Functional Requirements

Cafeteria Management System: Resolve Solution Partners (Pty) Limited

Client: Gareth Botha and Jaco Pieterse

T-RISE

Rendani Dau (13381467)

Elana Kuun (12029522)

Semaka Malapane (13081129)

Antonia Michael (13014171)

Isabel Nel (13070305)

https://github.com/toniamichael94/MainProjectCOS301

October 29, 2015

Contents

Document Title	Functional Requirements Document
Document Identification	Document 0.0.5
Author	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael, Isabel Nel
Version	0.0.7
Document Status	Seventh Version - Added the statistics, auditing, myHistory and reporting

1 Introduction

This document contains the functional requirements specification, architecture requirements and testing for the Resolve Cafeteria Management System that will be created for Software Engineering (COS 301) at the University of Pretoria 2015, by the group T-RISE. In this document we will thoroughly discuss and layout the project's functional requirements to provide a clear view of the system as a whole. An agile approach is being followed which involves an interactive and incremental method of managing and designing the system as described by the scrum methodology.

2 Vision

The vision of this project is to implement a flexible, pluggable, fully functional software application that will be maintainable, with detailed supporting documentation and an instruction manual for the Cafeteria Management System. This system will assist in managing the cafeteria's inventory/stock, placing orders made by employees, generating bills, and sending the appropriate information to the right parties.

3 Background

3.1 The current situation/ problems the client currently experience

As specified in the project proposal document from Resolve, the cafeteria is currently cash only and does not accept bank cards or electronic payments. This is inconvenient for employees as they have to carry around cash if they want to purchase anything from the cafeteria. Employees may choose to go to an external food outlet where they can pay with their preferred method of payment, which uses time and fuel. Thus, this means the cafeteria does not achieve the maximum amount of income which hinders its growth and improvement.
A problem with the cafeteria itself is that certain meal items are hardly in stock due to either lack of ingredients to make the meal or under estimating the quantity of the meal item required.

3.2 How the aforementioned problems will be alleviated by the CMS

The Cafeteria Management System will provide a means to accept payments from employees, at the canteen, using their employee access cards or access card numbers, with an amount being deducted from their salary at the end of the month. The option of cash payments ,however, will not be discarded. At the end of each month, the bill for the month will be sent to either payroll, to the employee, or to both. This option is thus configurable from the user's profile. The employee can also set a spending limit for each month. There will also be a system wide limit that users cannot exceed.
The system will predict which inventory items needs to be bought for the next week in order to avoid the "out of stock" situation described above. The system will also enforce that when the cafeteria manager adds meal items to the menu, he adds inventory items for each menu item. This is done so that each time a menu item requires an inventory item, the quantity of the inventory item will decrement until it reaches zero and is marked as "Out of Stock" on the menu. This is done so that when the user is ordering food, he/she can clearly see which items are not in stock and hence does not need to find this out at the canteen.

4 Functional Requirements and Application Design

In this section we will discuss the functional requirements.

4.1 Use Cases

Below is a list of all the use cases we have identified:

- Authentication
- Manage System
- Manage Profile
- Place Order
- Manage Cafeteria

- Manage Inventory
- Reporting

4.2 Use Case Prioritization

Below the use cases mentioned above will be categorized as critical, important or nice to have.

Critical

- **Authentication**
  This is a critical use case due to the fact that you cannot send through any order if you are not logged in. In such a case you will only be able to view the menu. In addition, you can not log into the system if you have not been registered on the system.

- **Place Order**
  This is critical due to the fact that the main functionality of the system revolves around the ability to place orders at the cafeteria as well as other functionality that is closely related to the placing order functionality.

- **Manage Inventory**
  This use case is considered critical because it deals with adding, removing, searching for and updating (i.e. incrementing stock that has been added and decrementing stock when it is purchased or expired). This is hence vital for achieving the purpose of the system. The items displayed on the menu will contain a field called "Not in stock" if there is not a sufficient supply of inventory for the various menu items and the option to order an 'out of stock' item will not be available.

- **Manage Cafeteria**
  This use case is considered critical because it deals with firstly adding menu items to the menu that the user will view, which again is vital for achieving the purpose of the system, as well as removing, updating and searching for various menu items.

Important

- **Manage Profile**
  This use case is considered important because the user must be able edit

their profile by resetting their personal spending limits and changing their email and passwords. The user must also be able to view his/her account history and current bill as well as his/her available balance for credit spending for the month. It is crucial that the user is able to configure spending limits according to their own preferences as well as keep track of monthly purchases.

- **Manage System**
  This use case is considered important because this is where the super user will configure the maximum spending limit, assign roles such as a cashier, change employee IDs as well as branding functionality such as setting the canteen name and uploading a canteen photo for the home page.

- **Reporting**
  This use case is considered important because the system can still function without it but it is needed to meet the requirements. It deals with checking how much a user spent in a particular period of time and how much the user needs to pay back and has paid back. This is where the finance person can check who owes how much to the cafeteria or who has spent how much in a given time frame.

5 Modular System

The system will be built using a modular approach to allow more modules to be added at a later stage. This will also provide for pluggability and integrability of the system.

5.1 High level use case diagram of the CMS

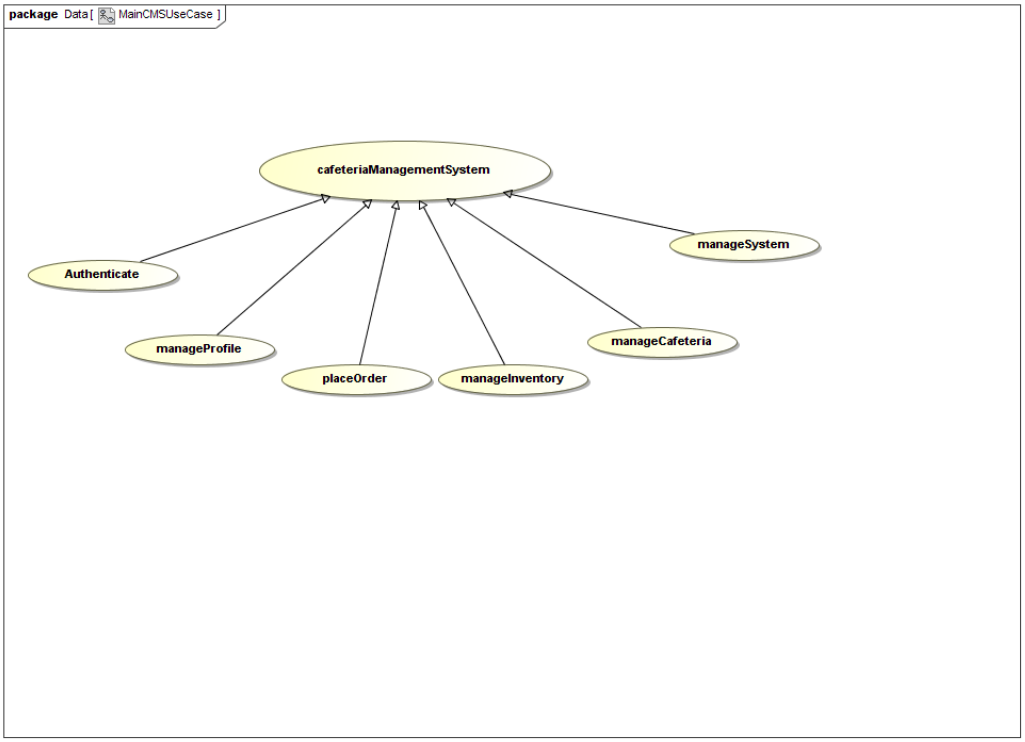

Figure 1: Cafeteria Management System Use Case

The core of the system is a cafeteria management system that will provide functionality such as allowing users to place orders once they have registered and logged on to the system. Different types of users will have different privileges.

5.2 Authentication Module

In this module, the functionality provided consists of validating the credentials entered into the system by a user whilst signing in. In addition, assistance is provided via the forgotPassword functionality. The different roles are also obtained in order to assign different functionality to users based on their roles. The register/ signup functionality is also included here.

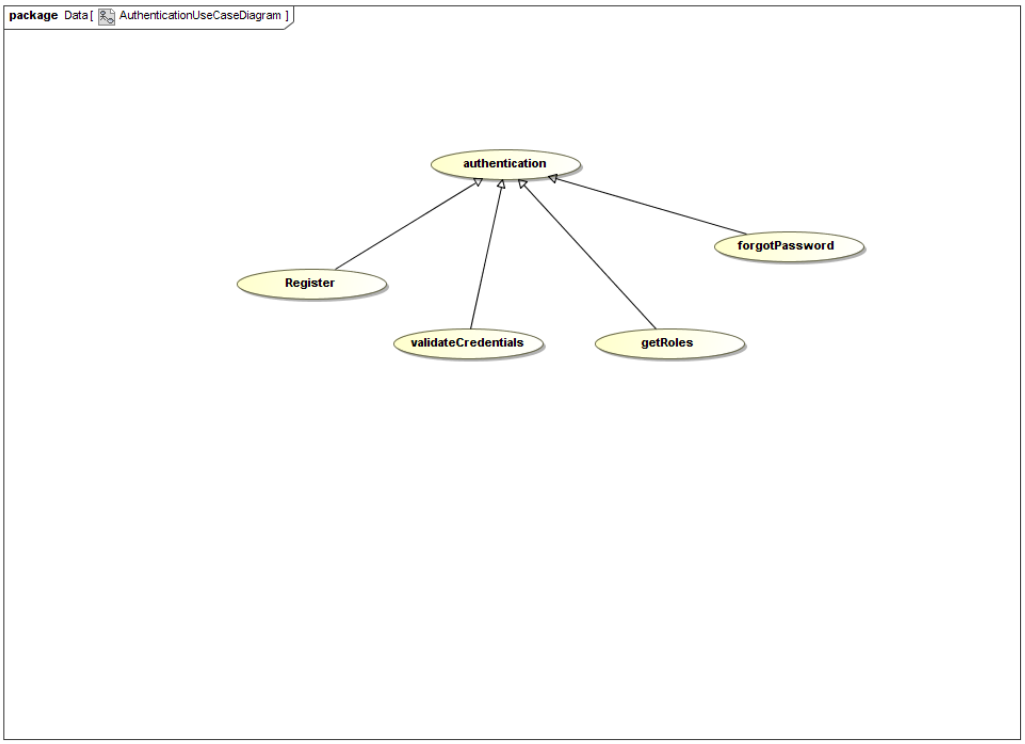

Figure 2: The main use case diagram for Authentication

Another crutial part of the authentication module is to verify if a user may have access to certain pages given his/her role which will also contribute to the security of the data stored on our system and to control which user has access to which functionality.

Forgot Password

The service contract and activity diagram for forgotPassword follow. forgotPassword falls under the use case for Authentication (refer to page 8 - figure 2 to view this use case diagram). Here, a user who has forgotten their password will be assisted via sending an email to the user's email account with steps to follow to create a new password.

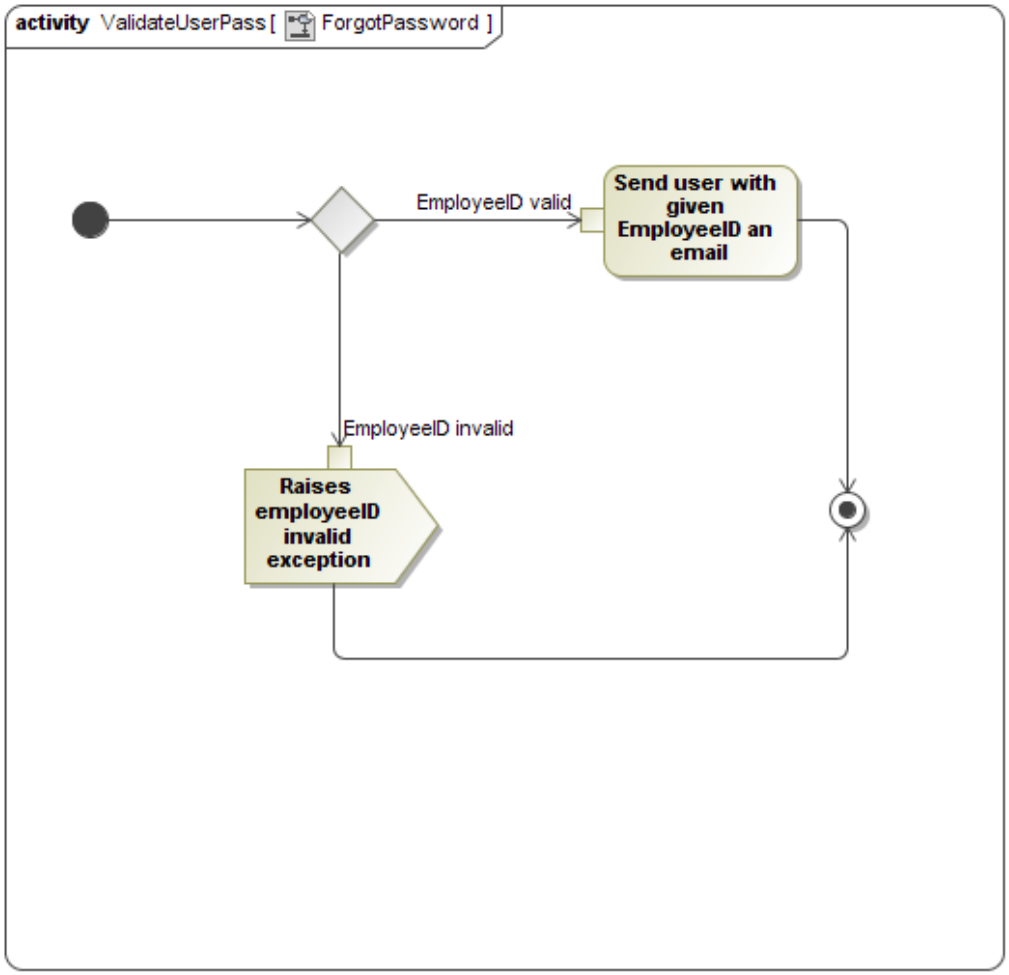

Figure 3: The activity diagram for forgotPassword

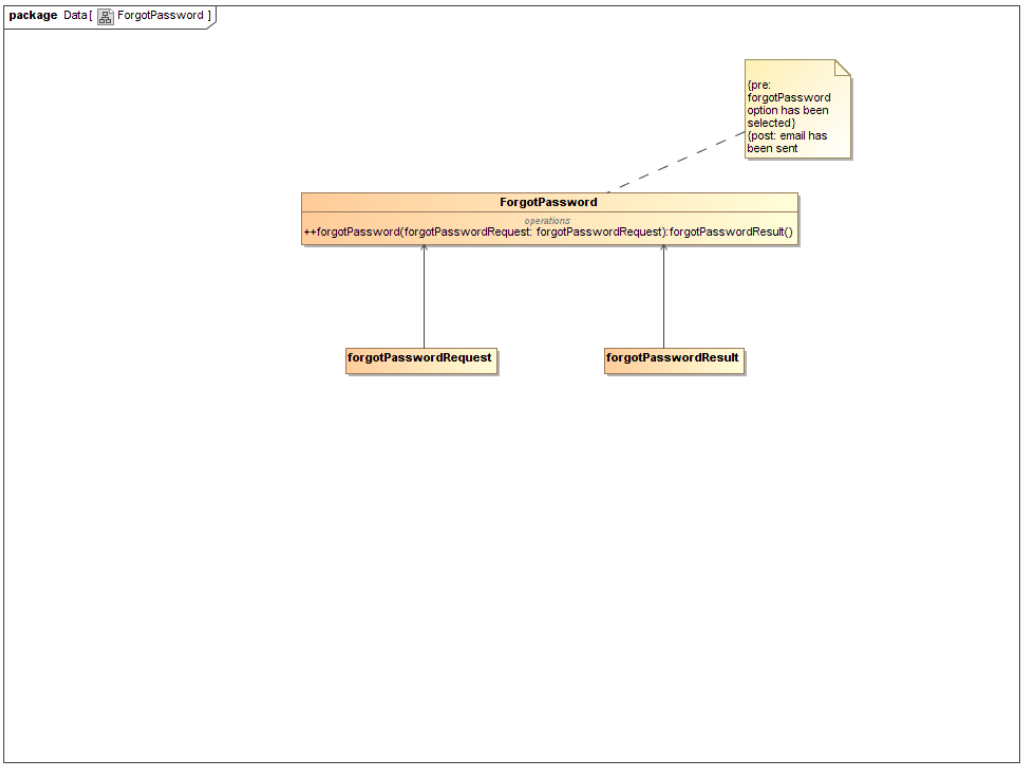

Figure 4: The service contract for forgotPassword

Validate User credentials

The service contract and activity diagram for validateUserCredentials follow. validateUserCredentials falls under the use case for Authentication (refer to page 8 - figure 2 to view this use case diagram). Here, a user who has forgotten their password will be assisted via sending an email to the user's email account with steps to follow to create a new password.

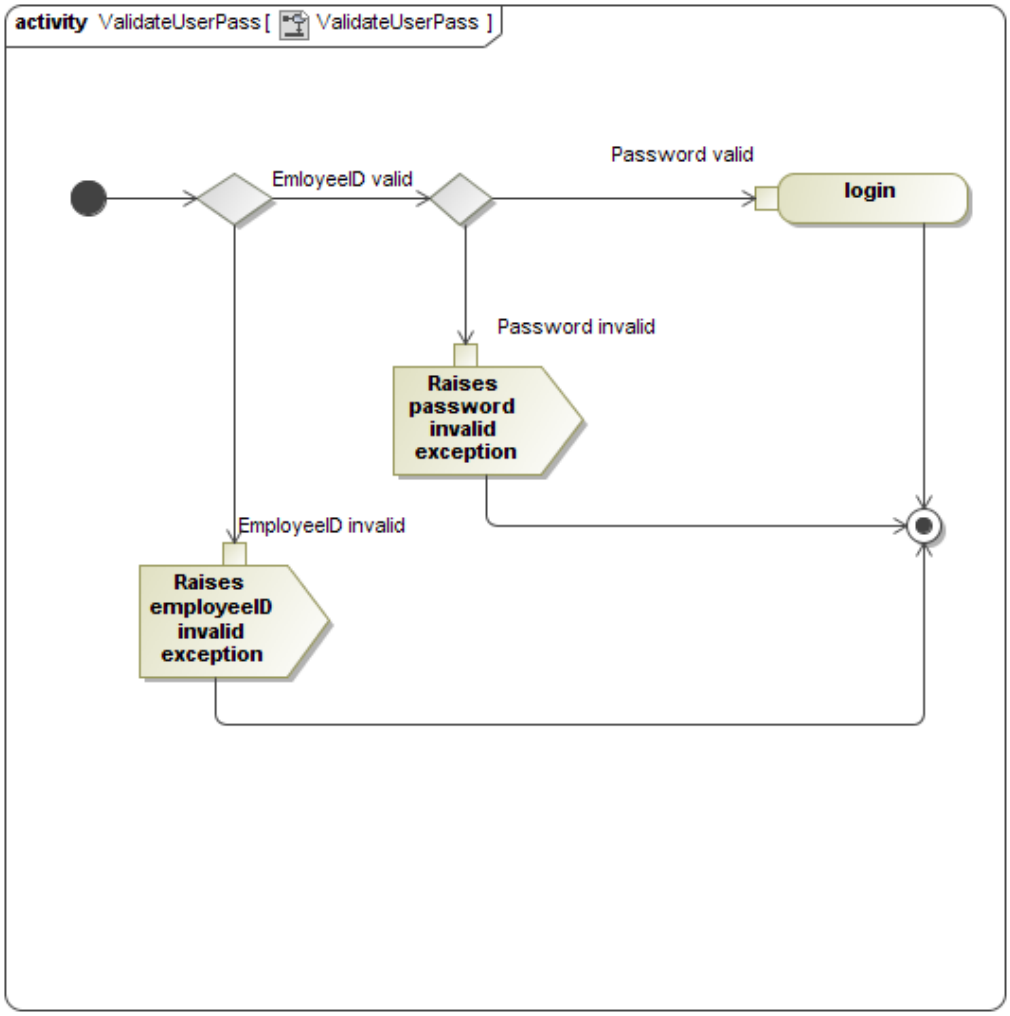

Figure 5: The activity diagram for validateUserCredentials

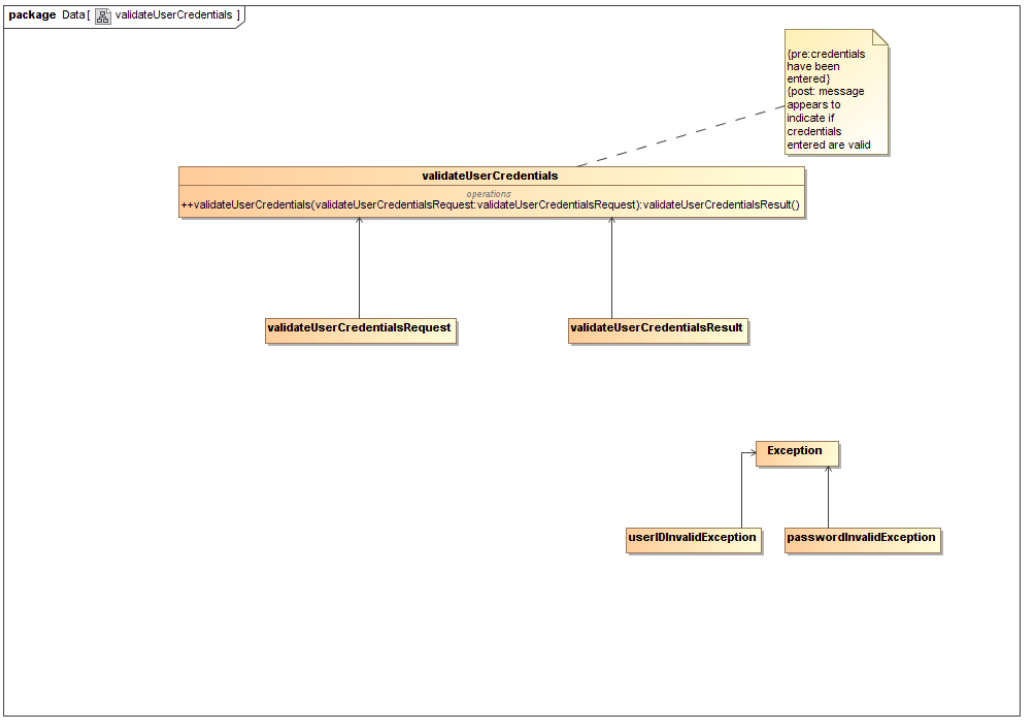

Figure 6: The service contract for validateUserCredentials

Register

Register provides the functionality to sign up as a user of the system. The user will set their limit, and personal details upon registration, as well as the recipient of their monthly bill. This is indicated in the following use case diagram.

