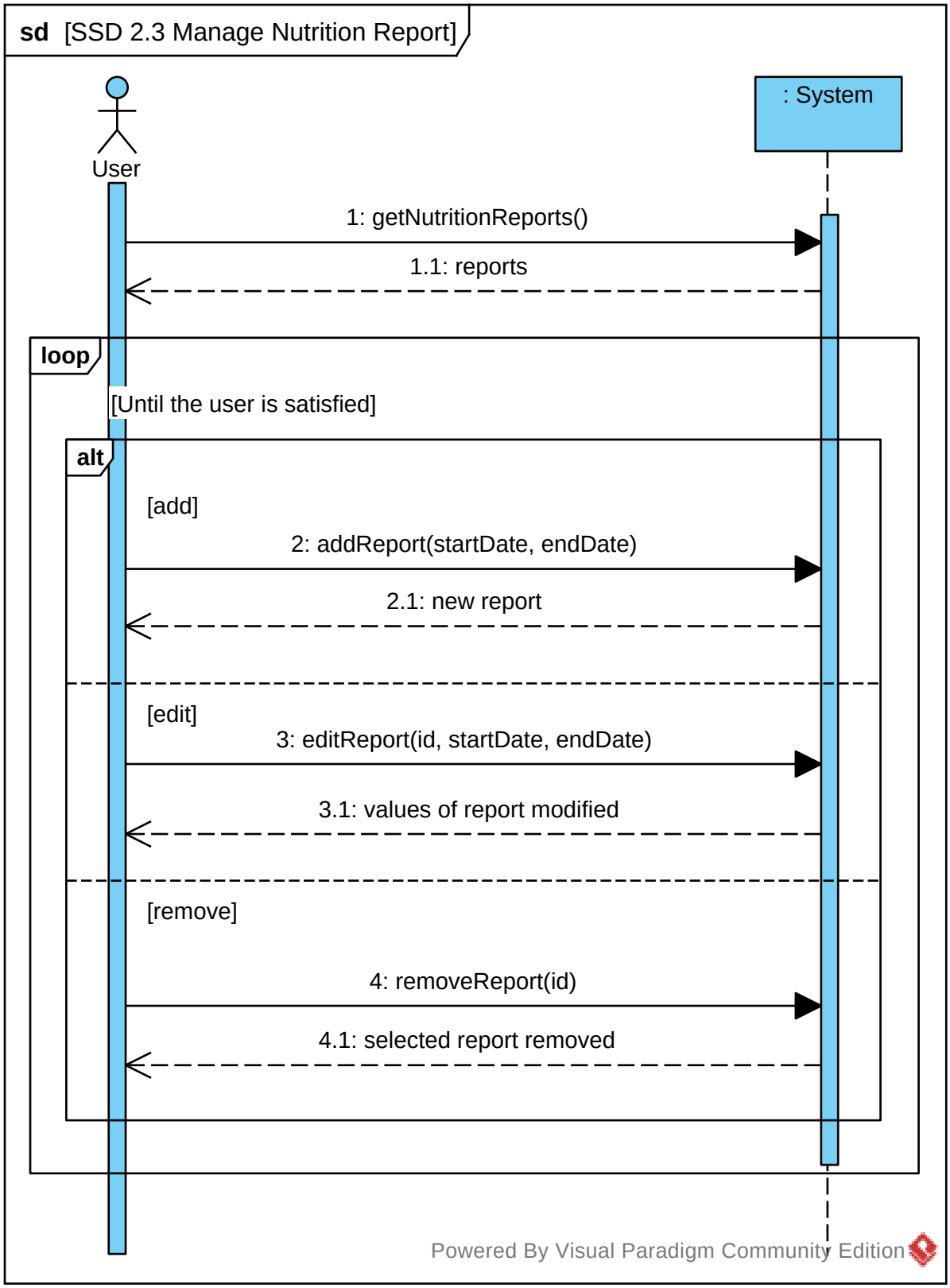

sd [UC 1.1]
:User
: System
1: getItems()
1.1: items
alt
[add food item]
2: addItem(foodId, quantity)
opt
[food id not found]
3: addFood(name, category, nutrition)
3.1: updated items
[edit food item]
4: editItem(foodId, quantity)
4.1: updated items
[remove food item]
5: removeItem(foodId)
5.1: updated items
[consume item]
6: consume(foodId, quantity)
[search items]
7: searchByFoodName(query)
Powered By Visual Paradigm Community Edition

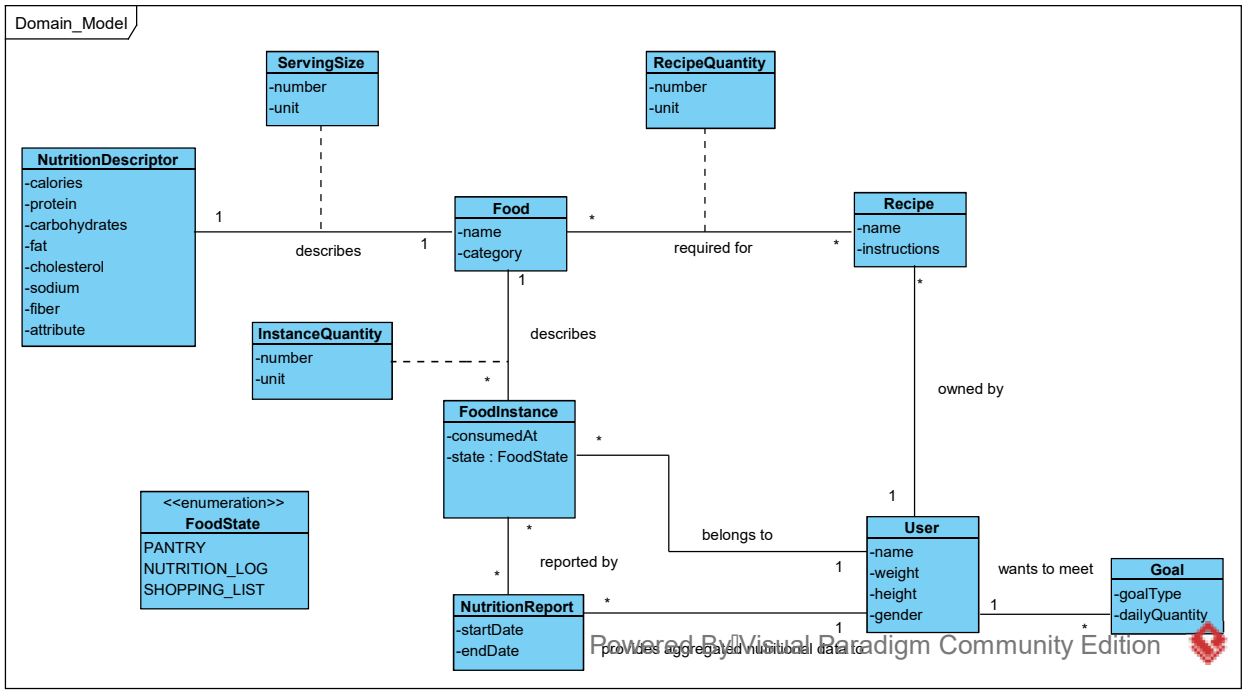

sd [UC 1.2]
:User
: System
1: getItems
1.1: items
2: searchByFoodName(query)
opt
[food item not found]
3: addFood(name, category, nutrition)
3.1: updated items
Powered By Visual Paradigm Community Edition


sd [UC 1.3]
:User
: System
1: getItems()
1.1: items
2: consume(foodId, quantity)
2.1: updated items
Powered By Visual Paradigm Community Edition


sd [SSD 2.1 Manage Nutrition Log]
:User
: System
loop
[add]
1: getNutritionItems(startDate, endDate)
1.1: items
alt
[add]
loop
[name not found]
2: getItems(startDate, endDate)
2.1: items
opt
[foods is empty]
3: addFood(name, category, nutrition)
3.1: food
4: addNutritionItem(foodId, quantity, consumedAt)
4.1: nutrition
[edit]
5: editNutritionItem(id, quantity, consumedAt)
5.1: item
[remove]
6: removeNutritionItem(foodId, consumedAt)
Powered By Visual Paradigm Community Edition


sd [SSD 2.2 Nutrition Goals]
:User
: System
alt
[add]
1: addGoal(nutrition, value, goalType)
1.1: goal added to nutrition goals menu
[edit]
2: editGoal(id, nutrition, value, goalType)
2.1: goal updated
[remove]
3: removeGoal(id)
3.1: goal removed from nutrition goals display
Powered By Visual Paradigm Community Edition


sd [SSD 2.3 Manage Nutrition Report]
User
: System
1: getNutritionReports()
1.1: reports
loop
[Until the user is satisfied]
alt
[add]
2: addReport(startDate, endDate)
2.1: new report
[edit]
3: editReport(id, startDate, endDate)
3.1: values of report modified
[remove]
4: removeReport(id)
4.1: selected report removed
Powered By Visual Paradigm Community Edition


Domain_Model
ServingSize
-number
-unit
RecipeQuantity
-number
-unit
NutritionDescriptor
-calories
-protein
-carbohydrates
-fat
-cholesterol
-sodium
-fiber
-attribute
Food
-name
-category
Recipe
-name
-instructions
describes
required for
InstanceQuantity
-number
-unit
describes
owned by
FoodInstance
-consumedAt
-state : FoodState
<<enumeration>>
FoodState
PANTRY
NUTRITION_LOG
SHOPPING_LIST
belongs to
reported by
User
-name
-weight
-height
-gender
wants to meet
Goal
-goalType
-dailyQuantity
NutritionReport
-startDate
-endDate
provides aggregated nutritional data to
Powered By Visual Paradigm Community Edition

Use Case Name	Use Case ID	REQ 1	REQ 2	REQ 3	REQ 4	REQ 5	REQ 6	REQ 7	REQ 8	REQ 9	REQ 10
Manage Pantry	1.1	X						X	X		X
Search Pantry	1.2	X							X		X
Consume Pantry Item	1.3	X							X		X
Manage Nutrition Log	2.1		X					X	X		X
Nutrition Goals	2.2		X						X	X	X
Nutritional Report	2.3		X						X		X
Create Recipe	3.1			X			X	X	X	X	
View Recipes	3.2			X				X	X		
Add Recipe Items to Shopping List	3.3			X			X		X		X
Modify Recipe	3.4			X					X	X	
Recommend Recipe	3.5			X				X	X		X
Cook Recipe	3.6	X		X					X		X
Manage Shopping List	4.1				X		X	X	X		X
Export Shopping List	4.2				X				X		X
Mark Purchased Items	4.3				X				X		X
Startup Tutorial	5.1	X	X	X	X	X	X	X	X	X	X
Manage Food Types	6.1						X	X	X	X	
Create New Food Type	6.2						X			X	

FoodPants-Gantt

Feb 23, 2022

http://

Boothverse

Project manager	Austin Huizinga
Project dates	Jan 18, 2022 - Apr 28, 2022
Completion	0%
Tasks	28
Resources	4