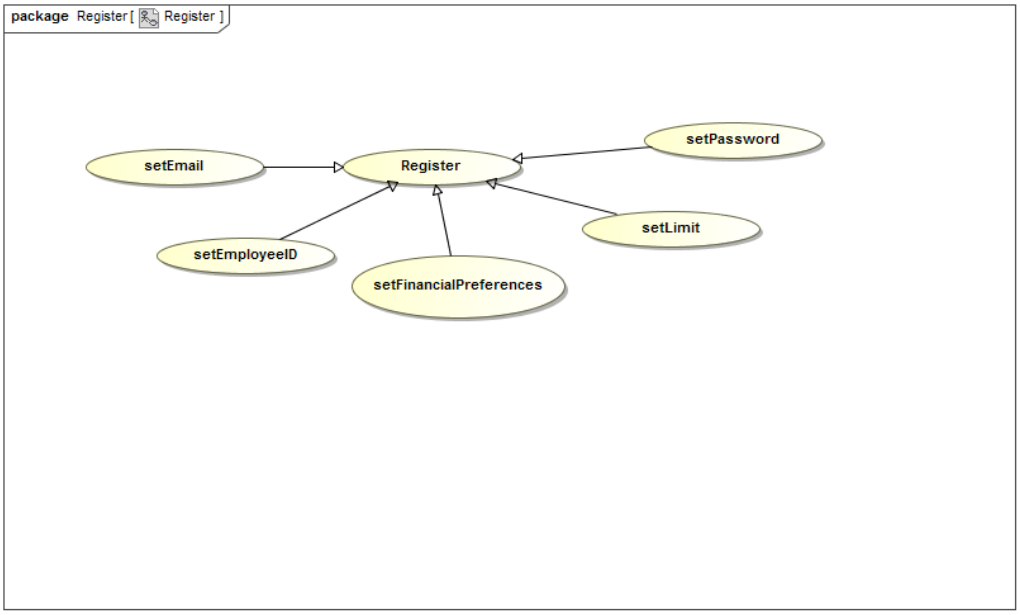

Figure 7: The use case for registering on the system

Set email

The service contract and activity diagram for setEmail follow. setEmail falls under the use case for Register (refer to page 13 - figure 7 to view this use case diagram). These details, entered by the user will be stored on the system.

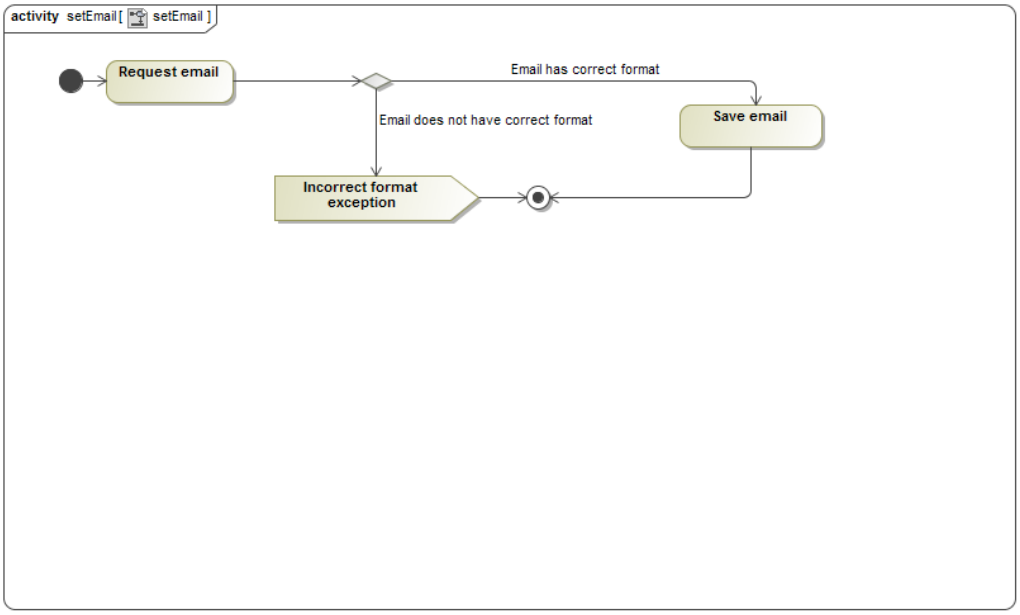

Figure 8: The activity diagram for setting an email address on the system

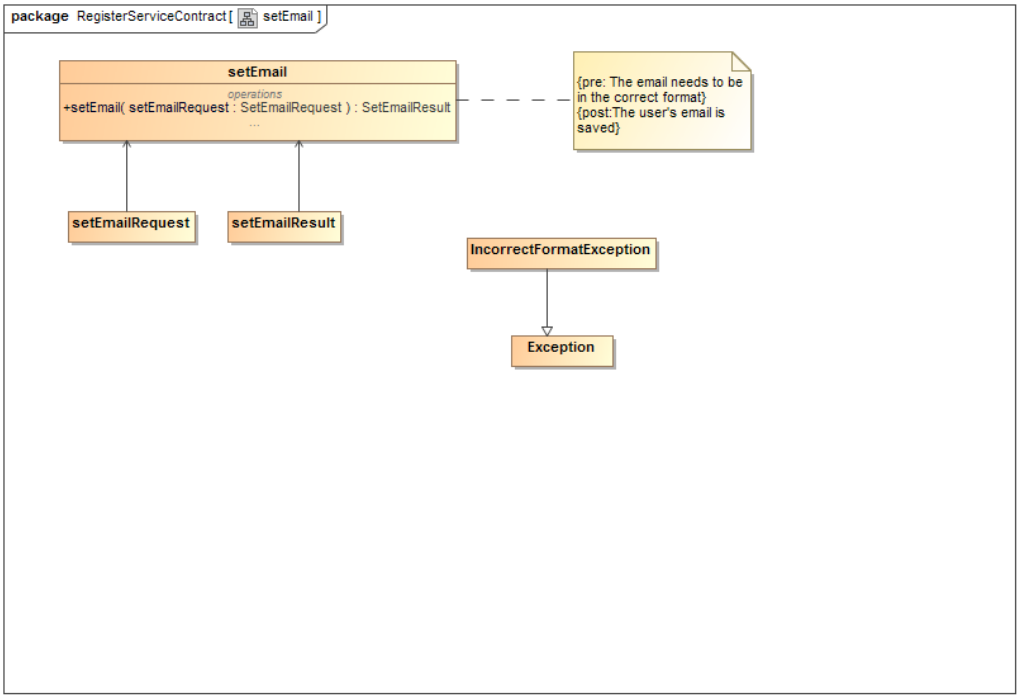

Figure 9: The service contract for setting an email address on the system

Set Password

The service contract and activity diagram for setPassword follow. setPassword falls under the use case for Register (refer to page 13 - figure 7 to view this use case diagram). These details, entered by the user will be stored on the system.

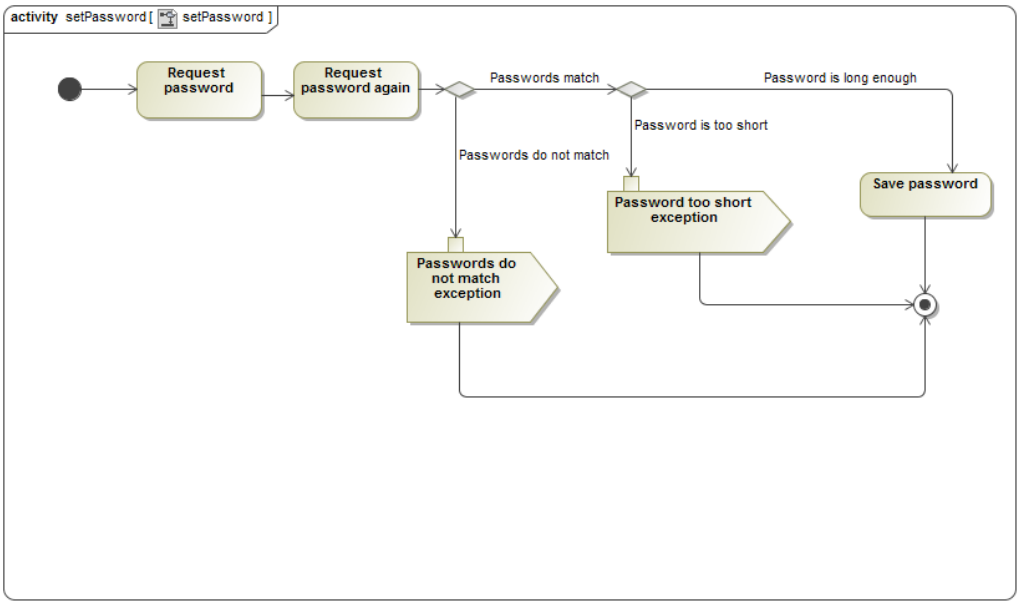

Figure 10: The activity diagram for setting a password on the system

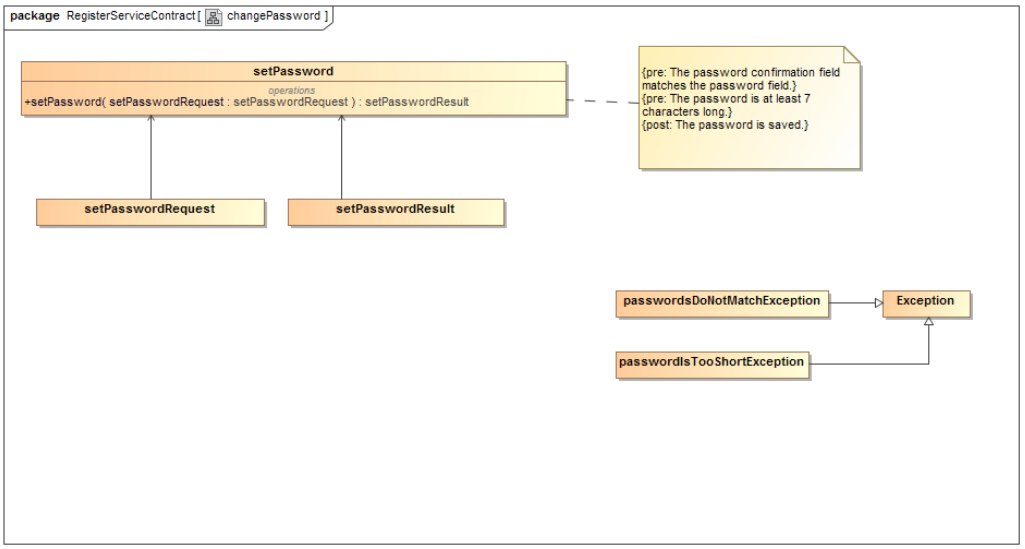

Figure 11: The service contract for setting a password on the system

Set Limit

The service contract and activity diagram for setLimit follow. setLimit falls under the use case for Register (refer to page 13 - figure 7 to view this use case diagram). These details, entered by the user will be stored on the system.

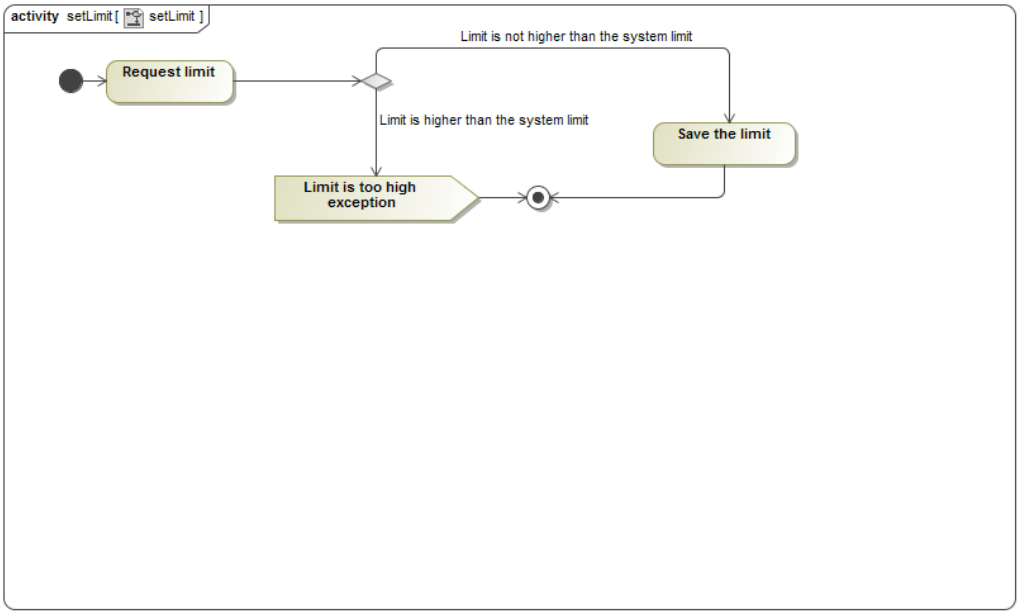

Figure 12: The activity diagram for setting a limit on the system

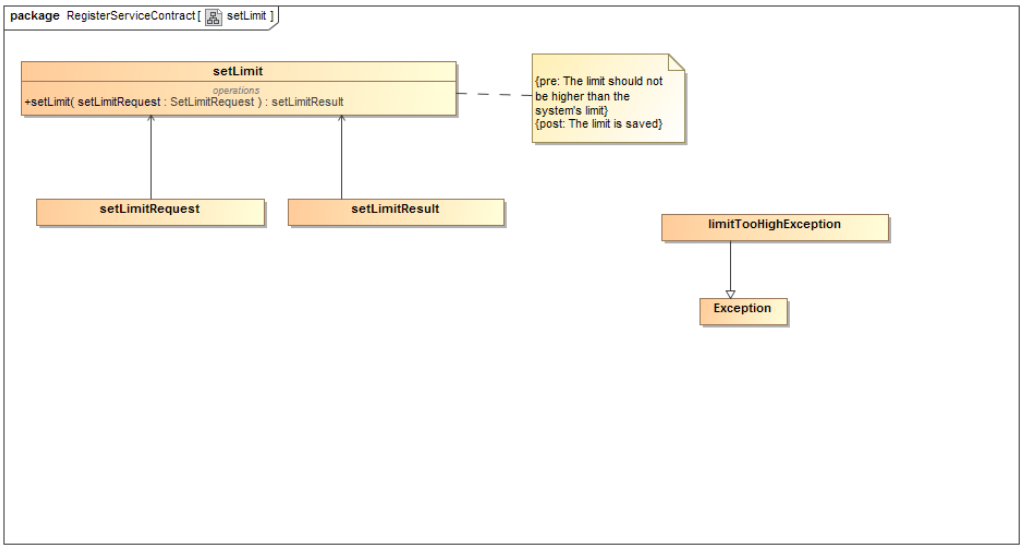

Figure 13: The service contract for setting a limit on the system

Set Employee ID

The service contract and activity diagram for setEmployeeID follow. setEmployeeID falls under the use case for Register (refer to page 13 - figure 7 to view this use case diagram). These details, entered by the user will be stored on the system.

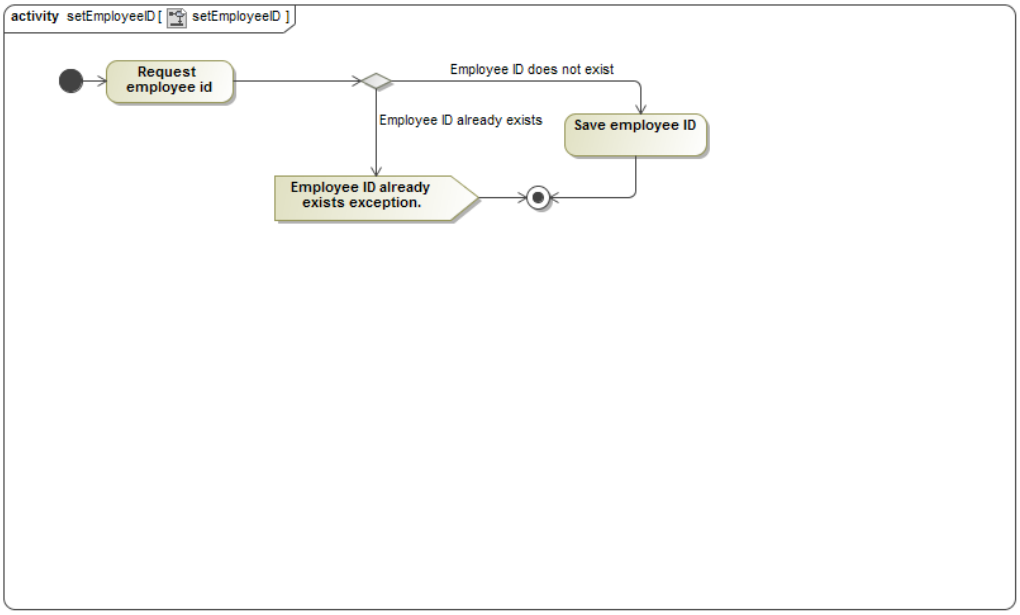

Figure 14: The activity diagram for setting an employeeId on the system

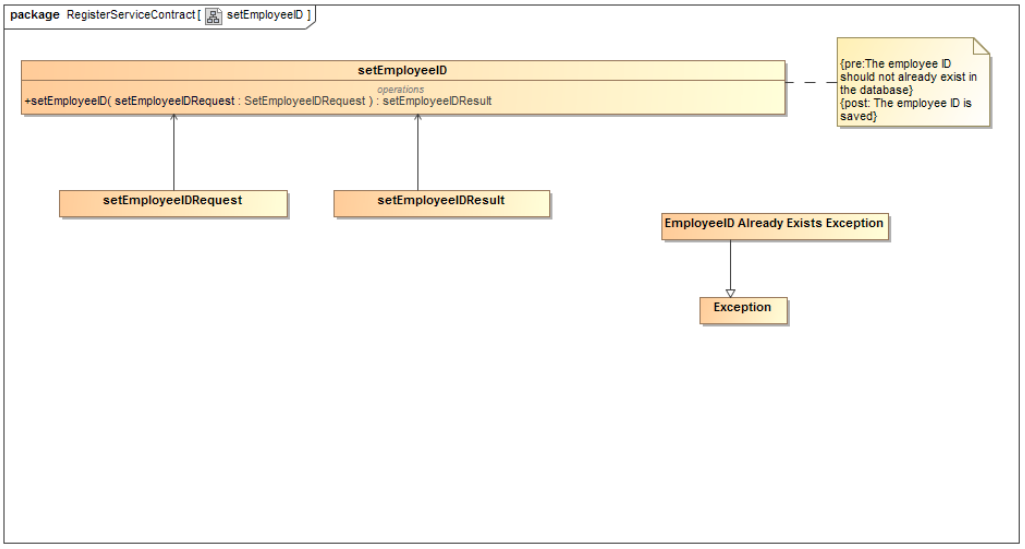

Figure 15: The service contract for setting an employeeId on the system

Set Financial Preferences

The service contract and activity diagram for setFinancialPreferences follow. setFinancialPreferences falls under the use case for Register (refer to page

13- figure 7 to view this use case diagram). These details, entered by the user will be stored on the system.

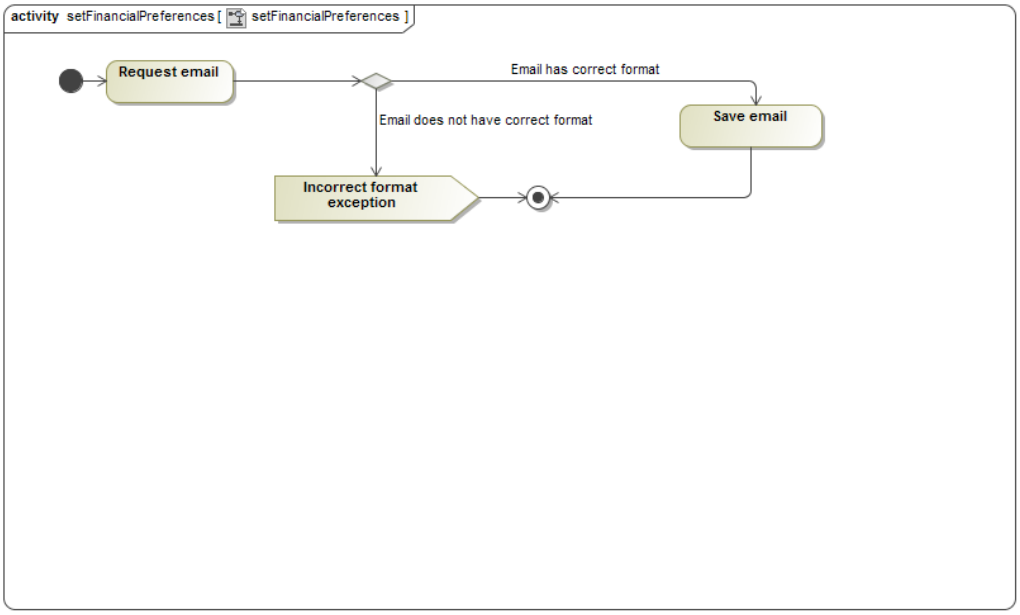

Figure 16: The activity diagram for setting financial preferences on the system

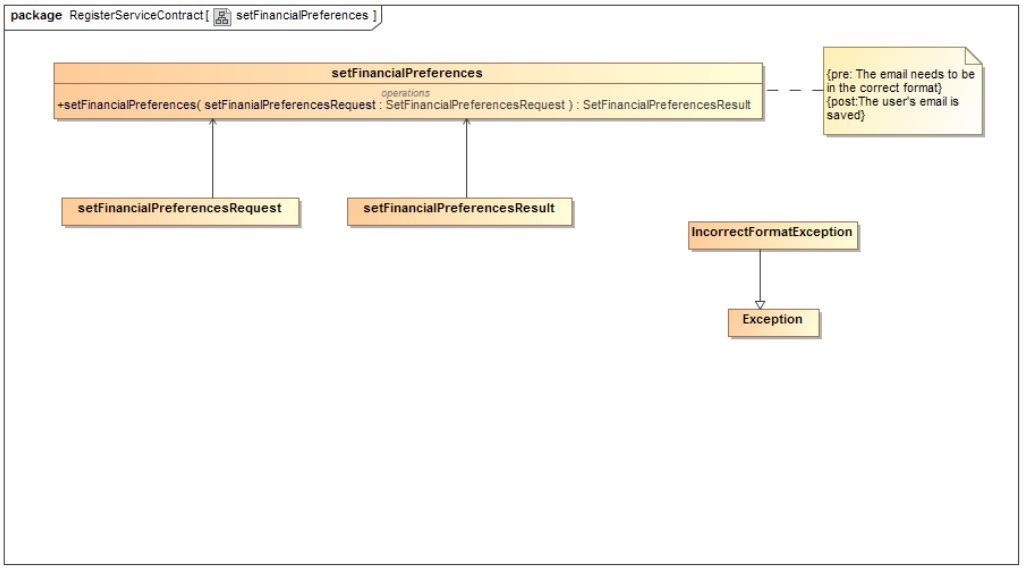

Figure 17: The service contract for setting financial preferences on the system

5.3 Manage Cafeteria Module

The menu module consists of the functionality of adding and removing menu items from the menu from which the user will place orders. This use case diagram indicates the functionality that this module consists of such as addMenuItem, updateMenuItem, searchMenuItem and deleteMenuItem, which are the use cases of this module.

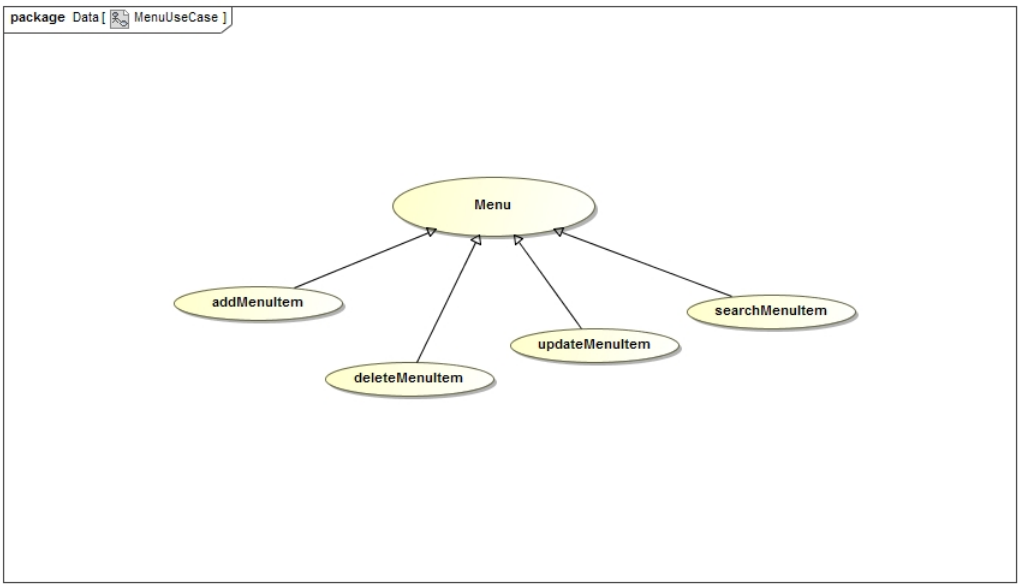

Figure 18: The use case for menu

addMenuItem

The service contract and activity diagram for addMenuItem follow. addMenuItem falls under the use case for Menu (refer to page 20 - figure 18 to view this use case diagram). The system allows the cafeteria manager to add items to the menu via the menu managing page.

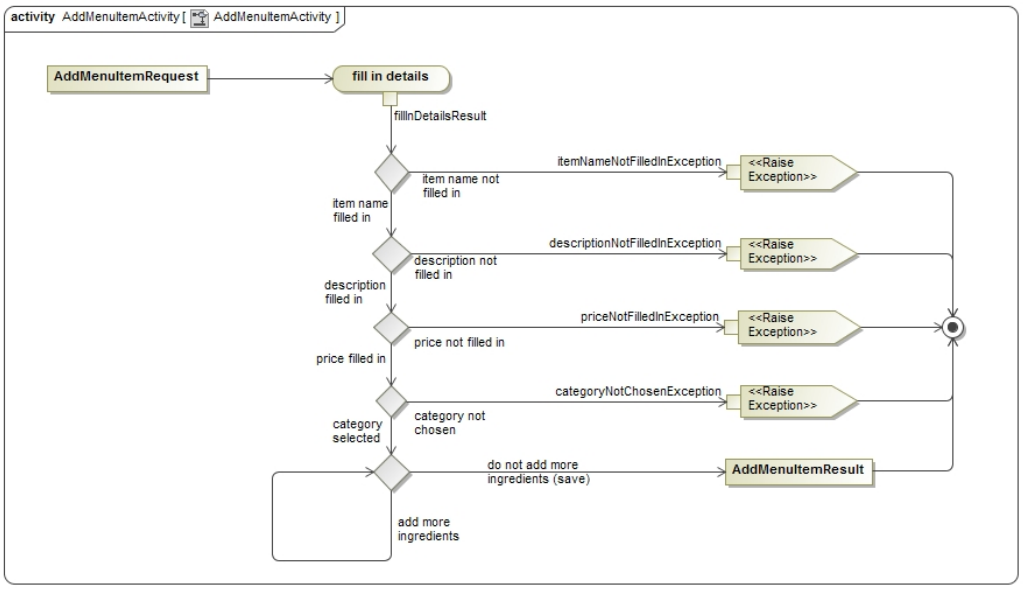

Figure 19: The activity diagram for adding a menu item

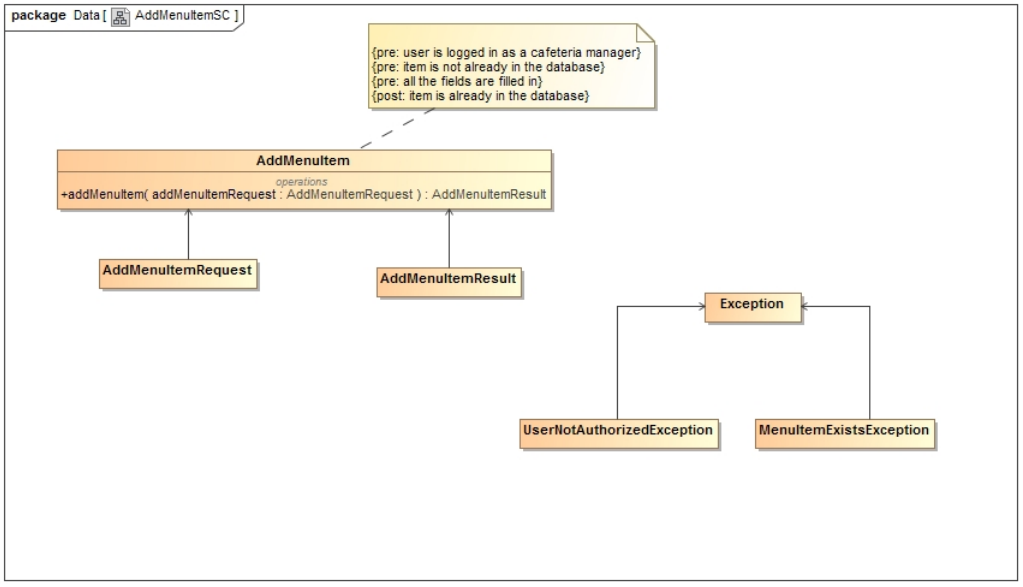

Figure 20: The service contract for adding a menu item

deleteMenuItem

The service contract and activity diagram for deleteMenuItem follow. deleteMenuItem falls under the use case for Menu (refer to page 20 - figure 18 to

view this use case diagram). The system allows the cafeteria manager to delete items from the menu via the menu managing page.

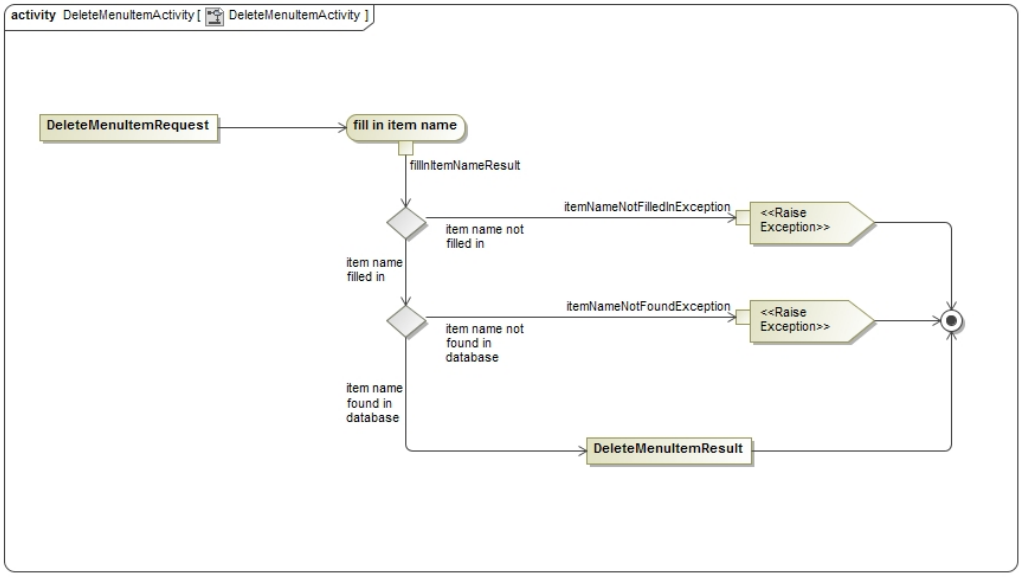

Figure 21: The activity diagram for deleting a menu item

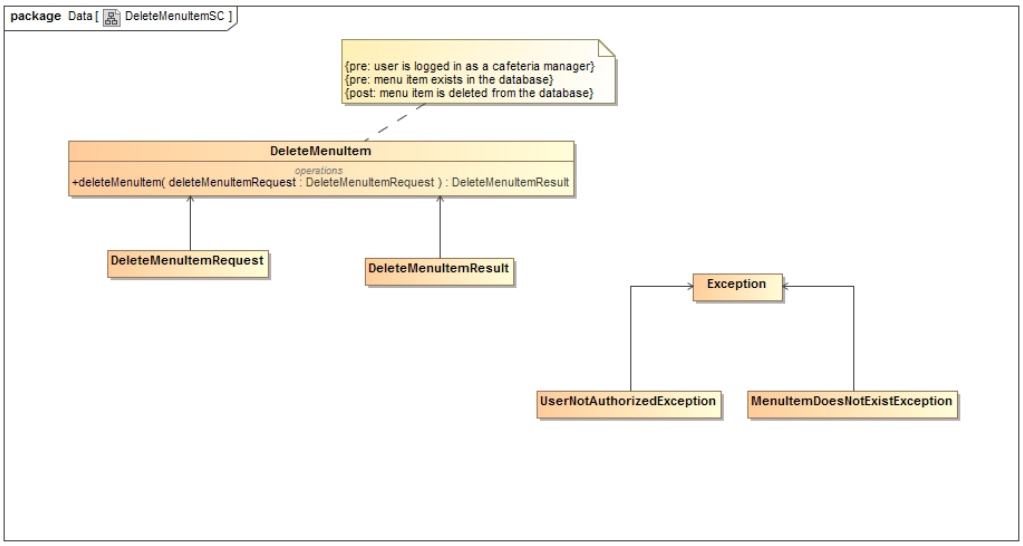

Figure 22: The service contract for deleting a menu item

searchMenuItem

The service contract and activity diagram for searchMenuItem follow. searchMenuItem falls under the use case for Menu (refer to page 20 - figure 18 to view this use case diagram). The system allows the cafeteria manager to search for items from the menu via the menu managing page.

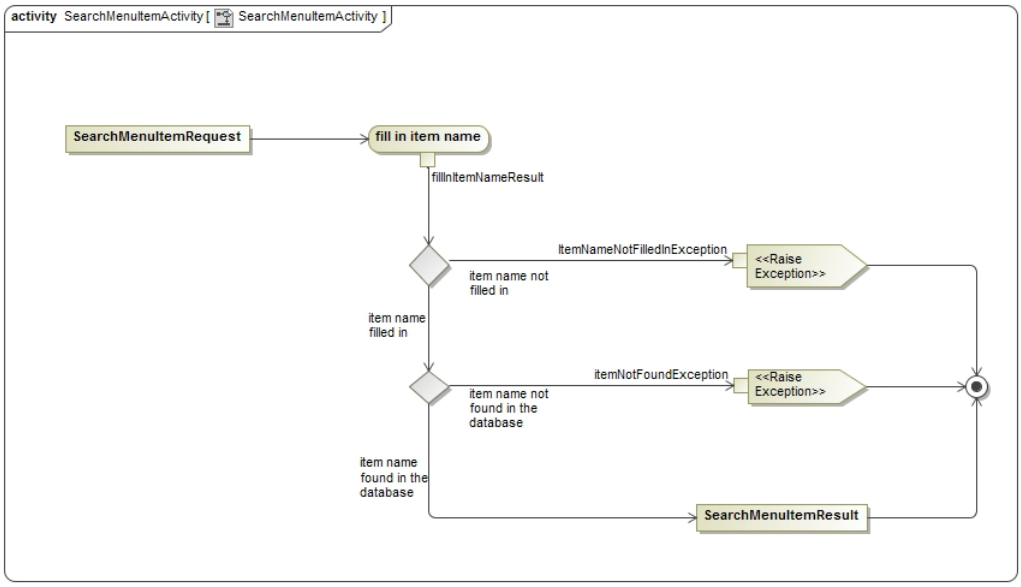

Figure 23: The activity diagram for searching for a menu item

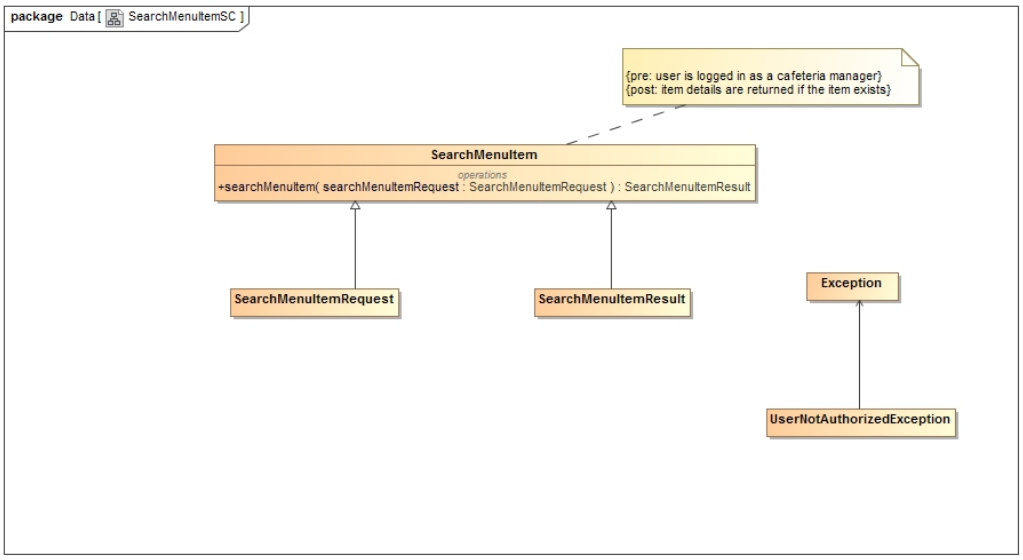

Figure 24: The service contract for searching for a menu item

updateMenuItem

The service contract and activity diagram for updateMenuItem follow. updateMenuItem falls under the use case for Menu (refer to page 20 - figure 18 to view this use case diagram). The system allows the cafeteria manager to update menu items' unit, quantity and name via the menu managing page.

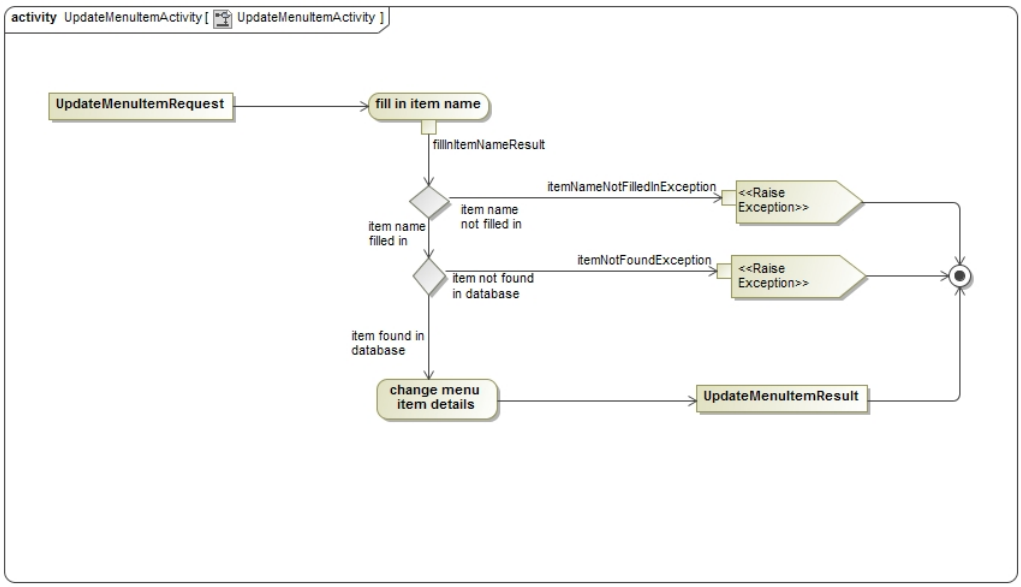

Figure 25: The activity diagram for updating a menu item

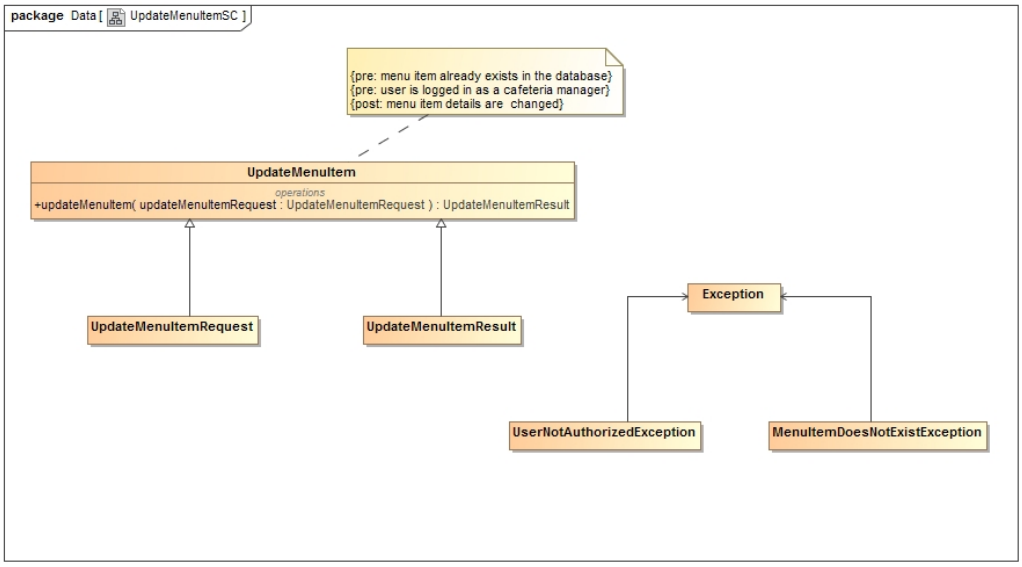

Figure 26: The service contract for updating a menu item

5.4 Place Orders Module

The main functionality the system serves to provide, is to allow the user to use their access card number to purchase food items from the cafeteria via

the system. This use case diagram indicates the functionality around placing orders, such as CheckProductAvailability and checkLimits, which are the use cases of this module.

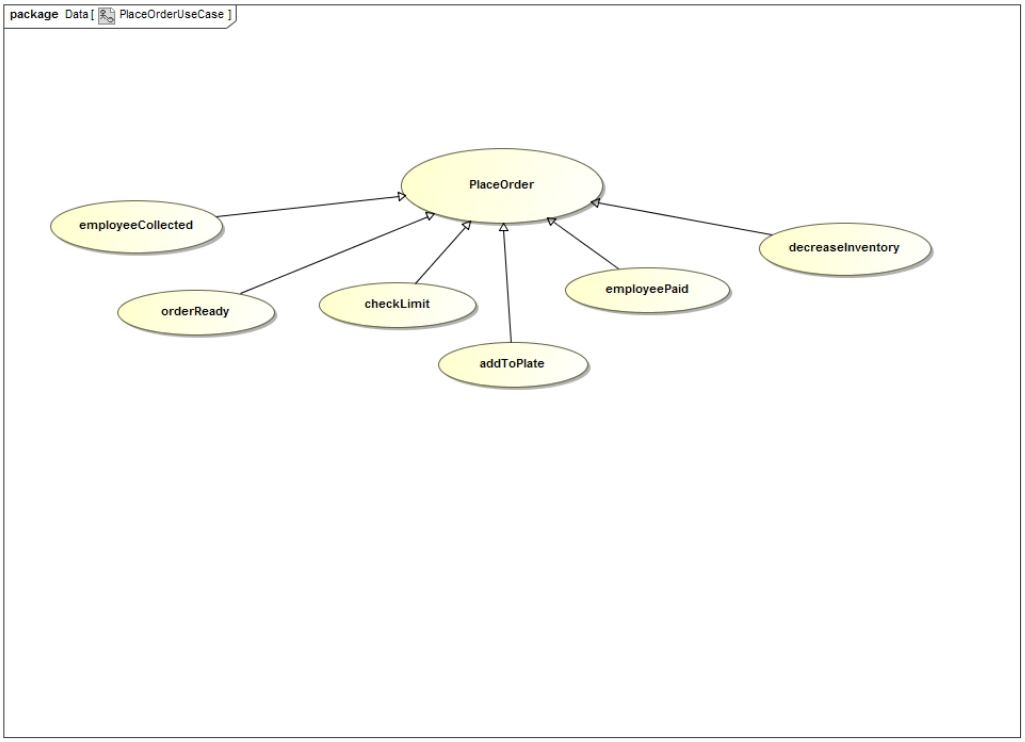

Figure 27: The use case for placing an order

CheckProductAvailability

The service contract and activity diagram for CheckProductAvailability follow. CheckProductAvailability falls under the use case for Place Orders (refer to page 26 - figure 27 to view this use case diagram). The system will check whether the product that the user has selected to purchase is currently in stock.