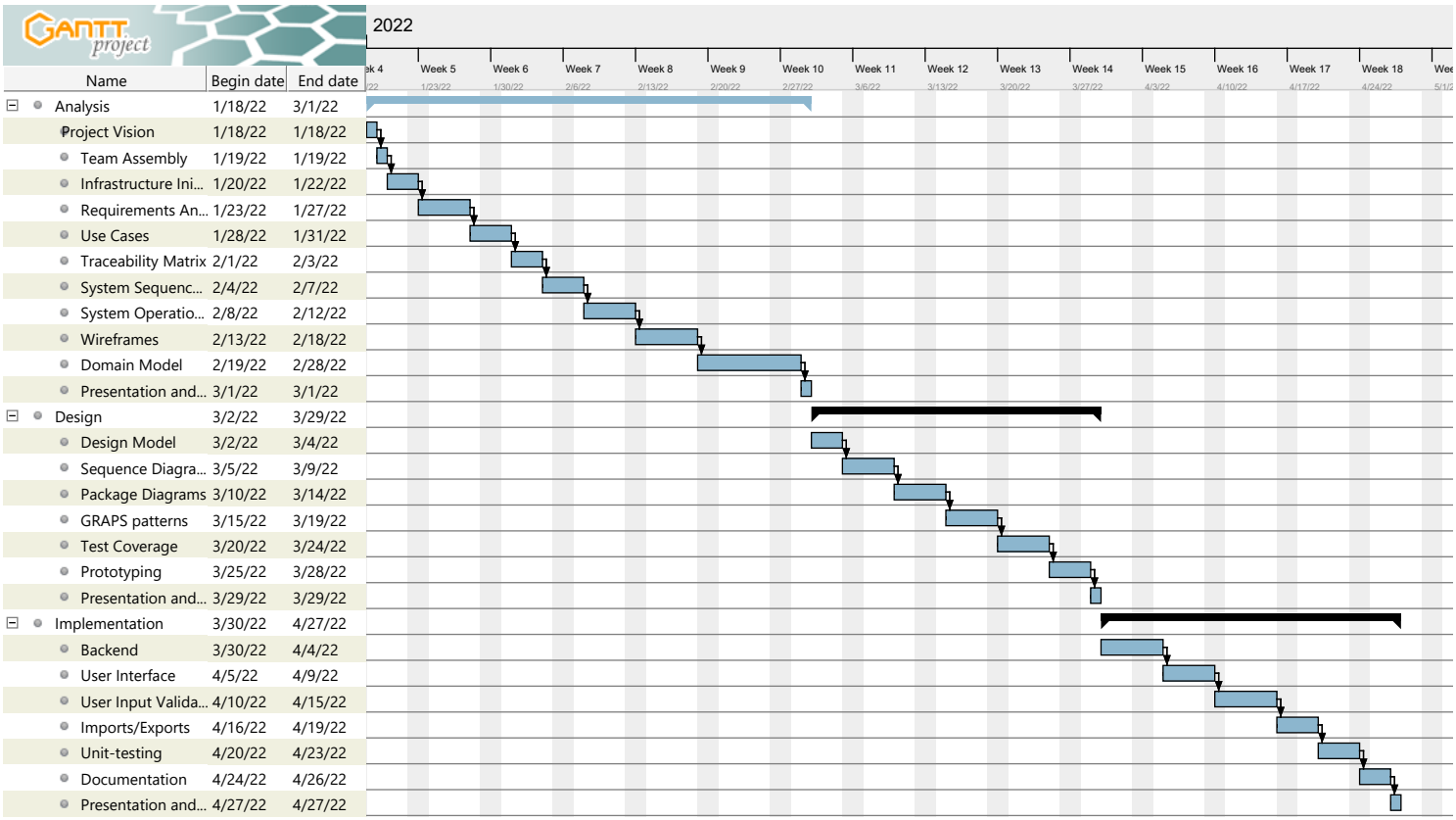
Tasks

Name	Begin date	End date
Analysis	1/18/22	3/1/22
Project Vision	1/18/22	1/18/22
Team Assembly	1/19/22	1/19/22
Infrastructure Initialization	1/20/22	1/22/22
Requirements Analysis	1/23/22	1/27/22
Use Cases	1/28/22	1/31/22
Traceability Matrix	2/1/22	2/3/22
System Sequence Diagrams	2/4/22	2/7/22
System Operations	2/8/22	2/12/22
Wireframes	2/13/22	2/18/22
Domain Model	2/19/22	2/28/22
Presentation and Reporting	3/1/22	3/1/22
Design	3/2/22	3/29/22
Design Model	3/2/22	3/4/22
Sequence Diagrams	3/5/22	3/9/22
Package Diagrams	3/10/22	3/14/22
GRAPS patterns	3/15/22	3/19/22
Test Coverage	3/20/22	3/24/22
Prototyping	3/25/22	3/28/22
Presentation and Reporting	3/29/22	3/29/22
Implementation	3/30/22	4/27/22
Backend	3/30/22	4/4/22
User Interface	4/5/22	4/9/22
User Input Validation	4/10/22	4/15/22
Imports/Exports	4/16/22	4/19/22
Unit-testing	4/20/22	4/23/22
Documentation	4/24/22	4/26/22
Presentation and Reporting	4/27/22	4/27/22

Resources

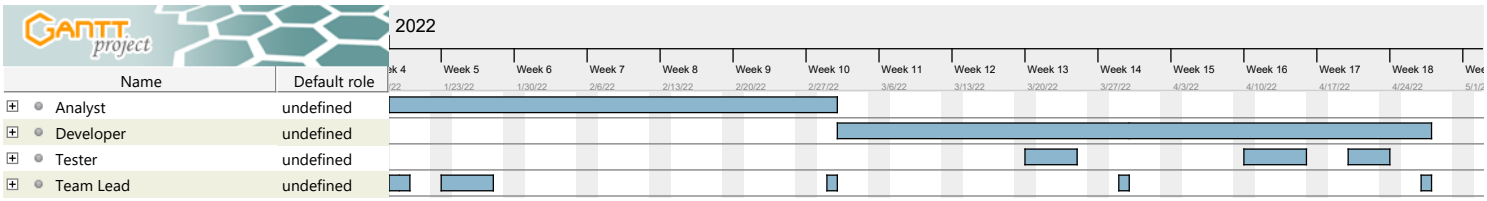
Name	Default role
Analyst	undefined
Developer	undefined
Tester	undefined
Team Lead	undefined

Gantt Chart



Name	Begin date	End date
Analysis	1/18/22	3/1/22
Project Vision	1/18/22	1/18/22
Team Assembly	1/19/22	1/19/22
Infrastructure Ini...	1/20/22	1/22/22
Requirements An...	1/23/22	1/27/22
Use Cases	1/28/22	1/31/22
Traceability Matrix	2/1/22	2/3/22
System Sequenc...	2/4/22	2/7/22
System Operatio...	2/8/22	2/12/22
Wireframes	2/13/22	2/18/22
Domain Model	2/19/22	2/28/22
Presentation and...	3/1/22	3/1/22
Design	3/2/22	3/29/22
Design Model	3/2/22	3/4/22
Sequence Diagra...	3/5/22	3/9/22
Package Diagrams	3/10/22	3/14/22
GRAPS patterns	3/15/22	3/19/22
Test Coverage	3/20/22	3/24/22
Prototyping	3/25/22	3/28/22
Presentation and...	3/29/22	3/29/22
Implementation	3/30/22	4/27/22
Backend	3/30/22	4/4/22
User Interface	4/5/22	4/9/22
User Input Valida...	4/10/22	4/15/22
Imports/Exports	4/16/22	4/19/22
Unit-testing	4/20/22	4/23/22
Documentation	4/24/22	4/26/22
Presentation and...	4/27/22	4/27/22

Resources Chart



Name	Default role
Analyst	undefined
Developer	undefined
Tester	undefined
Team Lead	undefined

STARTUP CONTRACTS

Contract CO1: register

Operation:	register(name, info)
Cross References:	Use Cases: 5.1 Startup
Preconditions:	● Application is running
Postconditions:	● A user instance has been instantiated ● User attributes have been initialized (name, opt. weight, gender) ● User data has been saved to the database

PANTRY CONTRACTS

Contract CO2: getPantryItems

Operation:	getPantryItems()
Cross References:	Use Cases: 5.1 Startup, 1.1 Manage Pantry, 1.2 Search Pantry, 1.3 Consume Pantry Item
Preconditions:	● User has been registered within the system and initialized ● Data has been loaded from the database
Postconditions:	● System provides pantry data to user

Contract CO3: addPantryItem

Operation:	addPantryItem(foodId, quantity)
Cross References:	Use Cases: 5.1 Startup, 1.1 Manage Pantry
Preconditions:	● Data has been loaded from the database ● Food already exists within the food database
Postconditions:	● Added food item exists within the user's pantry ● Pantry quantity of food item has been incremented to quantity + old quantity ● Pantry database updated

Contract CO4: editPantryItem

Operation:	editPantryItem(foodId, quantity)
Cross References:	Use Cases: 1.1 Manage Pantry
Preconditions:	● Data has been loaded from the database ● Food exists within the food database
Postconditions:	● Food item is updated with new quantity ● Pantry database is updated