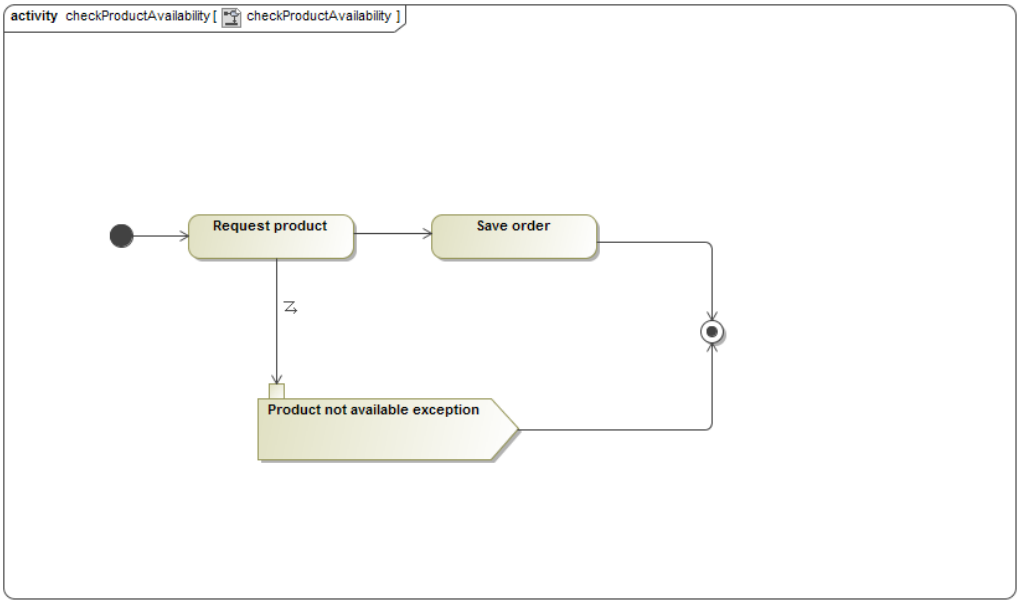

Figure 28: The activity diagram for checking product availability

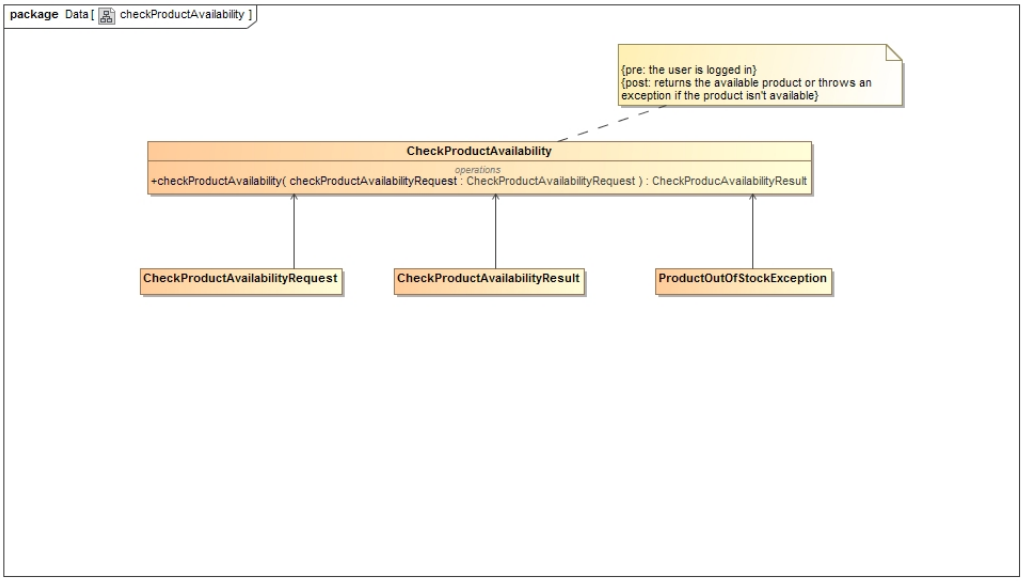

Figure 29: The service contract for checking product availability

markAsReady

The service contract and activity diagram for markAsReady follow. markAsReady falls under the use case for Place Orders (refer to page 26 - figure 27 to view this use case diagram). The cashier will be the only user with access to this functionality and when the order is marked as ready, the status of the order in the model is changed to ready. This will also send an email notification to the user that their order is ready for collection.

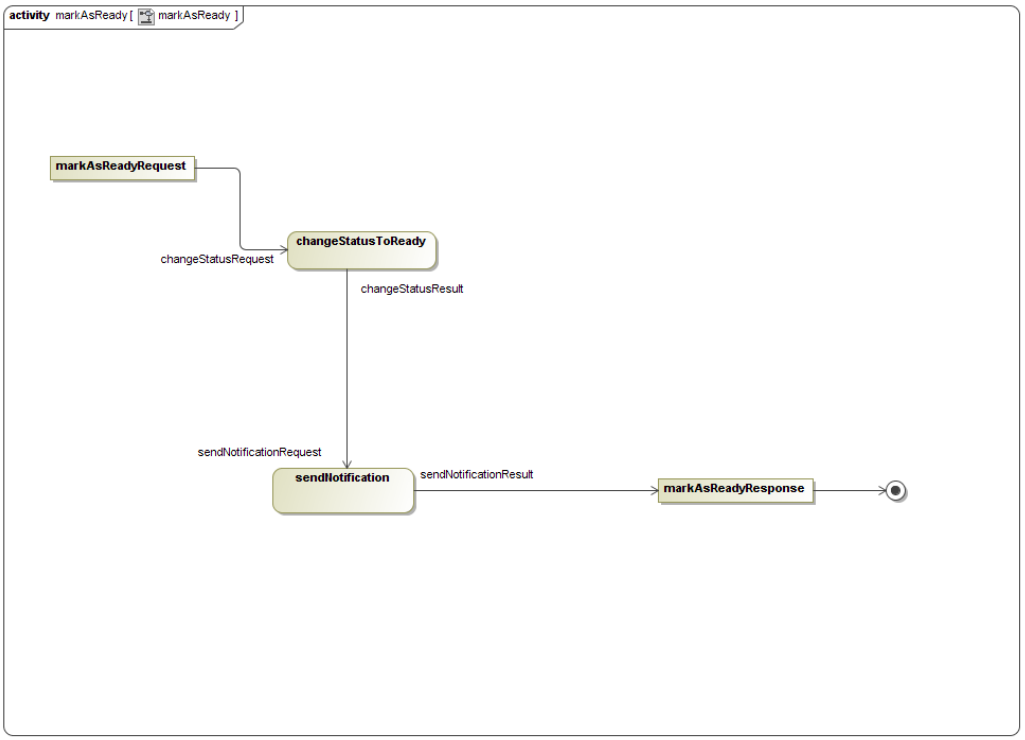

Figure 30: The activity diagram for marking an order as ready

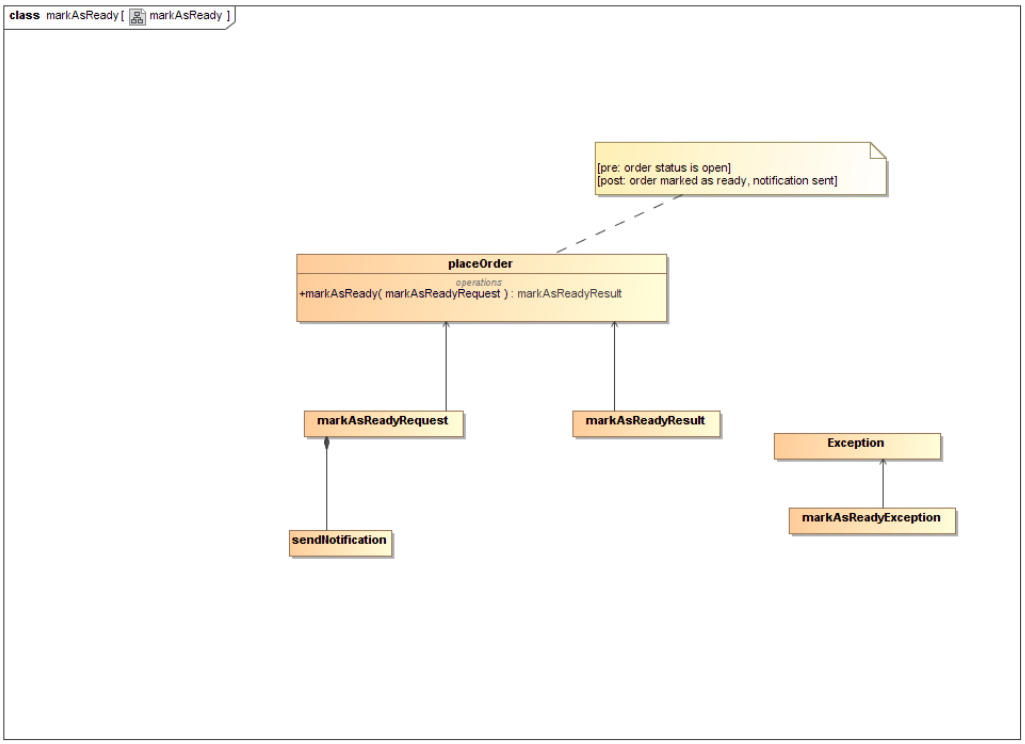

Figure 31: The service contract for marking an order as ready

addToPlate

The service contract and activity diagram for addToPlate follow. addToPlate falls under the use case for PlaceOrder (refer to page 26- figure 27 to view this use case diagram). This is where a user can select items from the menu and save them on their plate via the addToPlate button. The user can then view these via the "On my plate" page and can also edit these.

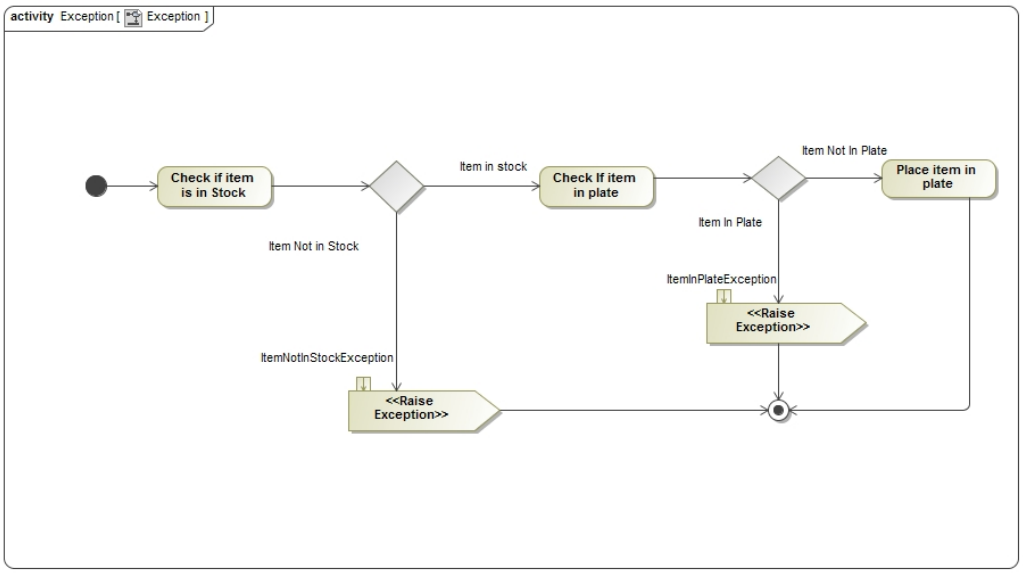

Figure 32: The activity diagram for adding a menu item to plate

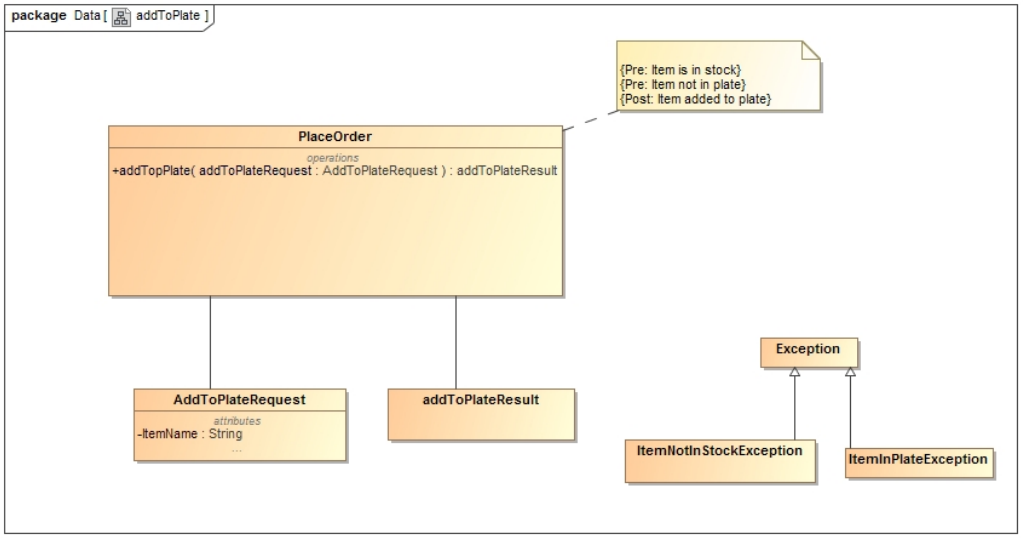

Figure 33: The service contract for adding a menu item to plate

markAsCollected

The service contract and activity diagram for markAsCollected follow. markAsCollected falls under the use case for Place Orders (refer to page 26 - figure 27 to view this use case diagram). The cashier will be the only user with

access to this functionality and when the order is marked as collected, the status of the order in the model is changed to closed. It is important to note that by marking it as just merely "collected", the system will know to deduct the amount from the user's monthly balance, as the user has not paid with cash. This hence marks that the order has collected their order and their order has been processed. It will then no longer be displayed on the cashiers page.

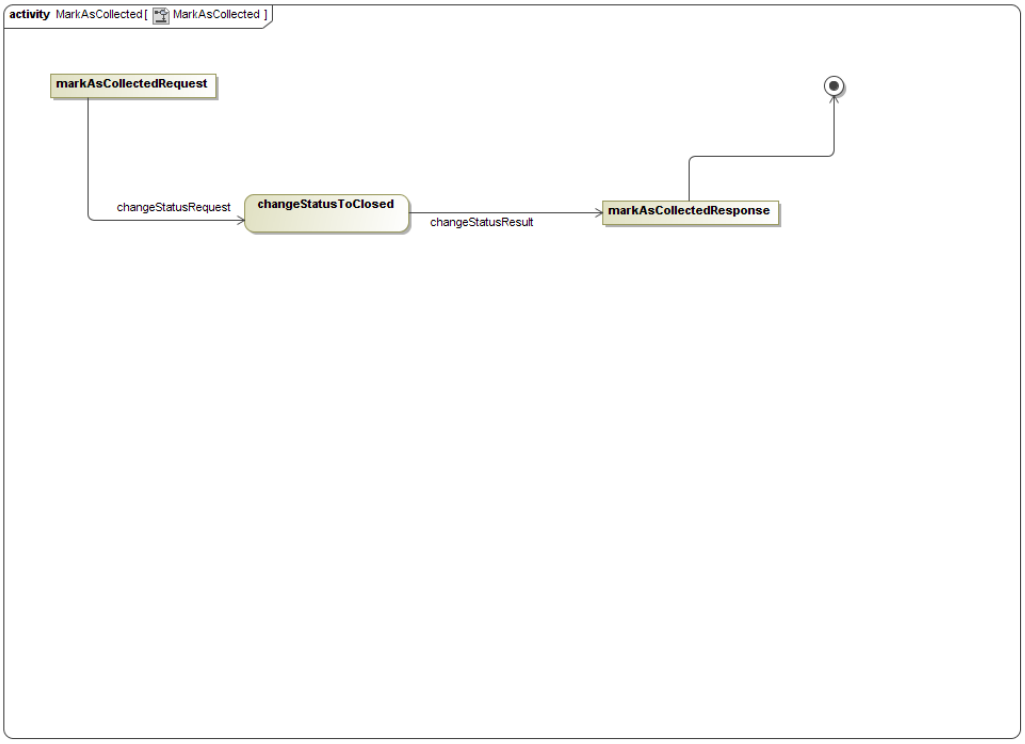

Figure 34: The activity diagram for marking an order as collected

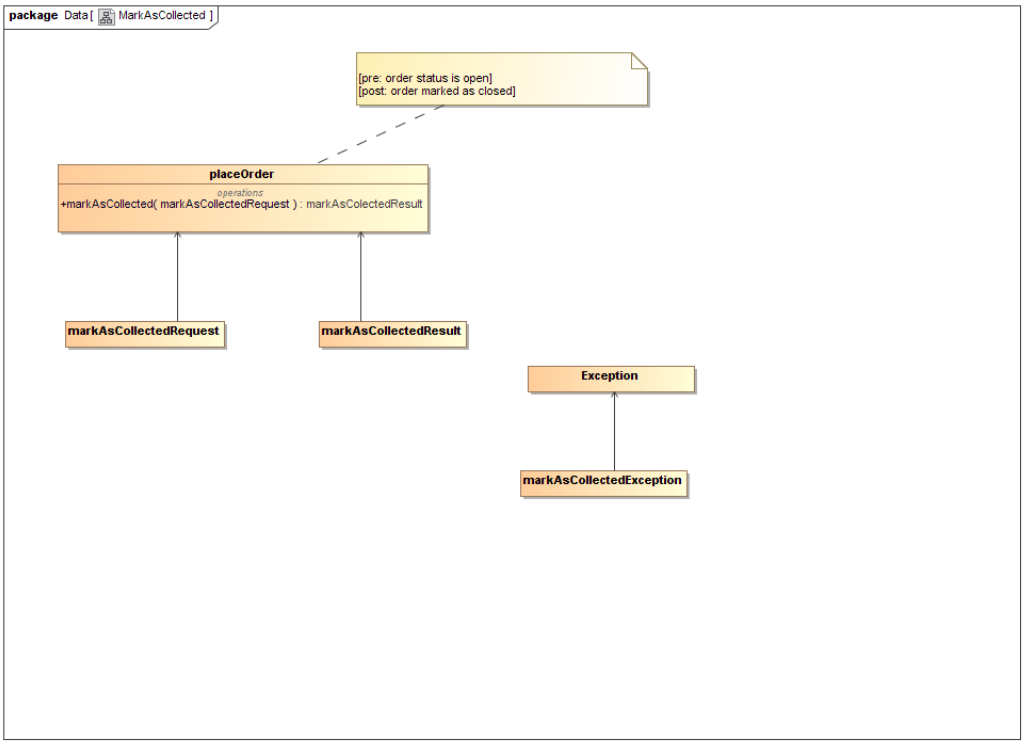

Figure 35: The service contract for marking an order as collected

checkLimit

The service contract and activity diagram for checkLimit follow. checkLimit falls under the use case for Place Orders (refer to page 26 figure 27 to view this use case diagram). The system will be able to view the user's personal limit to make sure that the total of the bill is not larger than the users spending limit.

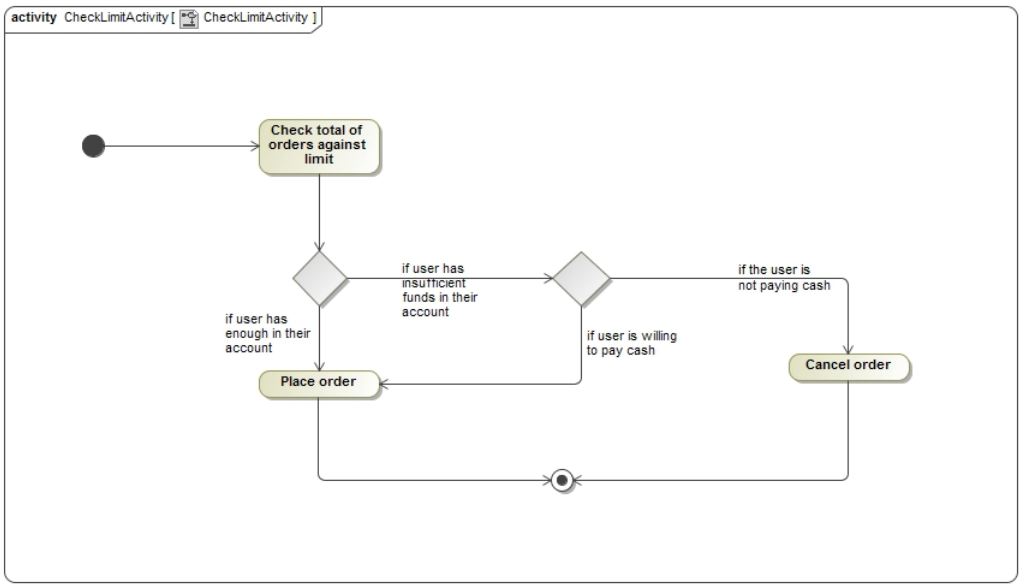

Figure 36: The activity diagram for checking limits

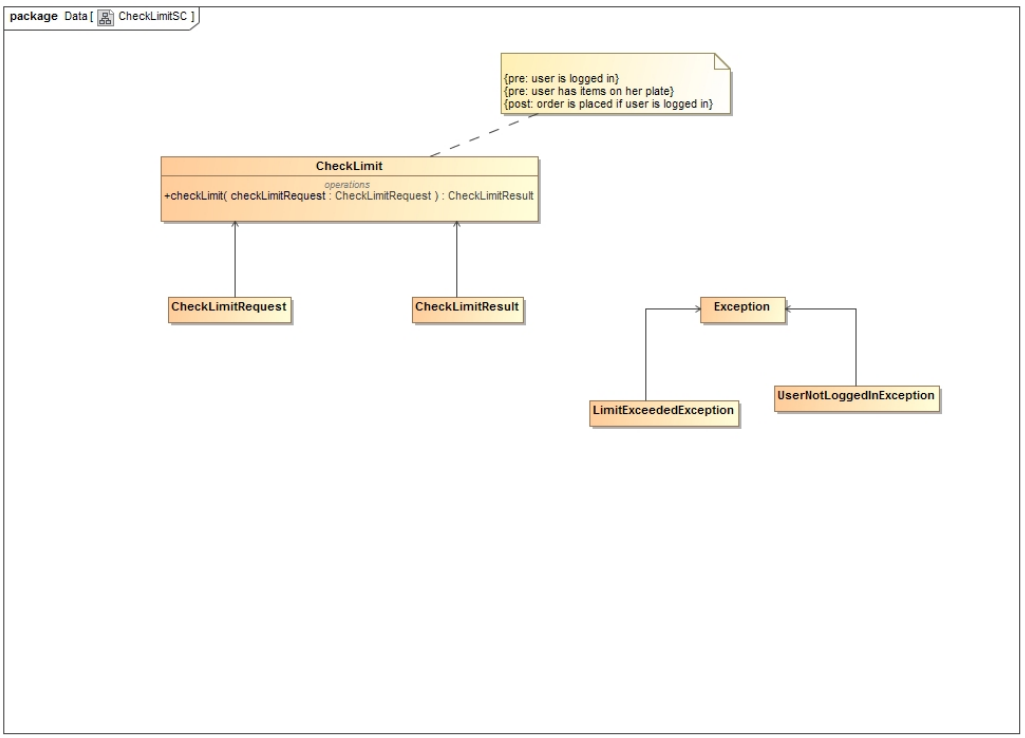

Figure 37: The service contract for checking limits

employeePaid

The service contract and activity diagram for employeePaid follow. employeePaid falls under the use case for Place Orders (refer to page 26 figure 27 to view this use case diagram). The cashier will be able to check this off if the client pays using cash instead of paying from their account. This will ensure that money isn't deducted from the user's limit because they paid cash.

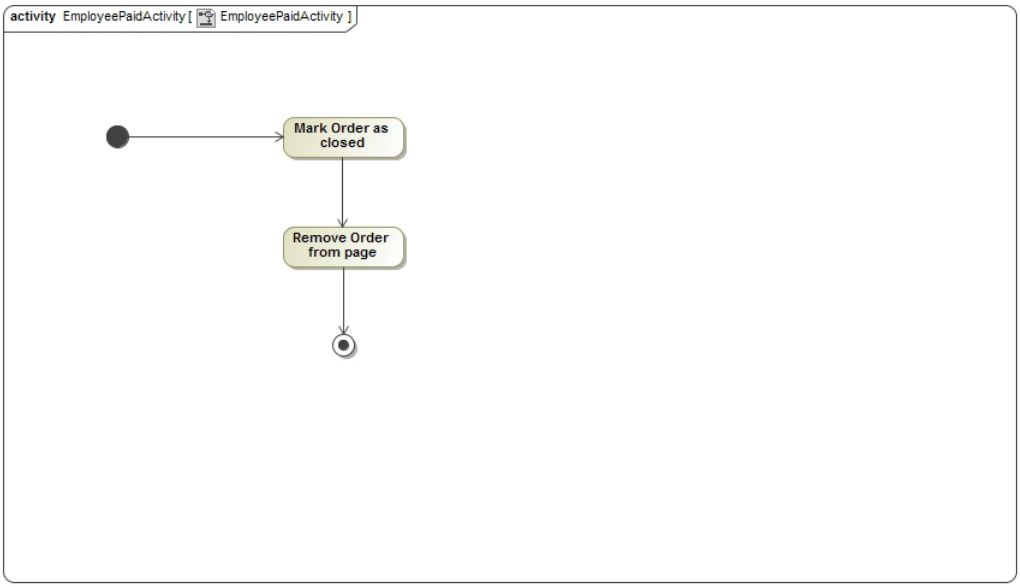

Figure 38: The activity diagram for employee paid

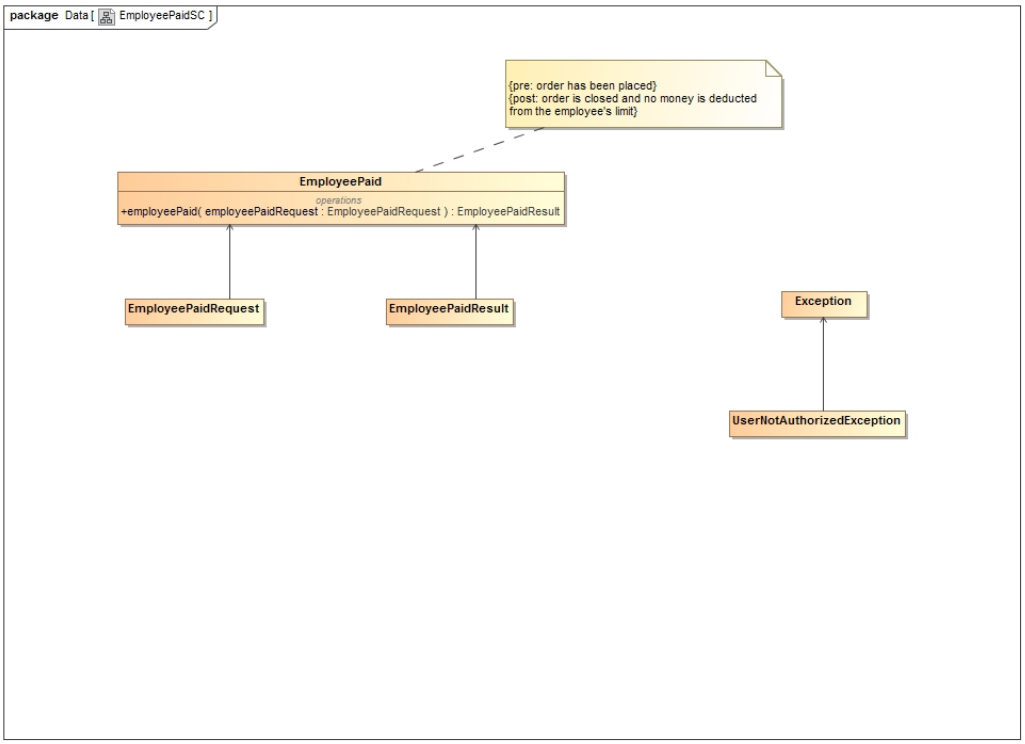

Figure 39: The service contract for employee paid

5.5 Manage Inventory Module

The inventory module is where the cafeteria manager will keep track of stock additions and removals. Different menu items require certain inventory items.This use case diagram indicates the functionality around adding, deleting, searching for and updating inventory.

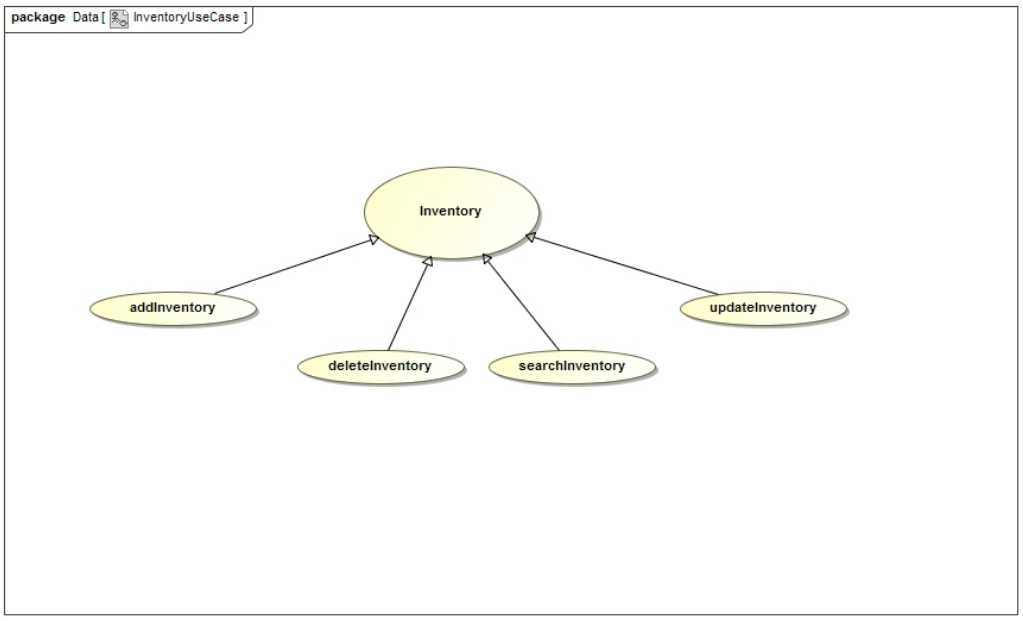

Figure 40: The use case for inventory

addInventory

The service contract and activity diagram for addInventory to follow. addInventory falls under the use case for Inventory (refer to page 36 figure 40 to view this use case diagram). The cafeteria manager will be able to manage inventory items via the allocated page.

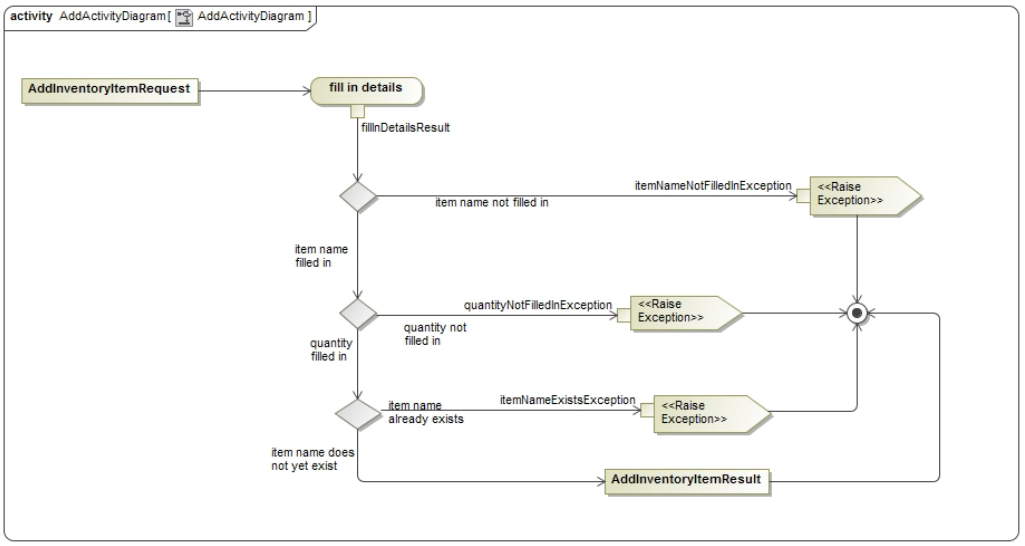

Figure 41: The activity diagram for adding inventory

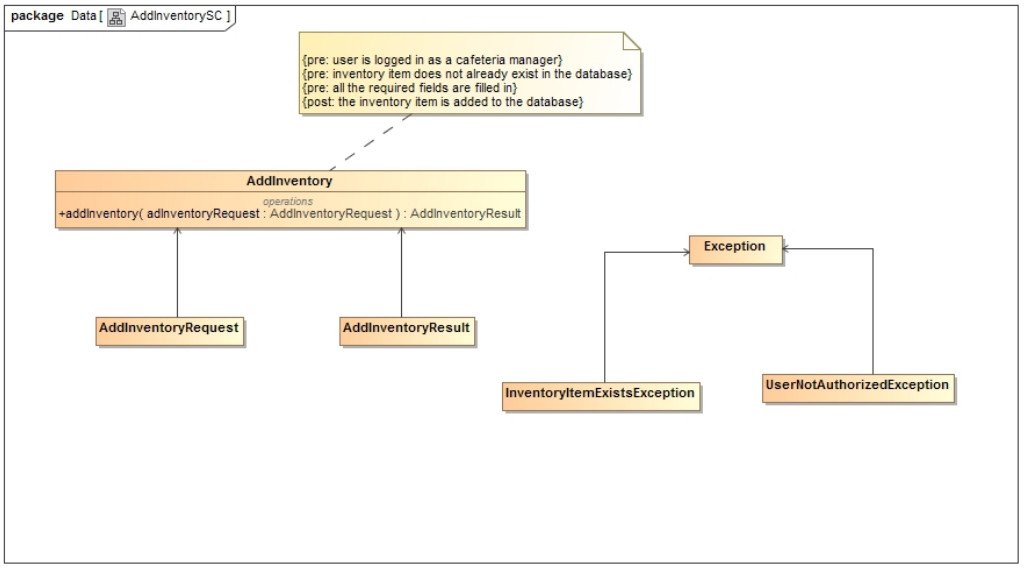

Figure 42: The service contract for adding inventory

searchInventory

The service contract and activity diagram for searchInventory to follow. searchInventory falls under the use case for Inventory (refer to page 36 figure 40 to view this use case diagram). The cafeteria manager will be able to

search for inventory items via the allocated page and from there be able to delete or update them.

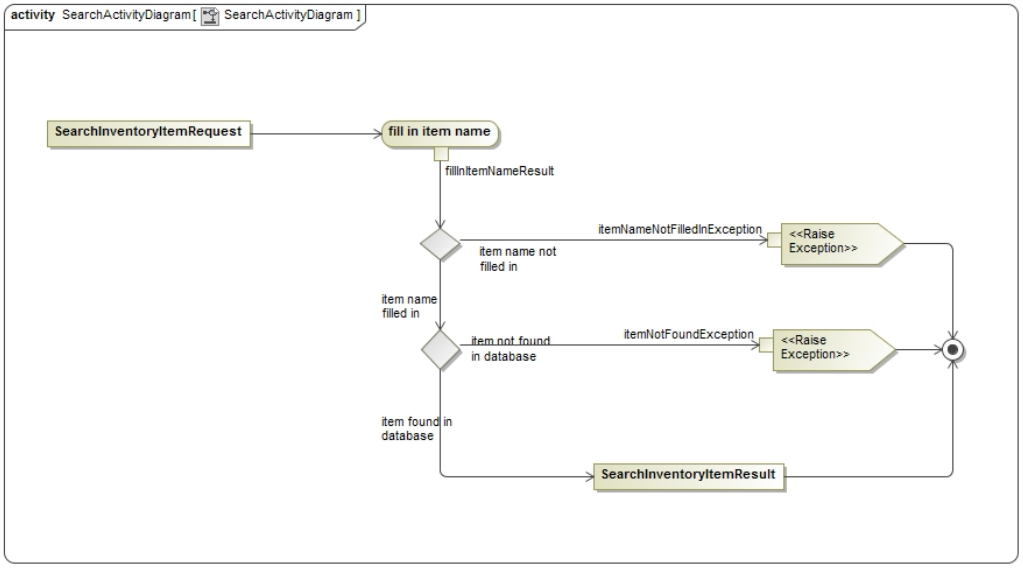

Figure 43: The activity diagram for searching for inventory

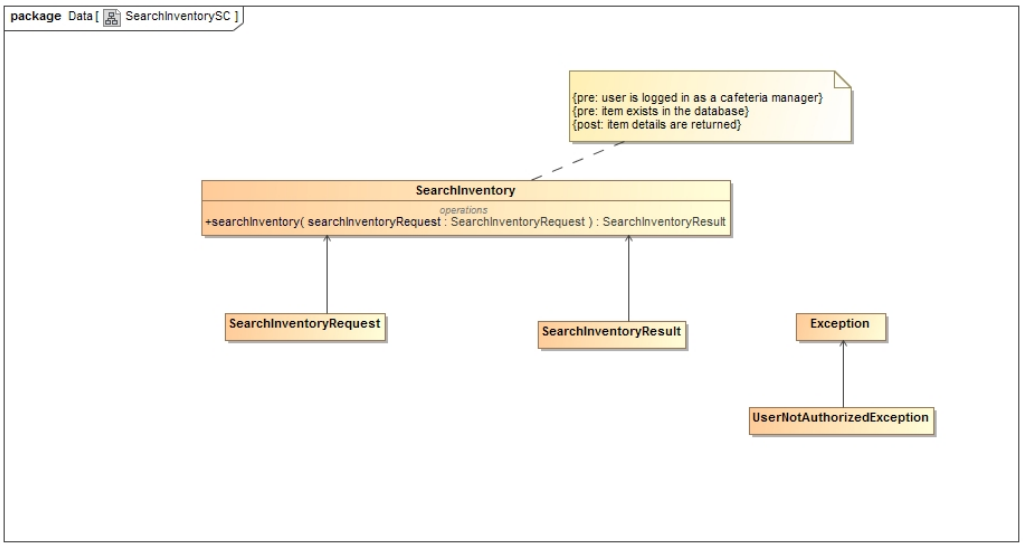

Figure 44: The service contract for searching for inventory

deleteInventory

The service contract and activity diagram for deleteInventory to follow. deleteInventory falls under the use case for Inventory (refer to page 36 figure 40 to view this use case diagram). The cafeteria manager will be able to delete inventory items via the allocated page.

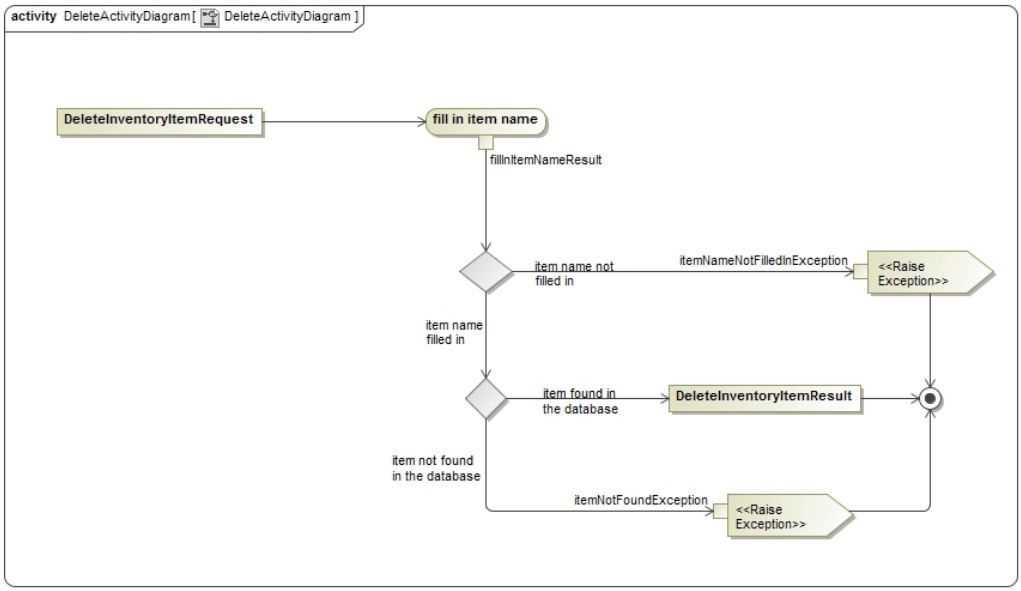

Figure 45: The activity diagram for adding inventory

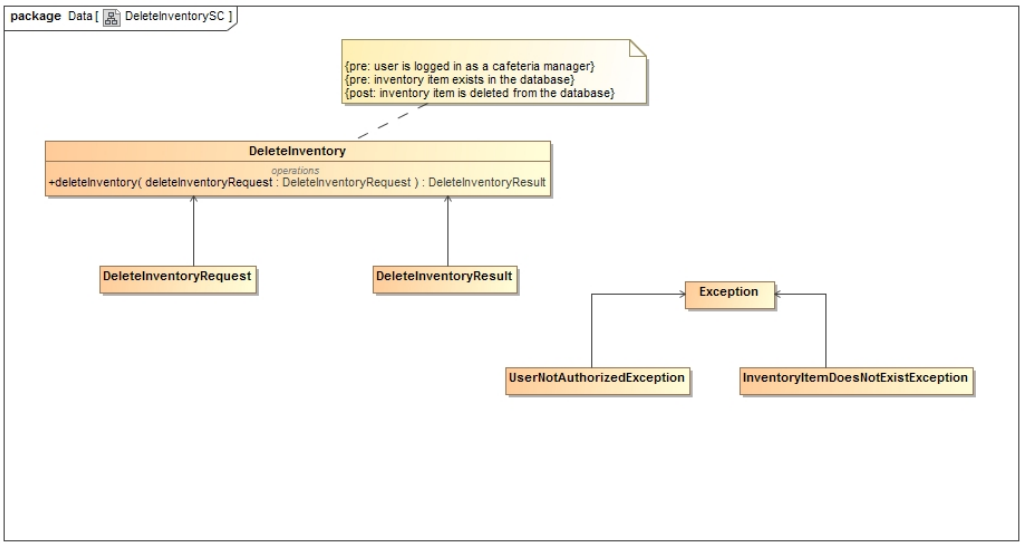

Figure 46: The service contract for adding inventory

updateInventory

The service contract and activity diagram for updateInventory to follow. updateInventory falls under the use case for Inventory (refer to page 36 figure 40 to view this use case diagram). The cafeteria manager will be able to update inventory items via the allocated page.

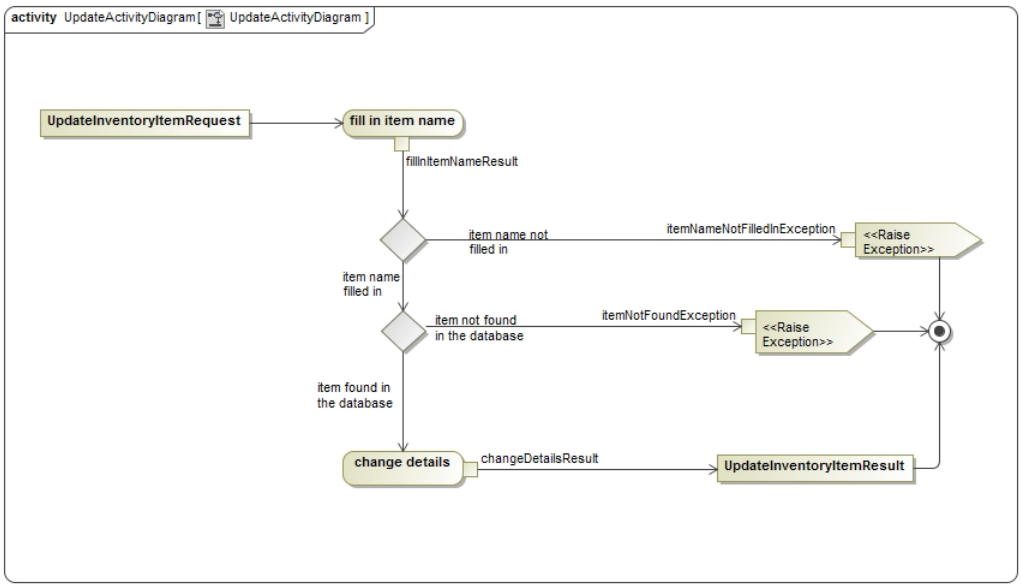

Figure 47: The activity diagram for updating inventory

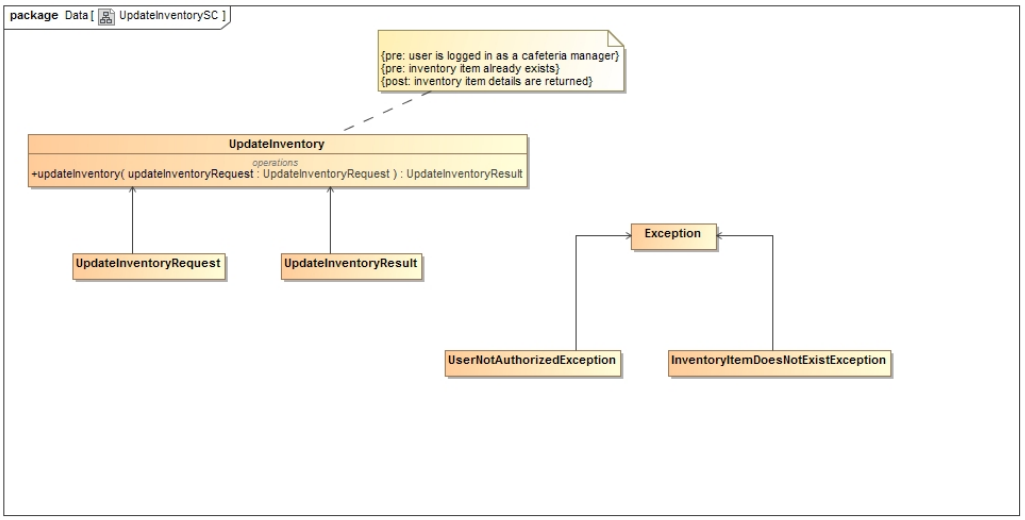

Figure 48: The service contract for updating inventory

5.6 Manage Profile Module

The user will be allowed to customize various settings such as resetting his/her personal limit, changing password and email, displaying the bill and viewing favourites. The following use case diagram indicates the above men-

tioned functionality around managing the profile. Hence, it has the use cases generateFavourite and edit profile.

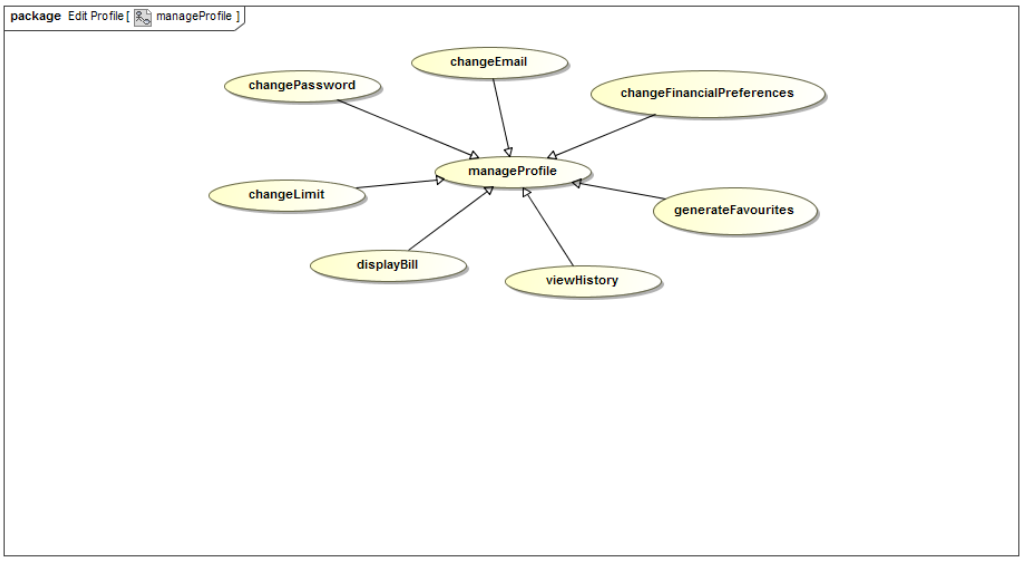

Figure 49: The use case diagram for managing profile

setFinancialPreferences

The service contract and activity diagram for setFinancialPreferences to follow. setFinancialPreferences falls under the use case for Manage Profile (refer to page 42 figure 49 to view this use case diagram). The user will be able to choose where their monthly bill is sent to.