Contract CO5: removePantryItem

Operation:	removePantryItem(foodId)
Cross References:	Use Cases: 5.1 Startup, 1.1 Manage Pantry
Preconditions:	● Data has been loaded from the database ● An item of the given food type exists within the pantry
Postconditions:	● Food item with given name has its instance removed from pantry ● Pantry database is updated

Contract CO6: searchPantryByFoodName

Operation:	searchPantryByFoodName(query)
Cross References:	Use Cases: 1.2 Search Pantry
Preconditions:	● Data has been loaded from the database
Postconditions:	● List of food items starting with item name are returned or an empty list if no pantry items match the itemName

Contract CO7: consume

Operation:	consume(foodId, quantity)
Cross References:	Use Cases: 1.3 Consume Pantry Item
Preconditions:	● Data has been loaded from the database ● FoodItem passed in exists in the pantry
Postconditions:	● Food in pantry with type foodItem has its quantity decremented ● If new quantity is 0, foodItem is removed from pantry ● Database is updated

SHOPPING LIST CONTRACTS

Contract CO8: getShoppingItems

Operation:	getShoppingItems()
Cross References:	Use Cases: 4.1 Manage Shopping List, 4.3 Mark Purchased Items
Preconditions:	• Data has been loaded from the database
Postconditions:	• System provides user with their shopping list

Contract CO9: addShoppingItem

Operation:	addShoppingItem(foodId, quantity)
Cross References:	Use Cases: 4.1 Manage Shopping List
Preconditions:	• Data has been loaded from the database • The food type exists within the list of registered food types • The quantity is valid (non-negative, etc)
Postconditions:	• Food item with given quantity added to the shopping list • Database is updated

Contract CO10: editShoppingItem

Operation:	editShoppingItem(foodId, quantity)
Cross References:	Use Cases: 4.1 Manage Shopping List
Preconditions:	• Data has been loaded from the database • The food type exists within the shopping list • The quantity is valid (non-negative, etc)
Postconditions:	• Quantity of given food item updated within the shopping list • Database is updated

Contract CO11: removeShoppingItem

Operation:	removeShoppingItem(foodId)
Cross References:	Use Cases: 4.1 Manage Shopping List
Preconditions:	• Data has been loaded from the database • The food type exists within the shopping list • The quantity is valid (non-negative, etc)
Postconditions:	• Food item removed from shopping list • Database is updated

Contract CO12: export

Operation:	export(fileFormat, destination)
Cross References:	Use Cases: 4.1 Manage Shopping List, 4.2 Export Shopping List
Preconditions:	• Data has been loaded from the database • User has access to the given destination • Given destination is valid • File format is valid
Postconditions:	• Shopping list has been exported to desired format

Contract CO13: purchaseItems

Operation:	purchaseItems(foodIds)
Cross References:	Use Cases: 4.3 Mark Purchased Items
Preconditions:	• Data has been loaded from the database • Given foods are registered in the list of foods • User purchased given items from the shopping list
Postconditions:	• System moves given items from shopping list to pantry • Database is updated

RECIPE CONTRACTS

Contract CO14: getRecipes

Operation:	getRecipes()
Cross References:	Use Cases: 3.1 Create Recipe, 3.2 View Recipe, 3.3 Add Recipe to Shopping List, 3.4 Modify Recipe, 3.5 Recommend Recipes
Preconditions:	● Data has been loaded from the database
Postconditions:	● System provides the user's list of recipes

Contract CO15: addRecipe

Operation:	addRecipe(name, ingredients, instructions, servings)
Cross References:	Use Cases: 5.1 Startup, 3.4 Modify Recipe
Preconditions:	● Data has been loaded from the database
Postconditions:	● Ingredients have been stored has food instances within the database ● The recipe represents a new recipe instance within the recipe system ● Recipe appears within the recipe list

Contract CO16: produceCookedRecipe

Operation:	produceCookedRecipe(recipeId, isUsePantry, consumedServings, leftoverServings)
Cross References:	Use Cases: 3.6 Cook Recipe
Preconditions:	● Recipe exists within the recipe system ● Ingredients exist within the food database ● Data has been loaded from the database
Postconditions:	● User is notified of any errors preventing cooking ● Consumed portion is logged to food log ● Pantry is updated to contain leftovers ● Database is updated

Contract CO17: editRecipe

Operation:	editRecipe(recipeId, recipe)
Cross References:	Use Cases: 3.4 Modify Recipe
Preconditions:	● Data has been loaded from the database ● recipeId is valid
Postconditions:	● User is notified of any errors preventing modification ● Recipe is saved to Recipe List ● Database is updated

Contract CO18: getRecipe

Operation:	getRecipe(recipeId)
Cross References:	Use Cases: 3.5 Recommend Recipes
Preconditions:	● Data has been loaded from the database ● Recipe exists within the recipe system
Postconditions:	● Recipe is displayed to the user

Contract CO19: getRecommendRecipes

Operation:	getRecommendRecipes()
Cross References:	Use Cases: 3.5 Recommend Recipe
Preconditions:	● Data has been loaded from the database
Postconditions:	● The user is presented with all recommended recipes from the Recipe List based off of items in their pantry

Contract CO20: searchByRecipeName

Operation:	searchByRecipeName(query)
Cross References:	Use Cases: 3.2 View Recipes
Preconditions:	● Data has been loaded from the database
Postconditions:	● List of matching recipes are displayed to the user

Contract CO21: addIngredientsToCart

Operation:	addIngredientsToCart(recipeId)
Cross References:	Use Cases: 3.3 Add Recipe to Shopping Cart
Preconditions:	● Data has been loaded from the database
Postconditions:	● Recipe ingredients added to shopping cart ● Database updated

Contract CO22: addMissingIngredientsToCart

Operation:	addMissingIngredientsToCart(recipeId)
Cross References:	Use Cases: 3.3 Add Recipe to Shopping Cart
Preconditions:	● Data has been loaded from the database
Postconditions:	● Recipe ingredients not in pantry added to shopping cart ● Database updated

NUTRITION LOG CONTRACTS

Contract CO23: addNutritionItem

Operation:	addNutritionItem(foodId, quantity, consumedAt)
Cross References:	Use Cases: 3.6 Cook Recipe, 2.1 Manage Nutrition Log
Preconditions:	● Data has been loaded from the database ● Food exists within the recipe system ● Quantity is valid (non-negative, valid unit, etc.)
Postconditions:	● Food of given quantity logged to nutrition log at given time ● Nutrition log is updated with new nutrition instance ● Database is updated

Contract CO24: getNutritionItems

Operation:	getNutritionItems(startDate, endDate)
Cross References:	Use Cases: 2.1 Manage Nutrition Log
Preconditions:	● Data has been loaded from the database
Postconditions:	● System provides the list of nutrition instances from the given time window