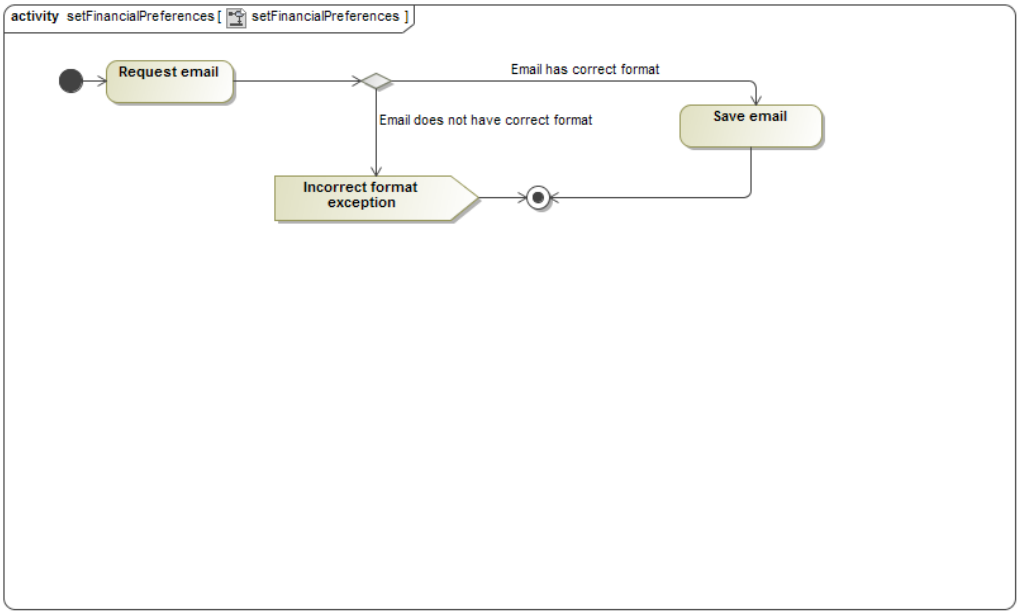

Figure 50: The activity diagram for setting financial preferences

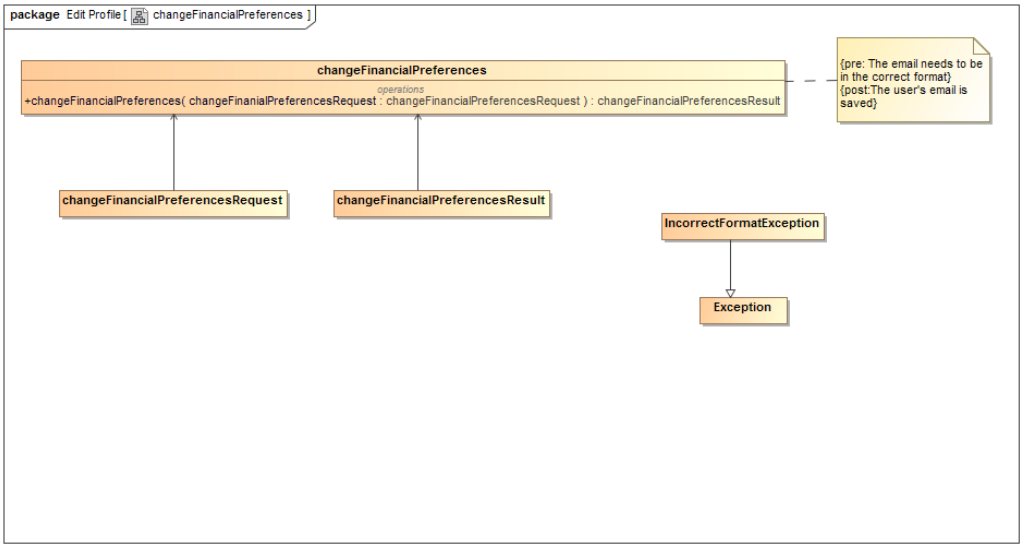

Figure 51: The service contract for setting financial preferences

viewHistory

The service contract and activity diagram for viewHistory to follow. viewHistory falls under the use case for Manage Profile (refer to page 42 figure 49

to view this use case diagram). viewHistory displays the order history of the user

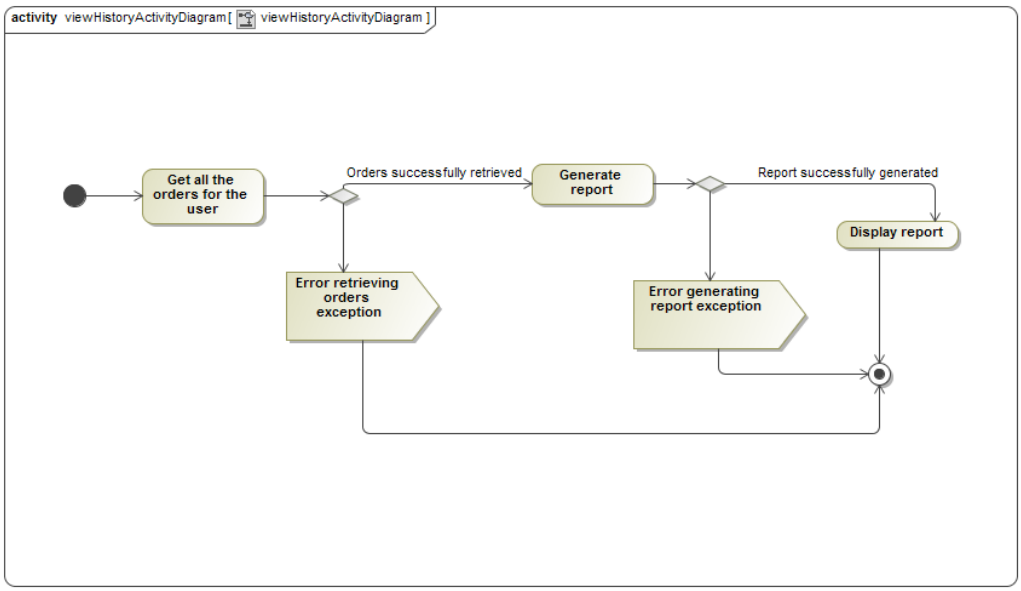

Figure 52: The activity diagram for view history

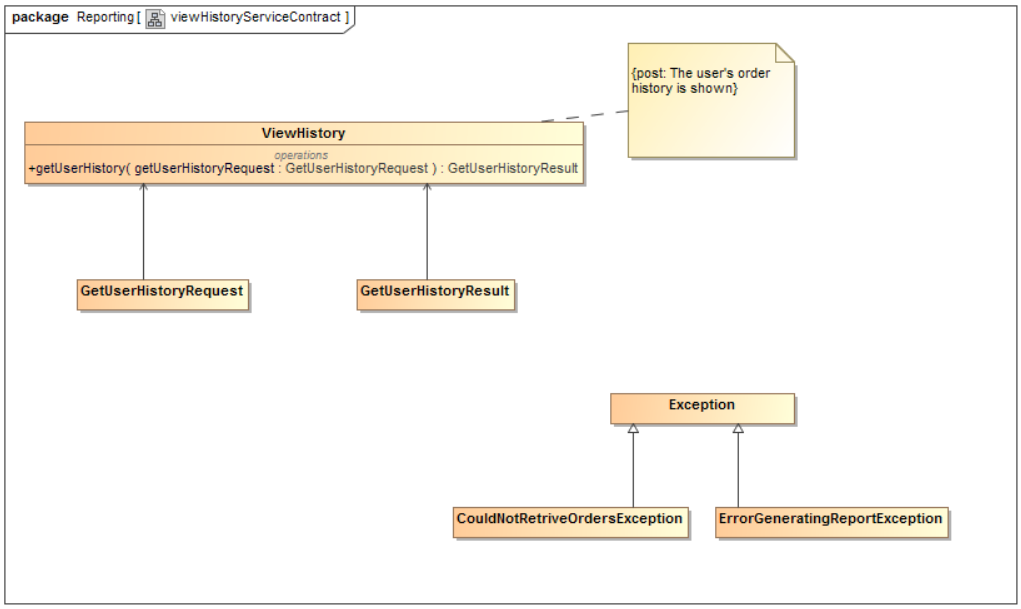

Figure 53: The service contract for view history

changePassword

The service contract and activity diagram for changePassword follow. changePassword falls under the use case for Manage Profile (refer to page 42 figure 49 to view this use case diagram). The user will be able to edit their password.

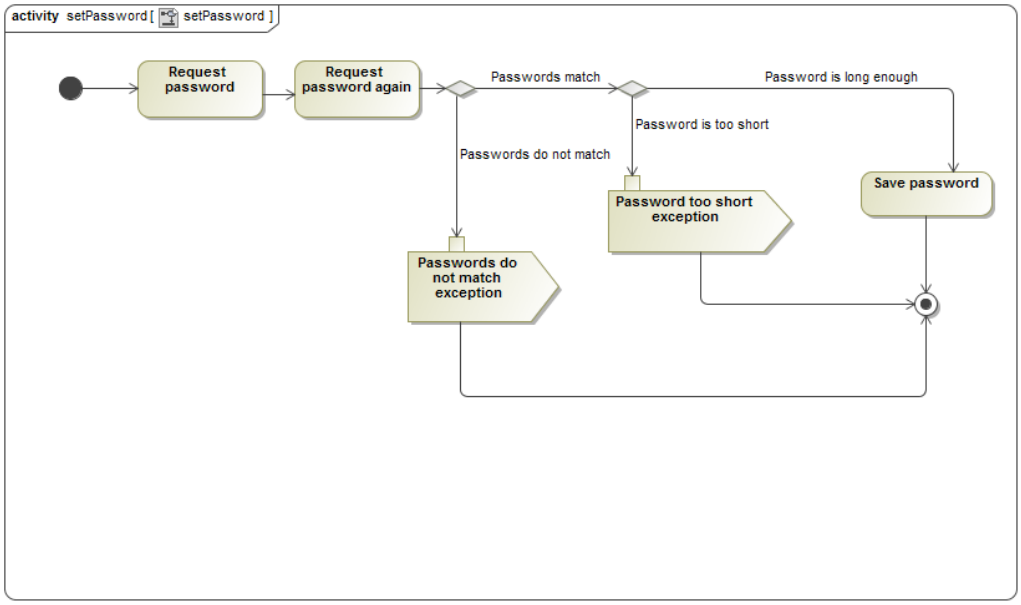

Figure 54: The activity diagram for changing password

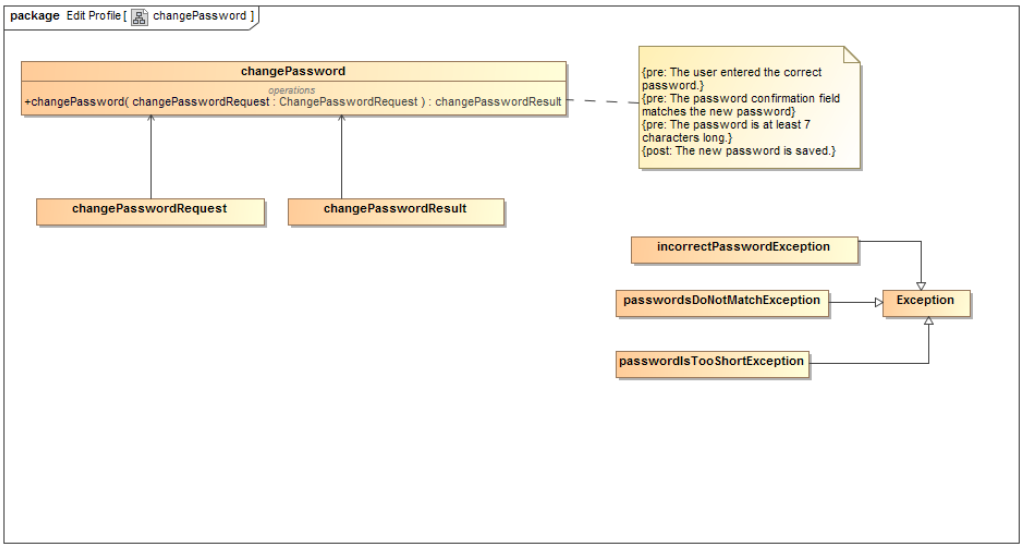

Figure 55: The service contract for changing password

changeEmail

The service contract and activity diagram for changeEmail follow. changeEmail falls under the use case for Edit Profile (refer to page 42 figure 49 to view this use case diagram). The user will be able to edit their email address.

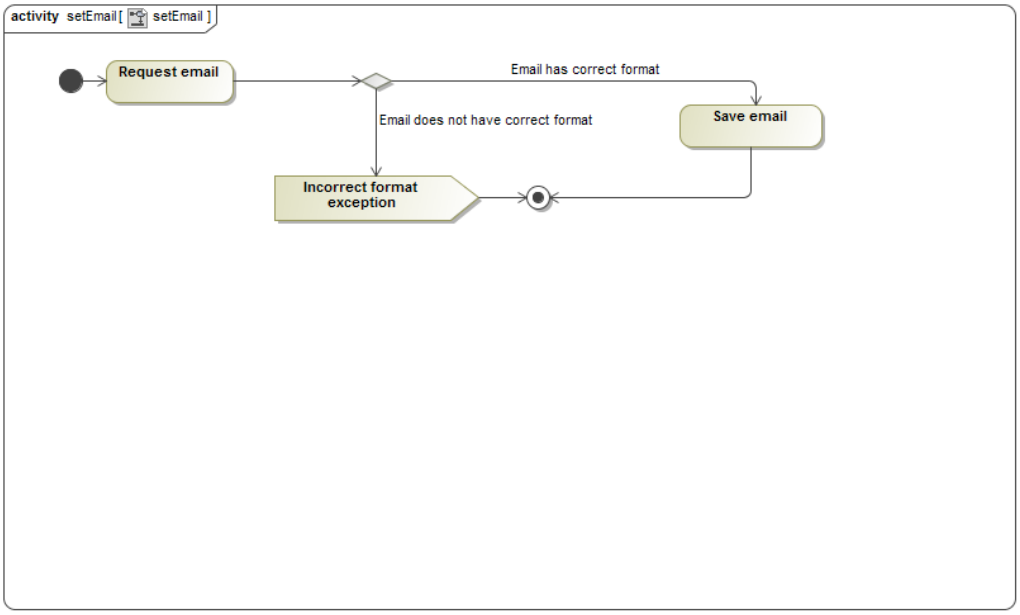

Figure 56: The activity diagram for changing email address

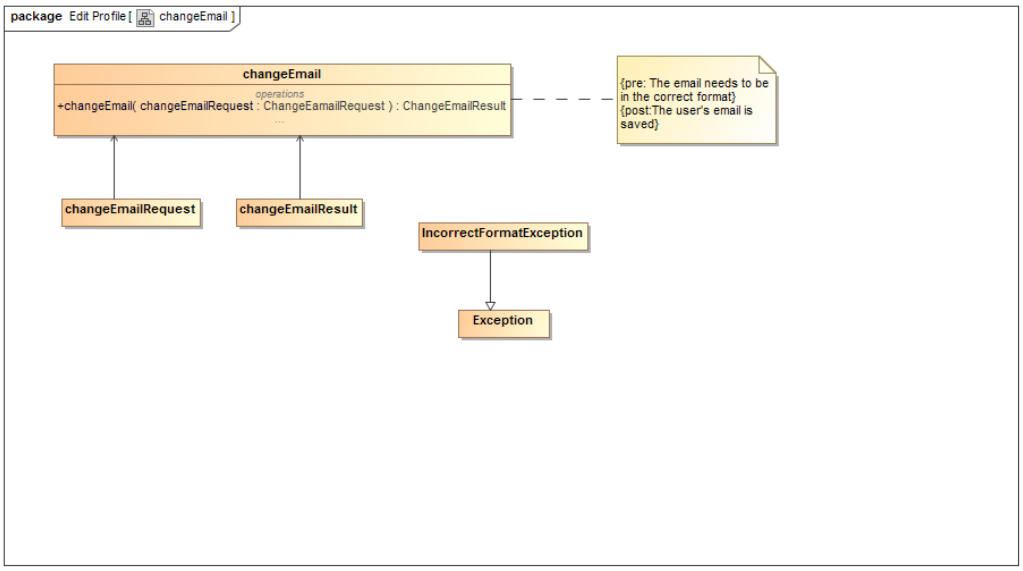

Figure 57: The service contract for changing email address

changeLimit

The service contract and activity diagram for changeLimit follow. changeLimit falls under the use case for Manage Profile (refer to page 42 figure 49 to view this use case diagram). The user will be able to change their personal spending limit on their profile, however, this must not exceed the system's limit.

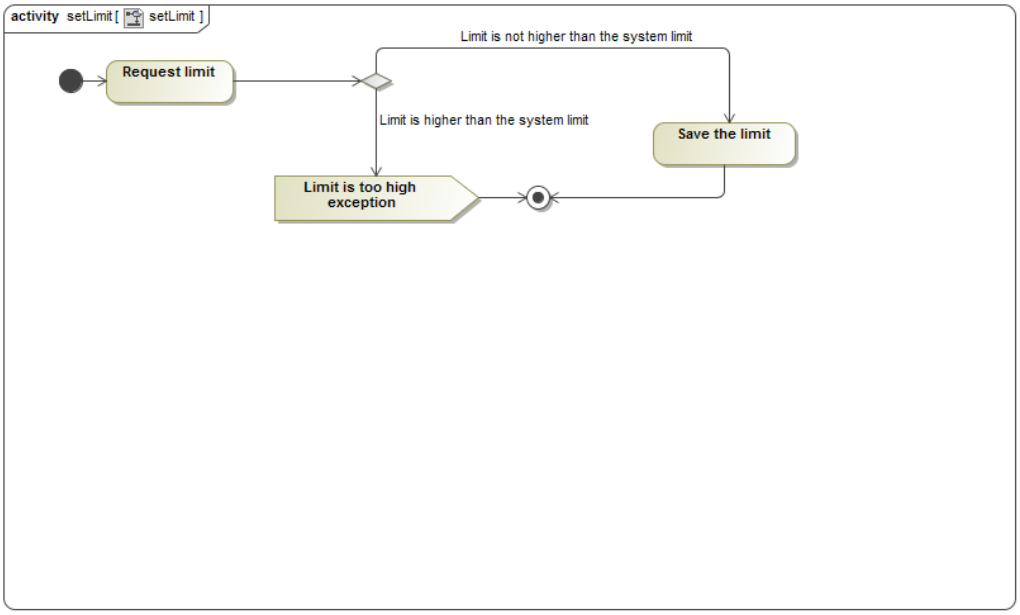

Figure 58: The activity diagram for setting limit

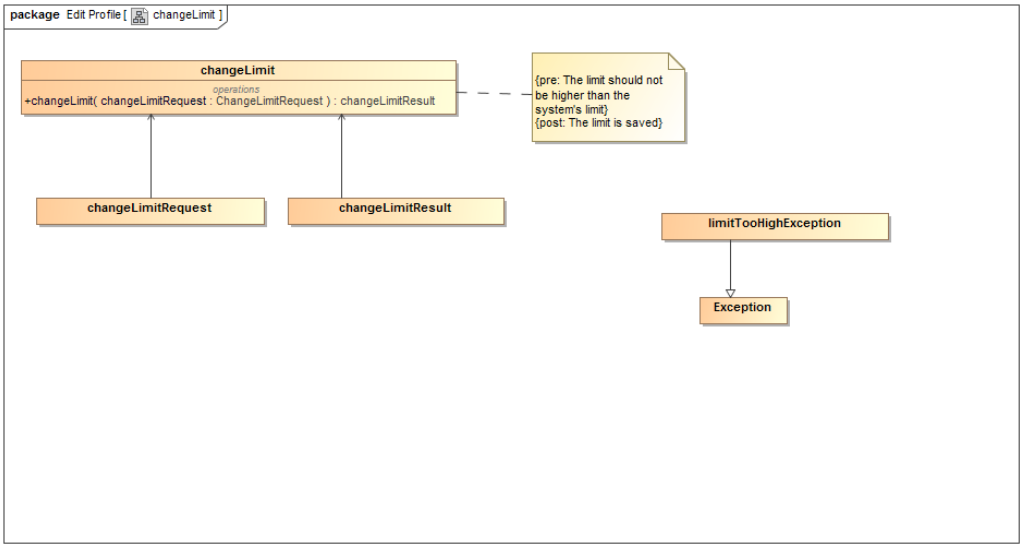

Figure 59: The service contract for setting limit

5.7 Manage System Module

The superuser will be allowed to customize various settings such as assign roles to the users, changing employee IDs, setting the maximum spending

limit as well as branding settings such as setting the canteen name and changing the cover photo. The super user will also be able to audit the system - see all actions performed on the system by the different users. The following use case diagram indicates the above mentioned functionality.

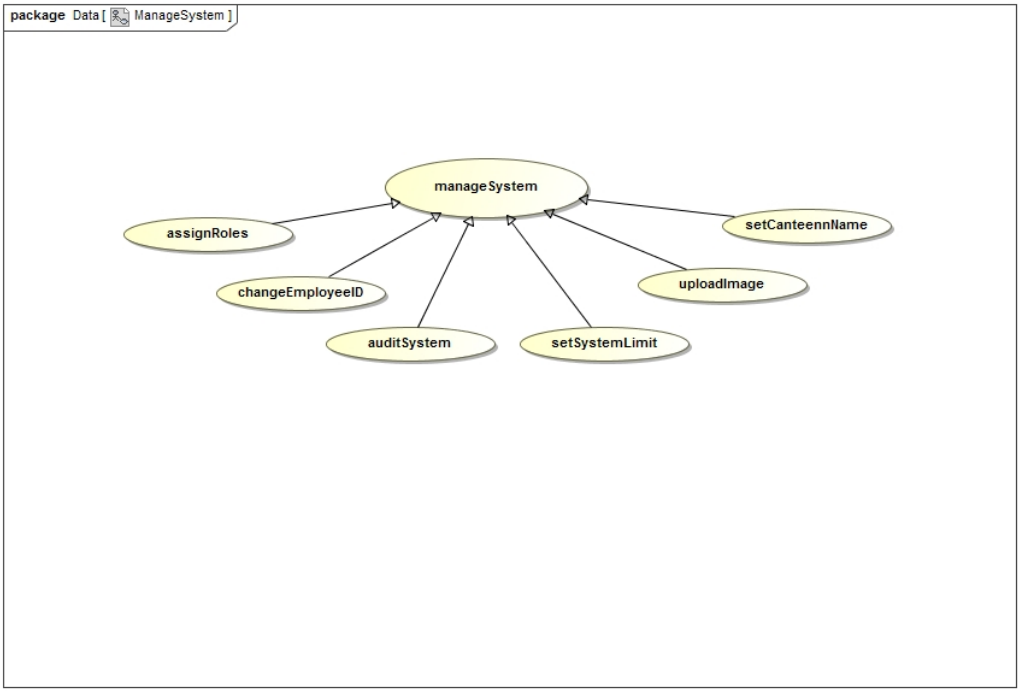

Figure 60: The use case diagram for managing system

assignRoles

The service contract and activity diagram for assignRoles to follow. assignRoles falls under the use case for Manage System (refer to page 49 figure 60 to view this use case diagram). The superuser is the only user who can allocate the various roles to users. Roles consist of finance managers, cashiers, cafeteria managers, and administrator.