Contract CO25: editNutritionItem

Operation:	editNutritionItem(foodId, quantity, consumedAt)
Cross References:	Use Cases: 2.1 Manage Nutrition Log
Preconditions:	● Data has been loaded from the database ● Quantity is valid (non-negative, valid unit, etc.)
Postconditions:	● The selected item has had its values changed as per user request ● Database is updated

Contract CO26: removeNutritionItem

Operation:	removeNutritionItem(foodId, consumedAt)
Cross References:	Use Cases: 2.1 Manage Nutrition Log
Preconditions:	● Data has been loaded from the database ● The given nutrition item is already in the food log
Postconditions:	● The selected item has been removed from the log ● Database is updated

Contract CO27: addGoal

Operation:	addGoal(nutrient, quantity, goalType)
Cross References:	Use Cases: 2.2 Nutrition Goals
Preconditions:	● Data has been loaded from the database ● The input values are valid and logical
Postconditions:	● A nutrition goal has been created using the given fields ● Database is updated

Contract CO28: editGoal

Operation:	editGoal(id, nutrient, quantity, goalType)
Cross References:	Use Cases: 2.2 Nutrition Goals
Preconditions:	● Data has been loaded from the database ● The input values are valid and logical
Postconditions:	● The selected goal has had its values changed as per user request ● Database is updated

Contract CO29: removeGoal

Operation:	removeGoal(id)
Cross References:	Use Cases: 2.2 Nutrition Goals
Preconditions:	● Data has been loaded from the database ● The given goal exists within the user's list of goals
Postconditions:	● The selected goal has been removed ● Database is updated

Contract CO30: addReport

Operation:	addReport(startDate, endDate)
Cross References:	Use Cases: 2.3 Manage Nutrition Report
Preconditions:	● The user is in the nutrition reports menu ● The input values are valid and logical
Postconditions:	● A nutrition report has been created using the given fields

Contract CO31: editReport

Operation:	editReport(id, startDate, endDate)
Cross References:	Use Cases: 2.3 Manage Nutrition Report
Preconditions:	● The user has selected a report to be edited ● The input values are valid and logical
Postconditions:	● A nutrition report has been edited

Contract CO32: removeReport

Operation:	deleteReport(id)
Cross References:	Use Cases: 2.3 Manage Nutrition Report
Preconditions:	● The user is in the nutrition report menu ● At least one report exists within the menu which the user wants to delete
Postconditions:	● The selected report has been removed from the report menu

FOOD CONTRACTS

Contract CO33: getFood

Operation:	getFood(id)
Cross References:	Use Cases: 6.1 Manage Food Types, 6.2 Create New Food Types, 2.1 Manage Nutrition Log
Preconditions:	● Data has been loaded from the database
Postconditions:	● System provides user with the food of the given id

Contract CO34: addFood

Operation:	addFood(name, category, nutrition), 6.2 Create New Food Types
Cross References:	Use Cases: 6.1 Manage Food Types, 4.1 Manage Shopping List, 1.1 Manage Pantry, 1.2 Search Pantry, 2.1 Manage Nutrition Log
Preconditions:	● Data has been loaded from the database ● The name is not already in use by another food ● The name is not empty ● The nutrition info is valid (non-negative macros, etc)
Postconditions:	● System adds food to list of foods ● Database is updated

Contract CO35: editFood

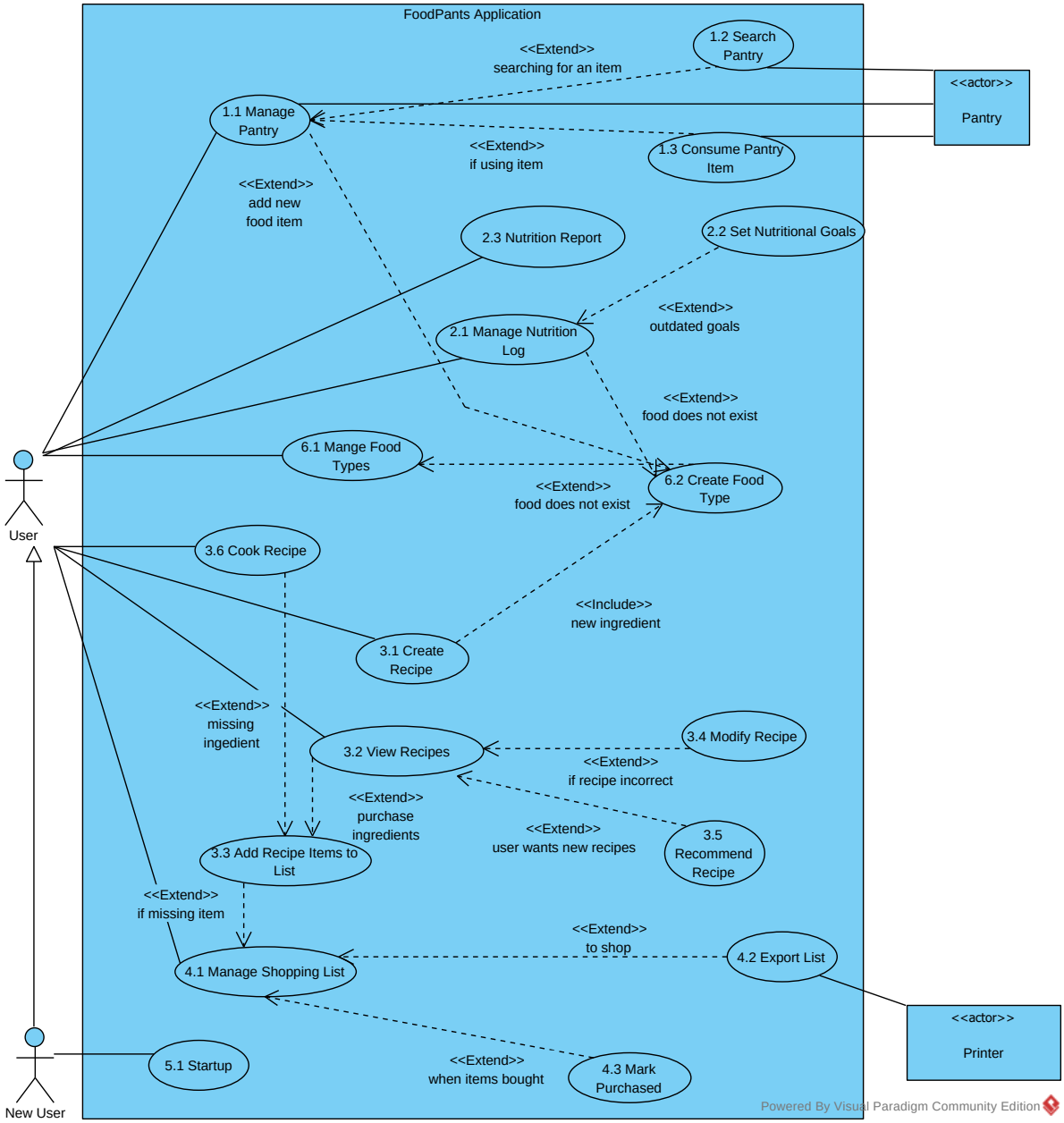
Operation:	editFood(id, name, category, nutrition)
Cross References:	Use Cases: 6.1 Manage Food Types
Preconditions:	• Data has been loaded from the database • The info is valid • The id refers to an already registered food
Postconditions:	• System updates food in list of foods • Database is updated

CO36: removeFood

Operation:	removeFood(id)
Cross References:	Use Cases: 6.1 Manage Food Types
Preconditions:	• Data has been loaded from the database • The id refers to an already registered food
Postconditions:	• System removes given food from the list of food • Database is updated

CO37: searchFoodsByName

Operation:	searchFoodsByName(query)
Cross References:	Use Cases: 4.1 Manage Shopping List
Preconditions:	• Data has been loaded from the database
Postconditions:	• List of food items starting with item name are returned or an empty list if no pantry items match the itemName


FoodPants Application
1.2 Search Pantry
<<Extend>>
searching for an item
<<actor>>
Pantry
1.1 Manage Pantry
<<Extend>>
if using item
1.3 Consume Pantry Item
<<Extend>>
add new food item
2.3 Nutrition Report
2.2 Set Nutritional Goals
<<Extend>>
outdated goals
2.1 Manage Nutrition Log
<<Extend>>
food does not exist
6.1 Mange Food Types
6.2 Create Food Type
<<Extend>>
food does not exist
User
3.6 Cook Recipe
<<Include>>
new ingredient
3.1 Create Recipe
<<Extend>>
missing ingedient
3.4 Modify Recipe
3.2 View Recipes
<<Extend>>
if recipe incorrect
<<Extend>>
purchase ingredients
<<Extend>>
user wants new recipes
3.5 Recommend Recipe
3.3 Add Recipe Items to List
<<Extend>>
if missing item
<<Extend>>
to shop
4.2 Export List
4.1 Manage Shopping List
<<actor>>
Printer
5.1 Startup
<<Extend>>
when items bought
4.3 Mark Purchased
New User
Powered By Visual Paradigm Community Edition

ACTORS

- **User:** Person interested in using the application to integrate their pantry, recipes, nutritional info, and shopping list into a single, easily-accessible experience.
- **System maintainer:** Person responsible for application execution, satisfying customer interests, and ensuring all functional requirements are met.

Iteration 1 Hours

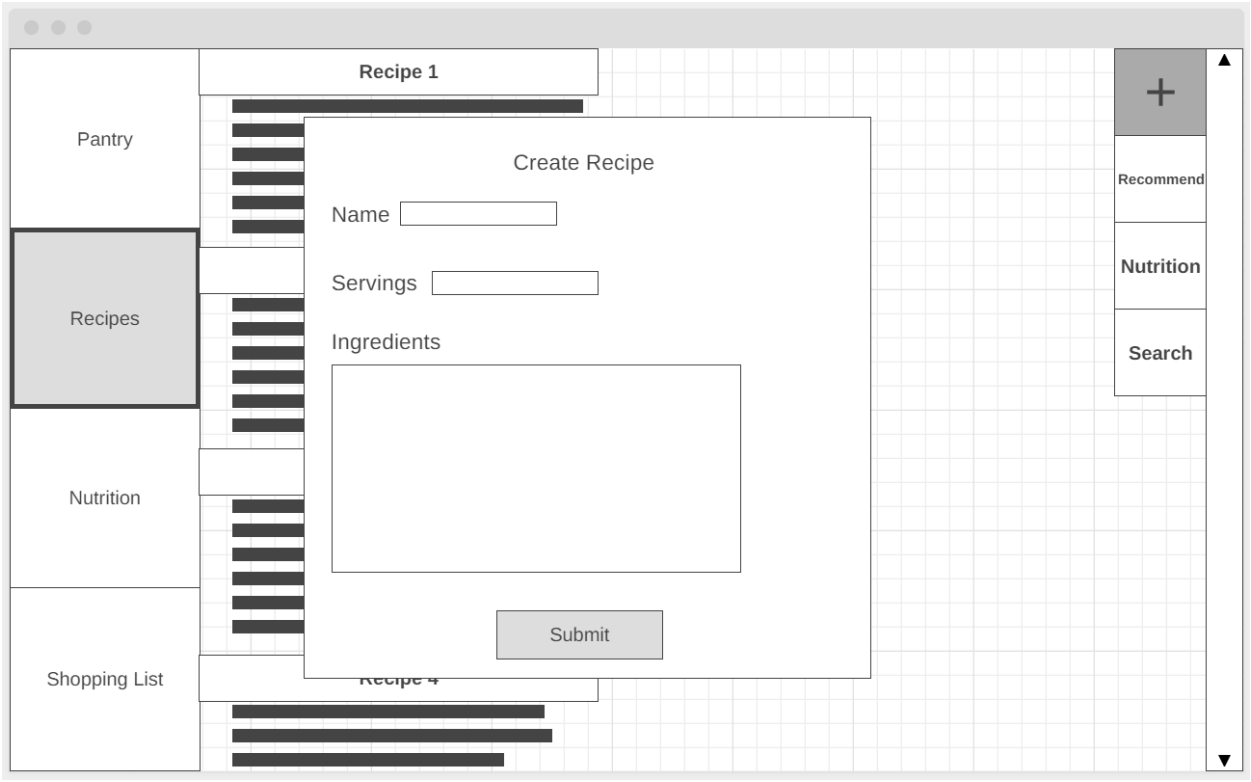
Users	Σ	03 Thu	04 Fri	05 Sat	06 Sun	07 Mon	08 Tue	09 Wed	10 Thu	11 Fri	12 Sat	13 Sun	14 Mon	15 Tue	16 Wed	17 Thu	18 Fri	19 Sat	20 Sun	21 Mon	22 Tue	23 Wed
Austin_Huizinga1	18h 44m	30m			30m		30m					2h 0m					4h 0m	1h 30m	3h 0m		1h 30m	5h 14m
Daniel Luper	12h 2m																		8h 56m	40m	1h 0m	1h 24m
Kurt_Wokoek1	13h 10m						30m					2h 0m						3h 19m	2h 35m			4h 46m
Patrick_Harris3	13h 30m						30m					1h 30m				2h 30m	2h 50m			2h 30m		3h 40m
PJ_Wallace1	15h 0m						30m				1h 30m	2h 0m						5h 30m		4h 0m	30m	1h 0m
Luka_Lelovic1	13h 15m						1h 0m					1h 30m									2h 0m	8h 45m
		30m	0m	0m	30m	0m	3h 0m	0m	0m	0m	1h 30m	9h 0m	0m	0m	0m	2h 30m	6h 50m	10h 19m	14h 31m	7h 10m	5h 0m	24h 50m