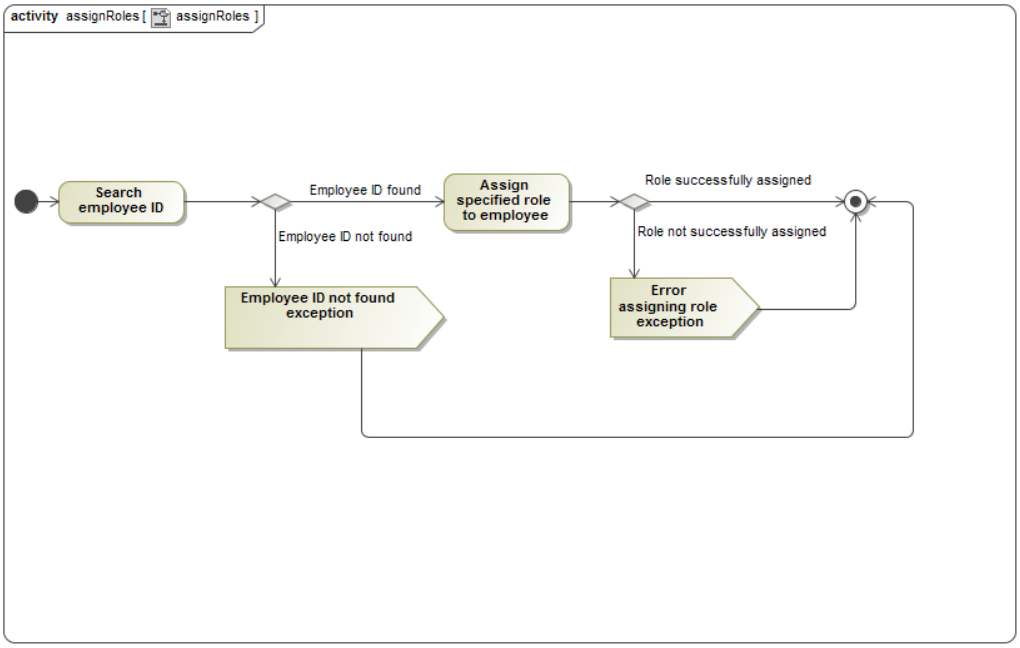

Figure 61: The activity diagram for assign roles

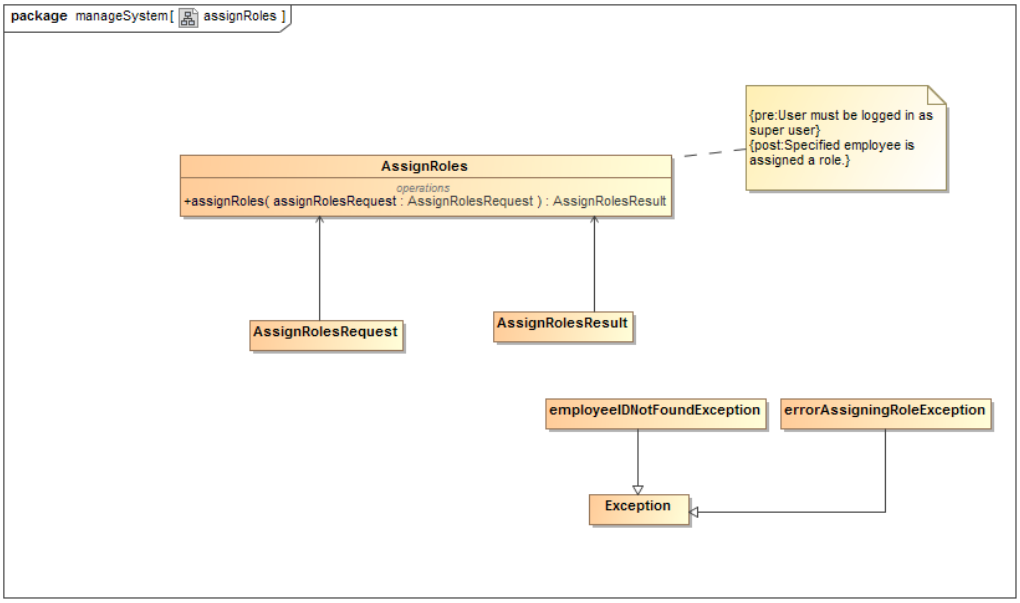

Figure 62: The service contract for assign roles

auditSystem

The service contract and activity diagram for auditSystem to follow. auditSystem falls under the use case for Manage System (refer to page 49 figure 60 to view this use case diagram). The superuser and the admin user are the only users who can access the audit of the system. This constitutes all actions performed by the different users of the system. This is to keep a record of what actions are performed by who on the system.

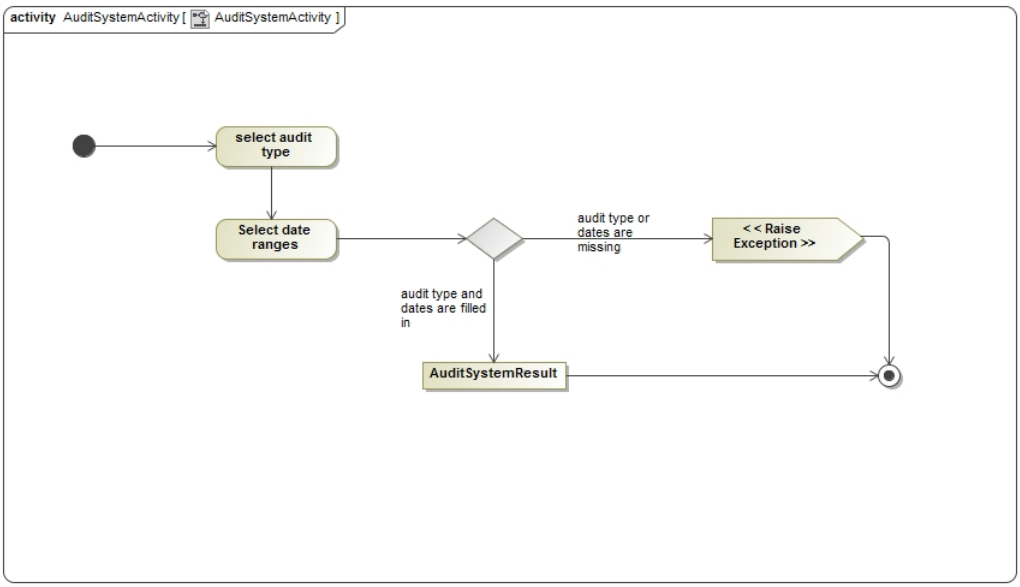

Figure 63: The activity diagram for audit system

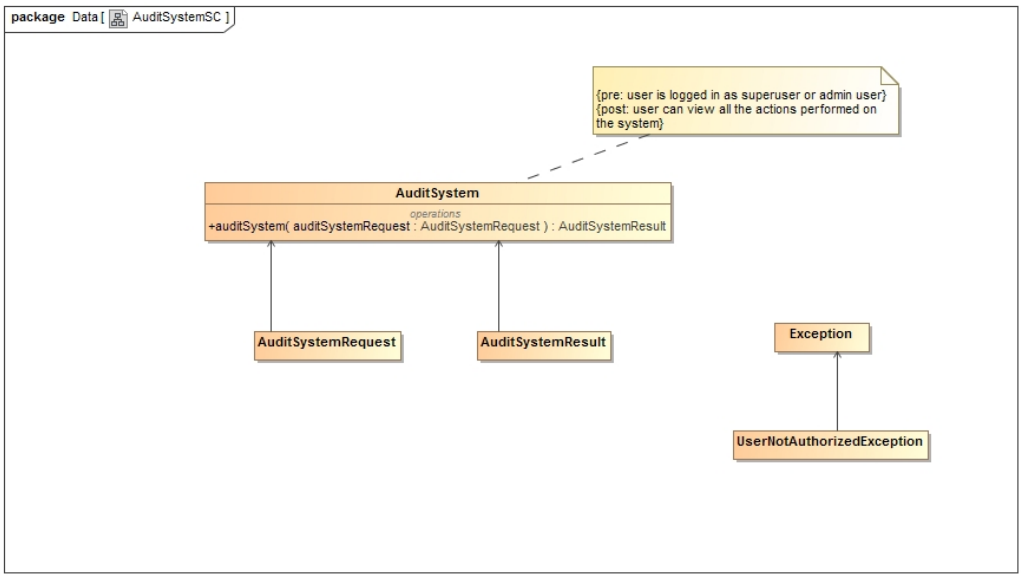

Figure 64: The service contract for audit system

changeEmployeeID

The service contract and activity diagram for changeEmployeeID to follow. changeEmployeeID falls under the use case for Manage System (refer to page 49 figure 60 to view this use case diagram). The superuser is the only user who can change the employee ID of employees if they typed their ID incorrectly or if the company changes the IDs.

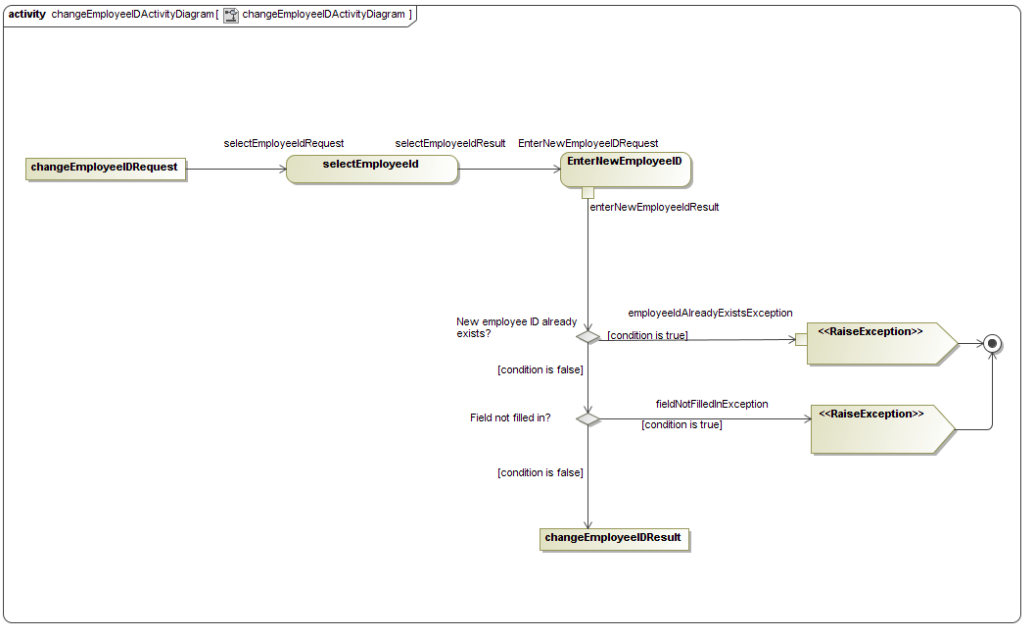

Figure 65: The activity diagram for change employee ID

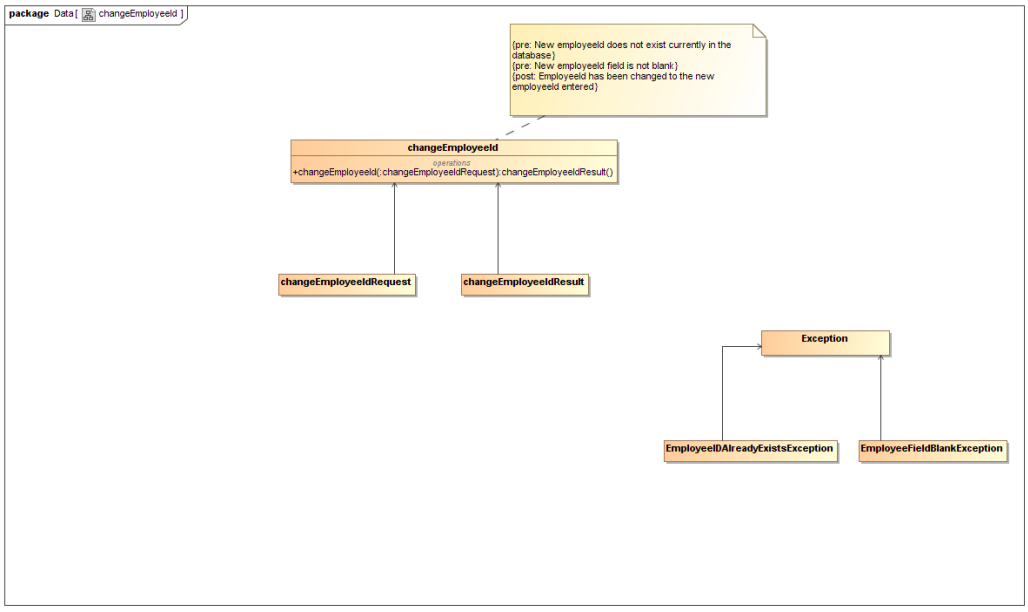

Figure 66: The service contract for change employee ID

setSystemWideLimit

The service contract and activity diagram for setSystemWideLimit to follow. setSystemWideLimit falls under the use case for Manage System (refer to page 49 figure 60 to view this use case diagram). The superuser is the only user who can change the maximum monthly spending limit. Users will then not be able to set their personal monthly spending limits to a value higher than this.

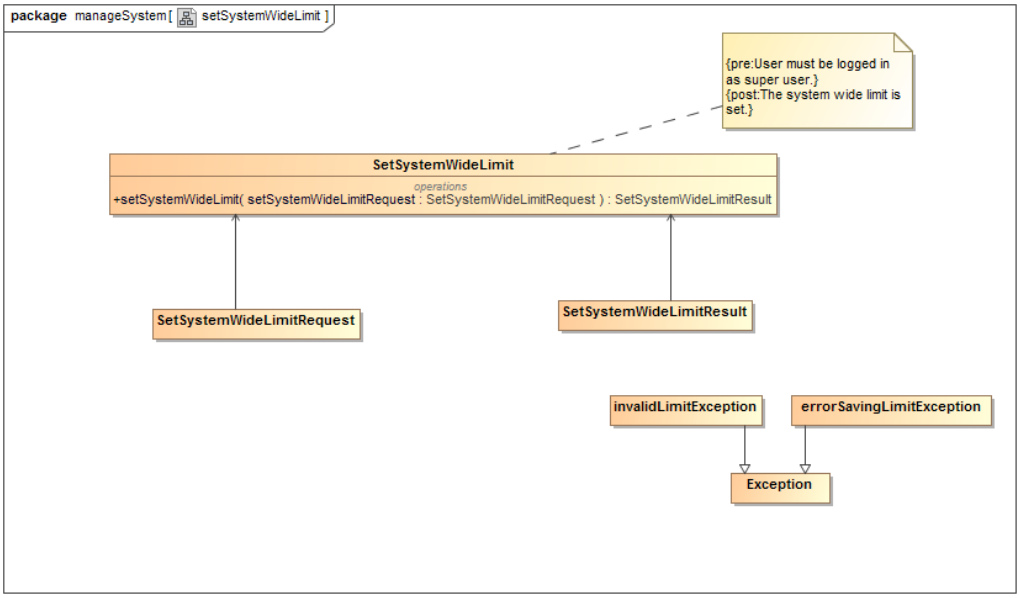

Figure 67: The activity diagram for setting the system wide limit

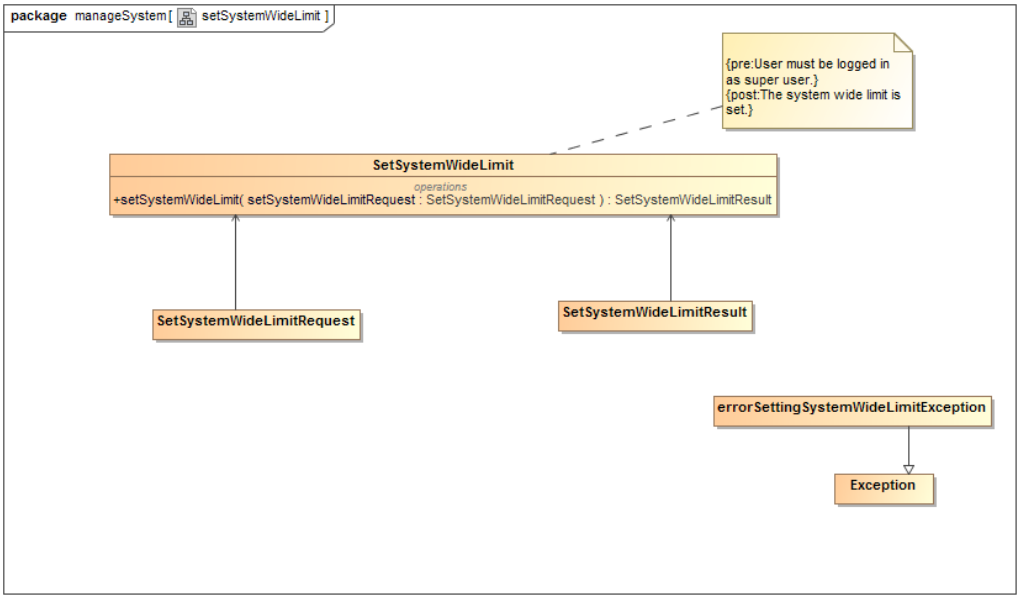

Figure 68: The service contract for setting the system wide limit

setCanteenName

The service contract and activity diagram for setCanteenName to follow. setCanteenName falls under the use case for Manage System (refer to page 49 figure 60 to view this use case diagram). The superuser can change the canteen name hence not restricting the system to be used at only one canteen - making the system portable.

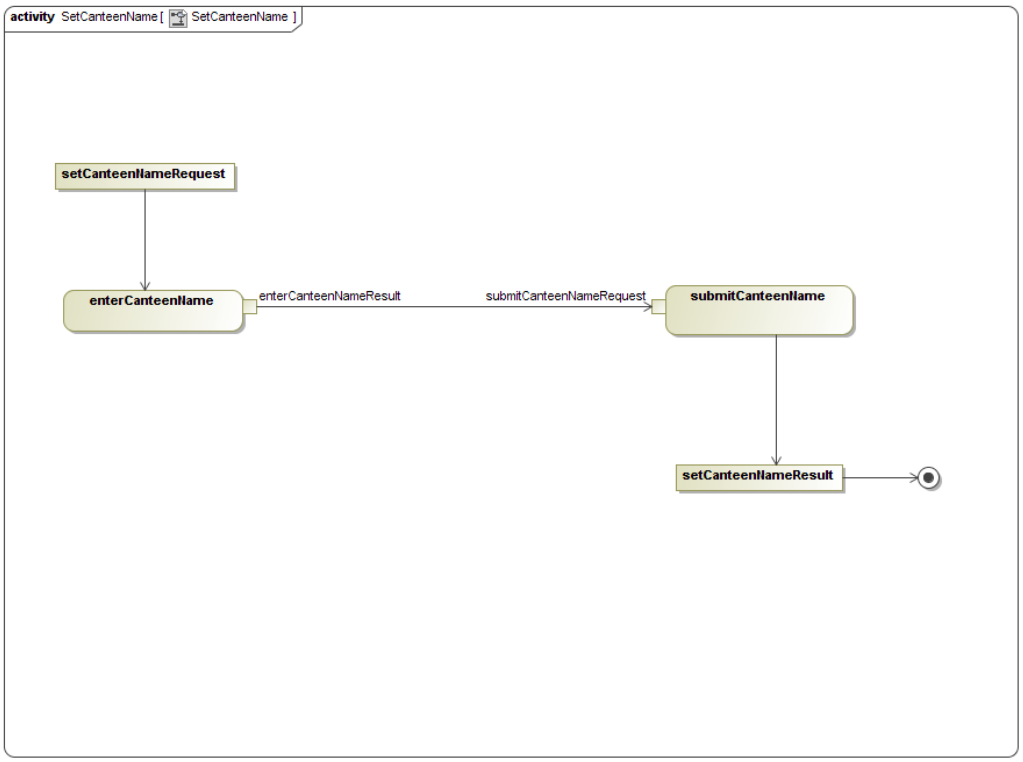

Figure 69: The activity diagram for setting the canteen name

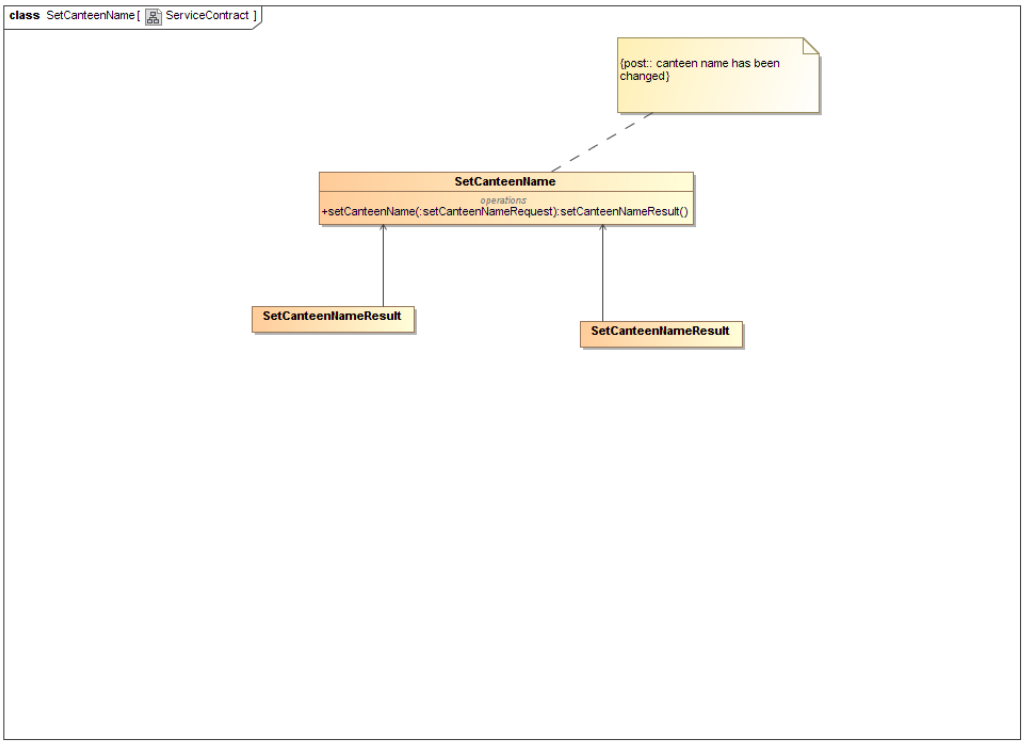

Figure 70: The service contract for setting the canteen name

uploadImage

The service contract and activity diagram for uploadImage to follow. uploadImage falls under the use case for Manage System (refer to page 49 figure 60 to view this use case diagram). The superuser can change the canteen cover photo hence not restricting the system to be used at only one canteen - making the system portable.