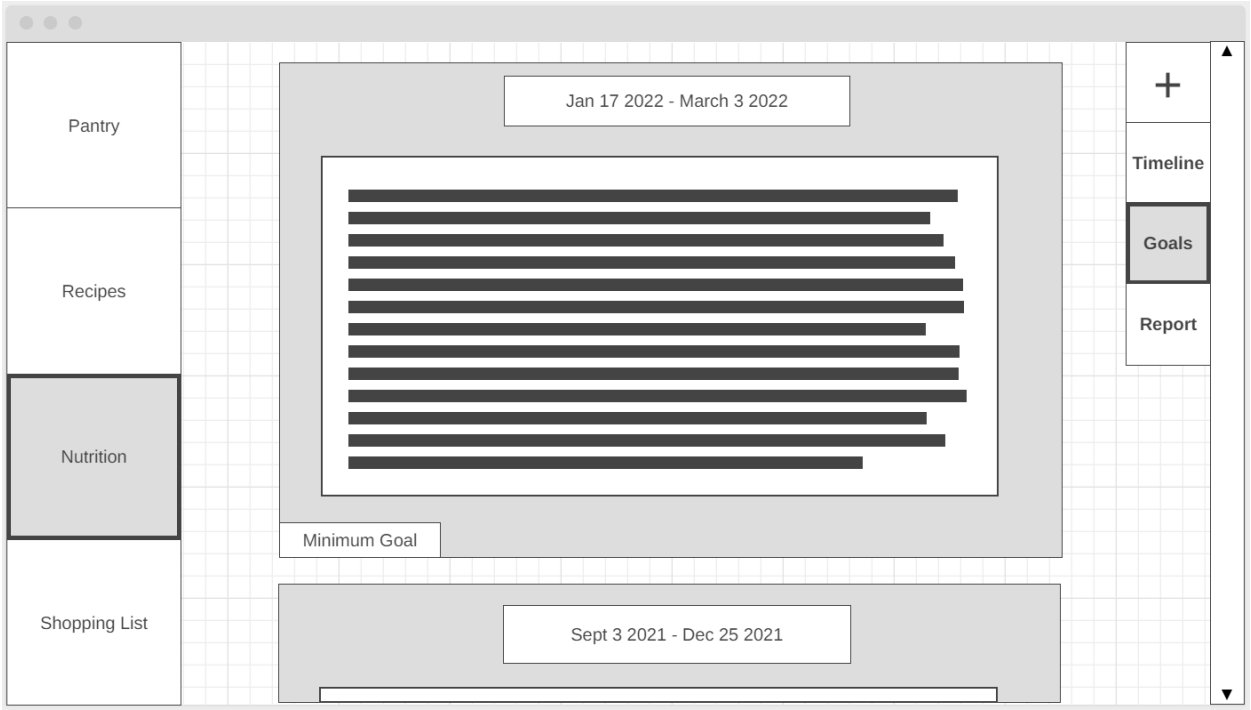
Project Vision

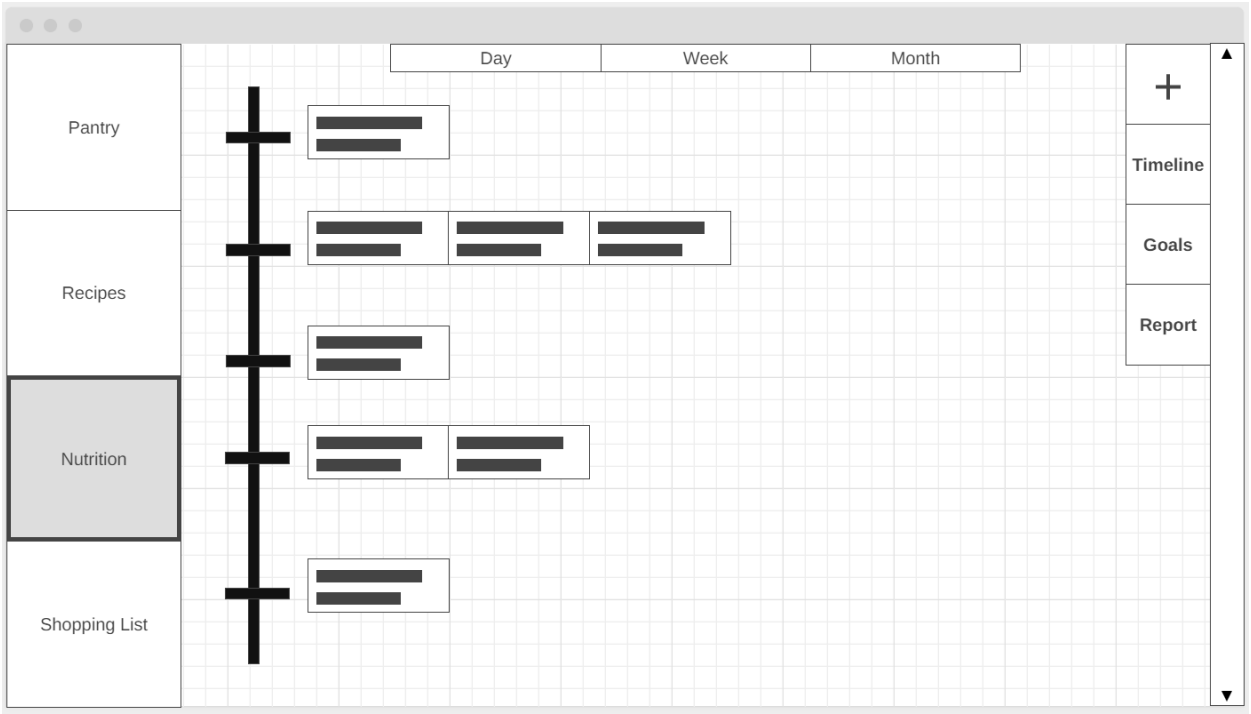
Our team aims to integrate your pantry, recipes, nutritional info, and shopping list into a single application for the ultimate culinary experience.

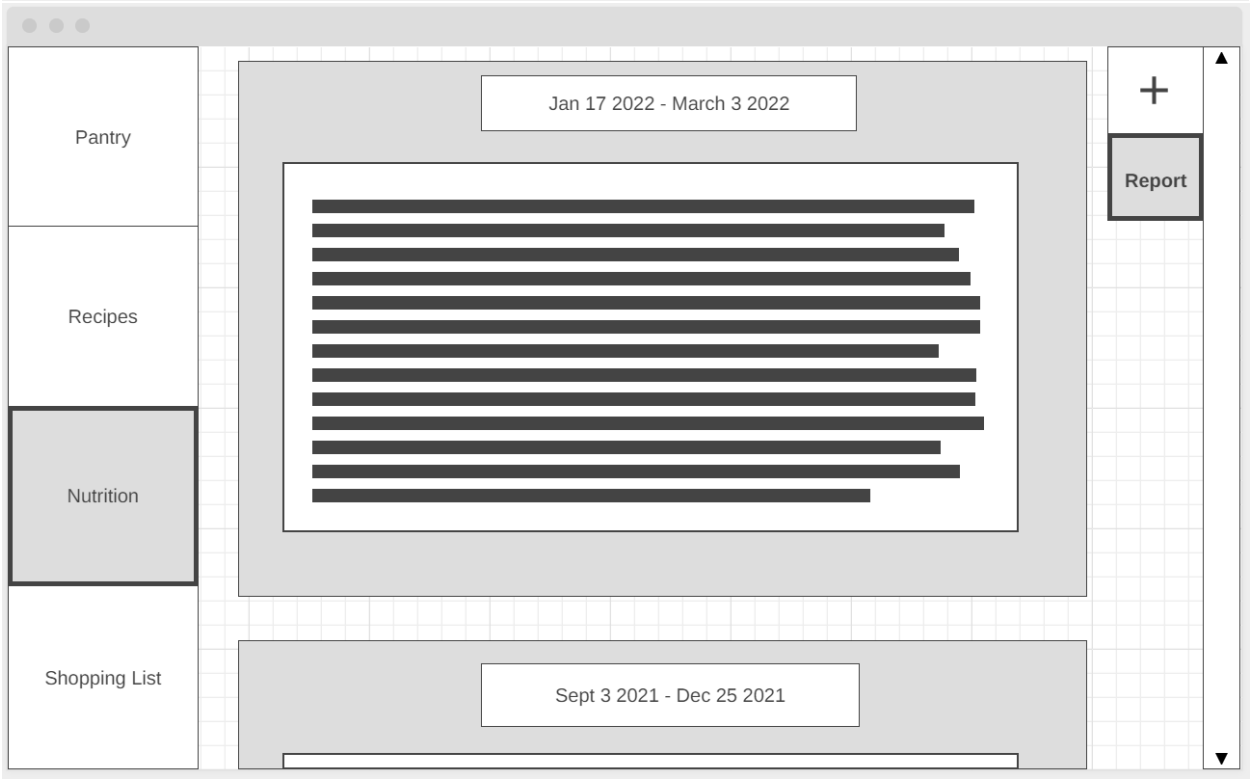
Many applications currently encompass some of these features, but we aim to provide the entire toolset for digitizing your kitchen.

Wireframes


Pantry
Recipes
Nutrition
Shopping List
Recipe 1
Create Recipe
Name
Servings
Ingredients
Submit
+
Recommend
Nutrition
Search


Pantry
Recipes
Nutrition
Shopping List
Jan 17 2022 - March 3 2022
Minimum Goal
Sept 3 2021 - Dec 25 2021
+
Timeline
Goals
Report

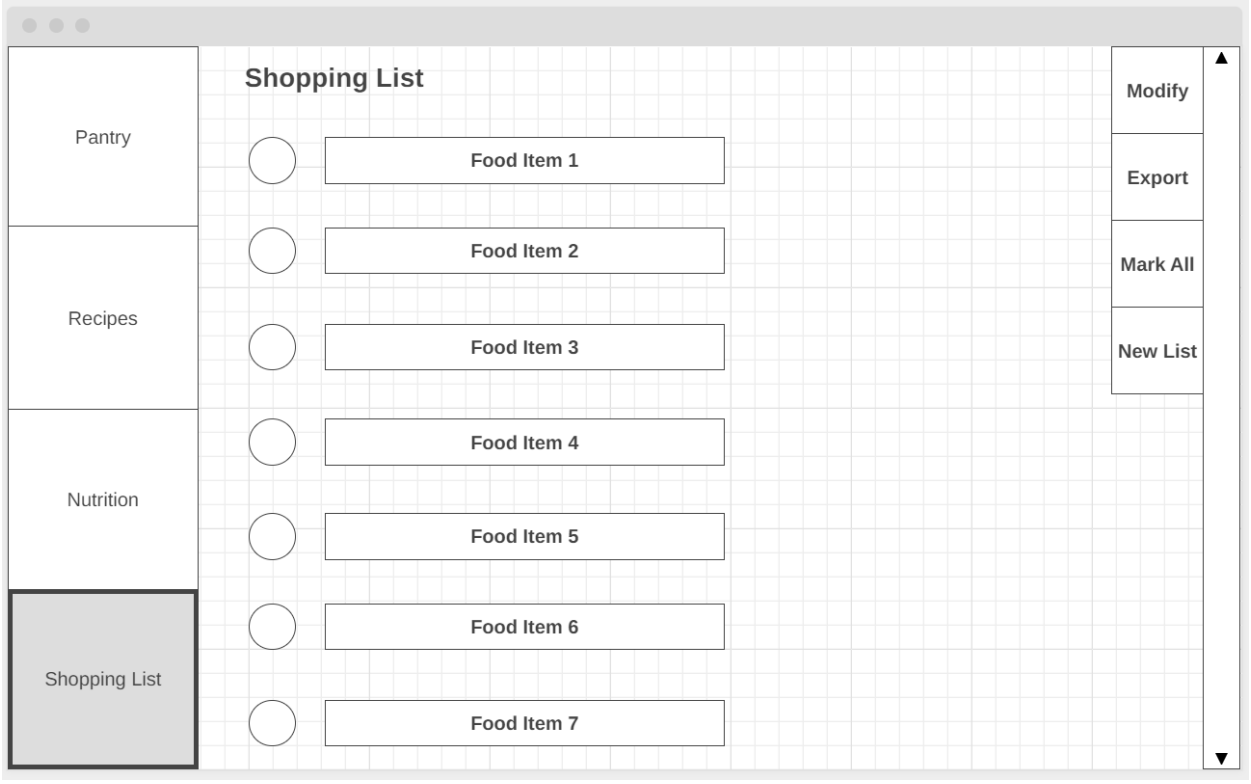

Day
Week
Month
Pantry
Recipes
Nutrition
Shopping List
Timeline
Goals
Report


Pantry
Recipes
Nutrition
Shopping List
Jan 17 2022 - March 3 2022
Sept 3 2021 - Dec 25 2021
Report


Pantry
Recipes
Nutrition
Shopping List
Edit Delete
+
Search
Modify


Pantry
Recipes
Nutrition
Shopping List
+
Search
Modify


Pantry
Recipes
Nutrition
Shopping List
Recipe 1
Recipe 2
Recipe 3
Recipe 4
+
Recommend
Nutrition
Search


Pantry
Recipes
Nutrition
Shopping List
Shopping List
Food Item 1
Food Item 2
Food Item 3
Food Item 4
Food Item 5
Food Item 6
Food Item 7
Modify
Export
Mark All
New List