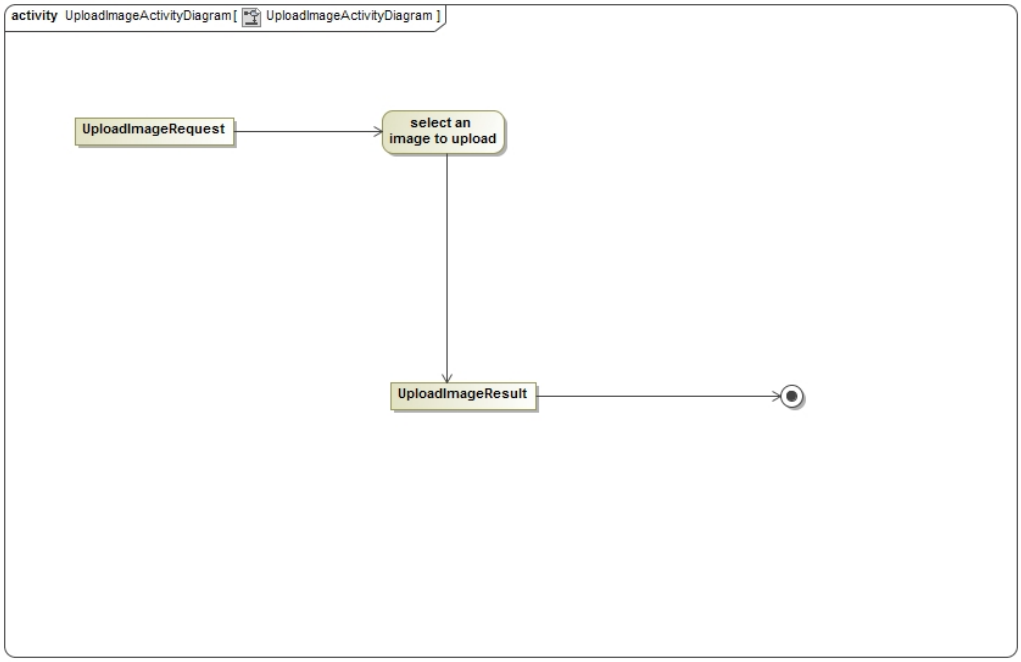

Figure 71: The activity diagram for uploading a cover image

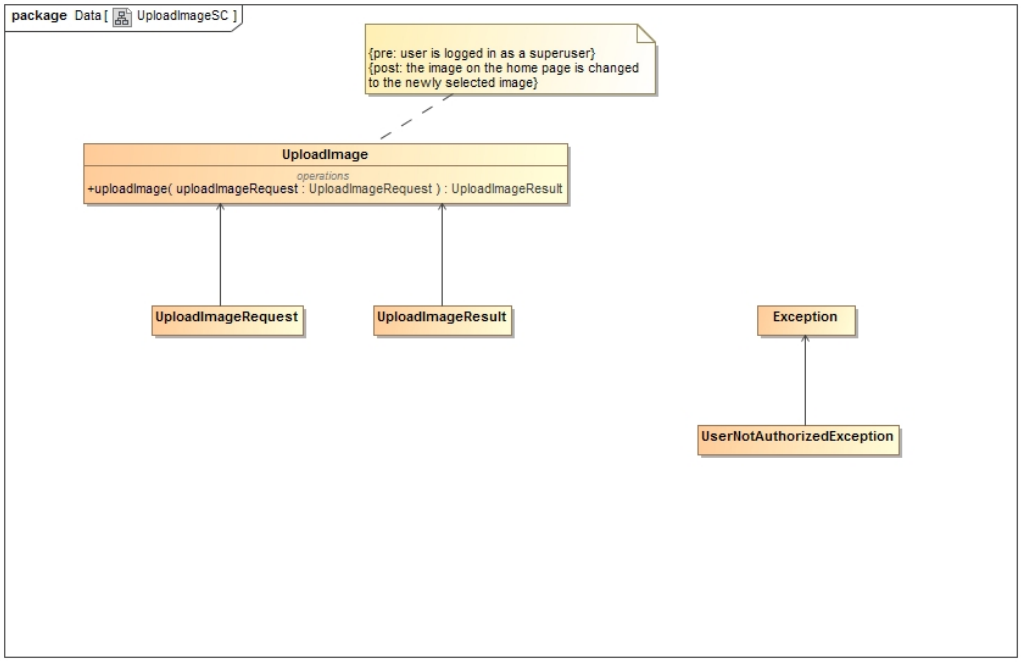

Figure 72: The service contract for uploading a cover image

5.8 Reporting Module

In this module, the functionality provided consists of checking how much the user spent for a given time period, creating statistics of how much of the inventory items was used up and popular menu items. In addition, the reporting module also encompasses notifications that are sent to the user when their order is ready (via email and on their profile).

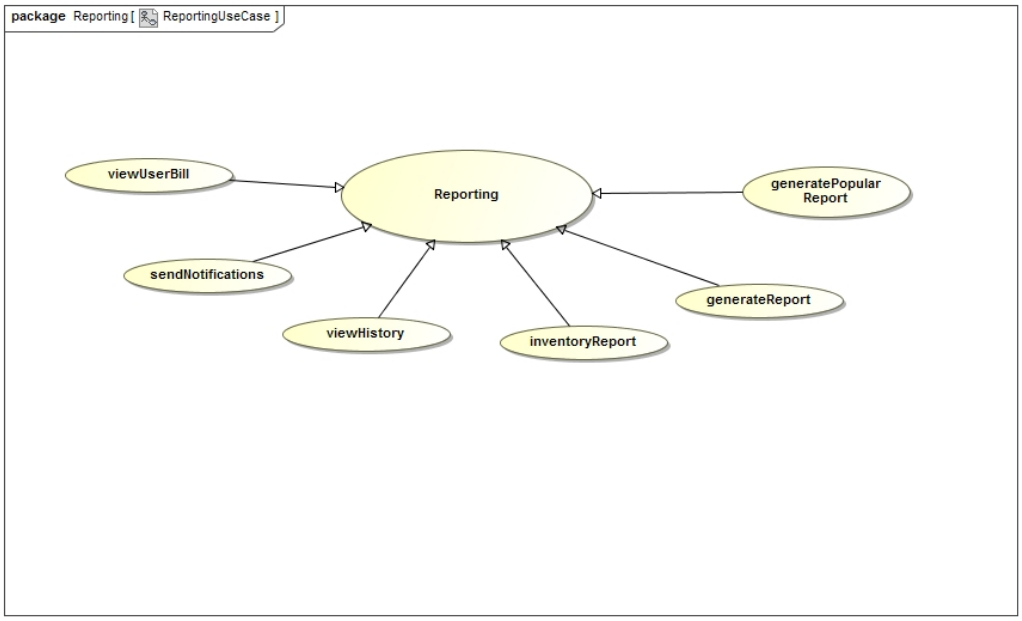

Figure 73: The main use case diagram for Reporting

Another crutial part of the authentication module is to verify if a user may have access to certain pages given his/her role which will also contribute to the security of the data stored on our system and to control which user has access to which functionality.

Inventory Report

The service contract and activity diagram for inventoryReport follow. inventoryReport falls under the use case for Reporting (refer to page 59 - figure 73 to view this use case diagram). Here, the cafeteria manager can generate statistics to see which inventory items are used most. This information can be used to predict which itmes should be bought more often and when.

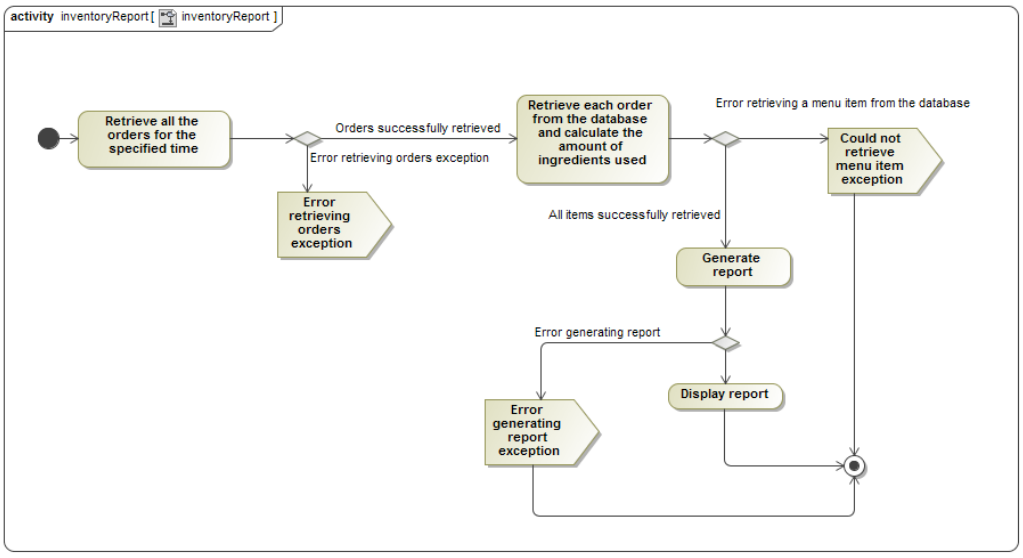

Figure 74: The activity diagram for inventoryReport

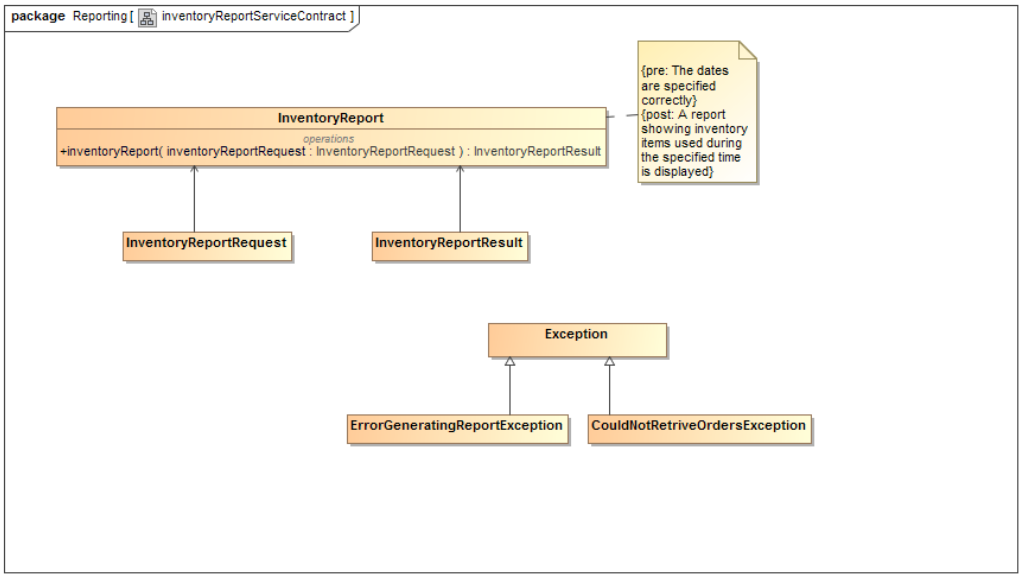

Figure 75: The service contract for inventoryReport

Generate Report

The service contract and activity diagram for generateReport follow. generateReport falls under the use case for Reporting (refer to page 59 - figure 73 to view this use case diagram). Generate report generates a graph that

displays how many menu items were sold in each category and how many of a specific menu item was orederd, all during a specific time period.

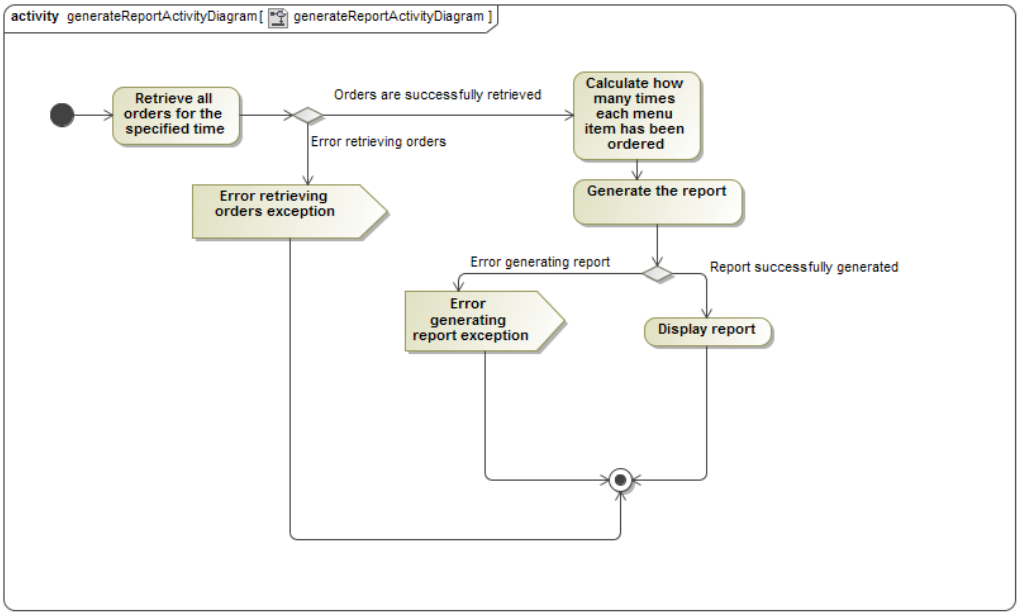

Figure 76: The activity diagram for generateReport

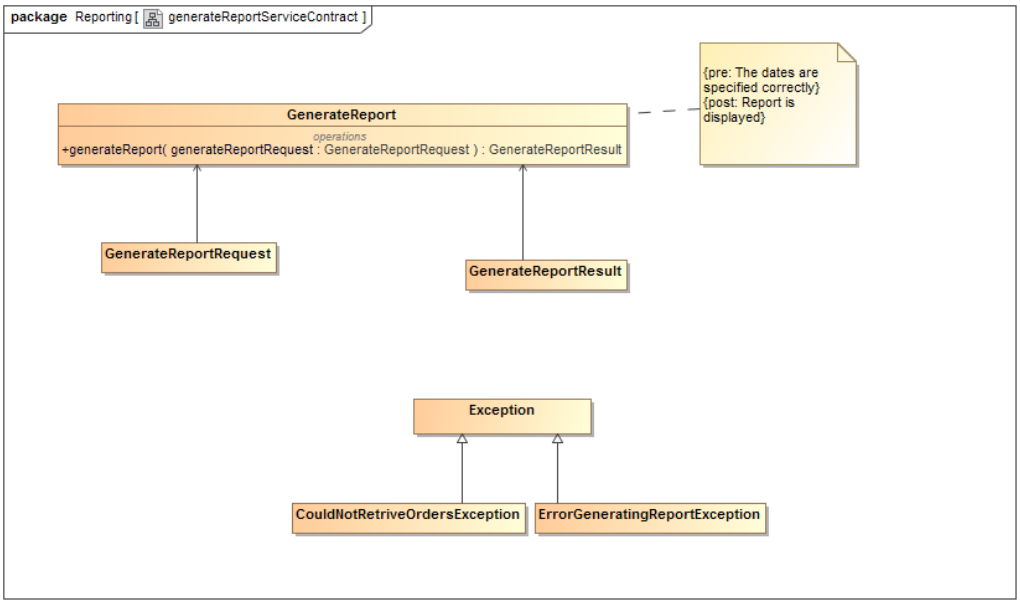

Figure 77: The service contract for generateReport

Generate Popular Report

The service contract and activity diagram for generatePopularReport follow. generatePopularReport falls under the use case for Reporting (refer to page 59 - figure 73 to view this use case diagram). Generate popular report generates a graph that displays a specific number of menu items that were ordered the most over a specified time.

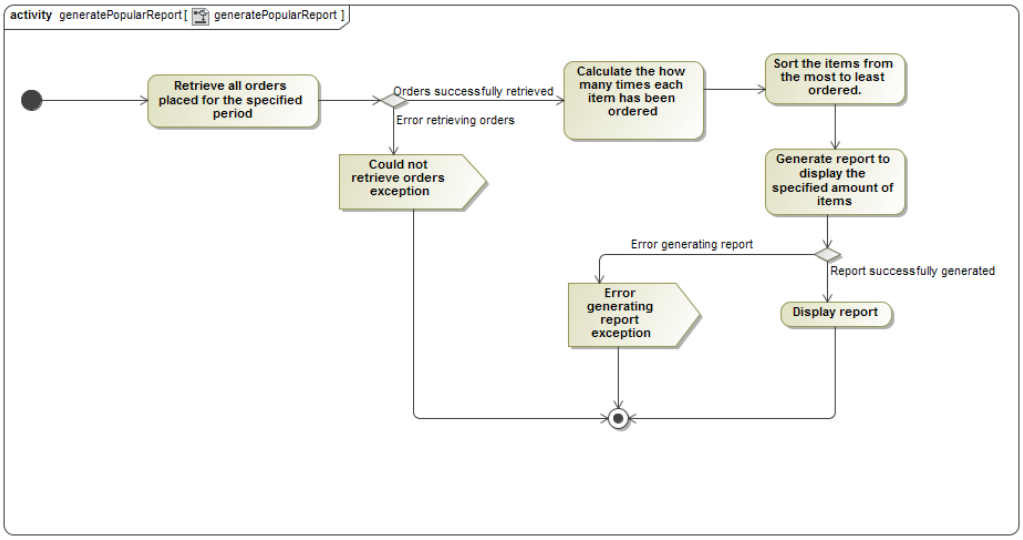

Figure 78: The activity diagram for generatePopularReport

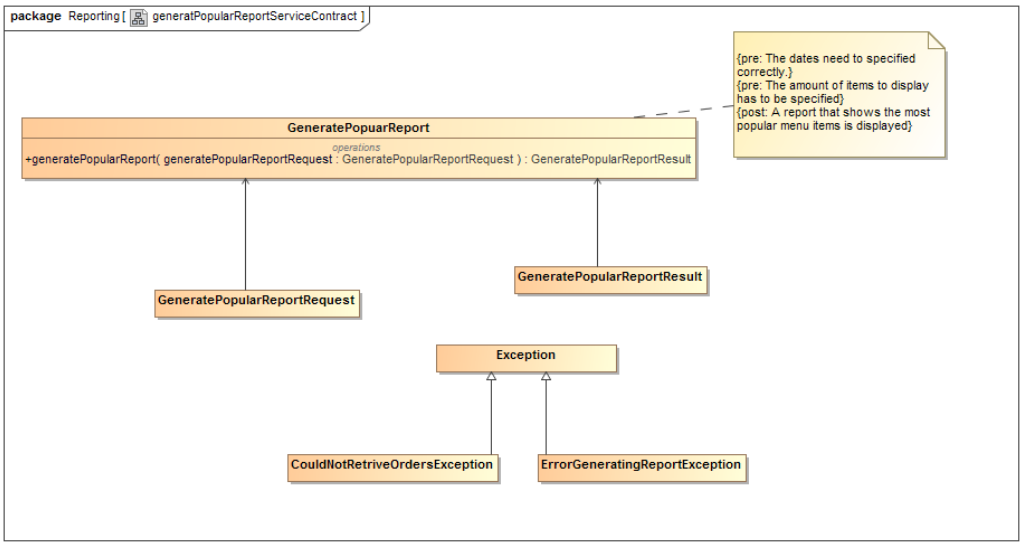

Figure 79: The service contract for generatePopularReport

Send Notifications

The service contract and activity diagram for sendNotifications follow. sendNotifications falls under the use case for Reporting (refer to page 59 - figure 73 to view this use case diagram). Here, when an order is ready, the cashier can let the client know their order is ready by sending them a notification (via email or on their profile).

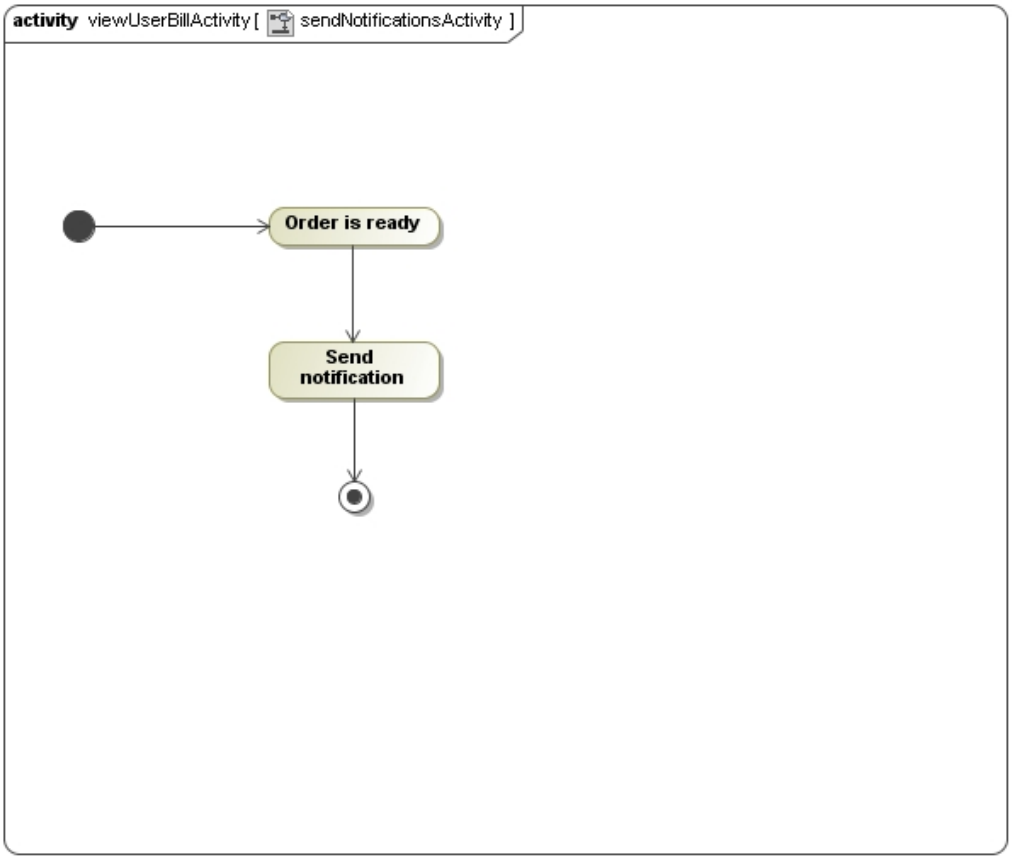

Figure 80: The activity diagram for sendNotifications

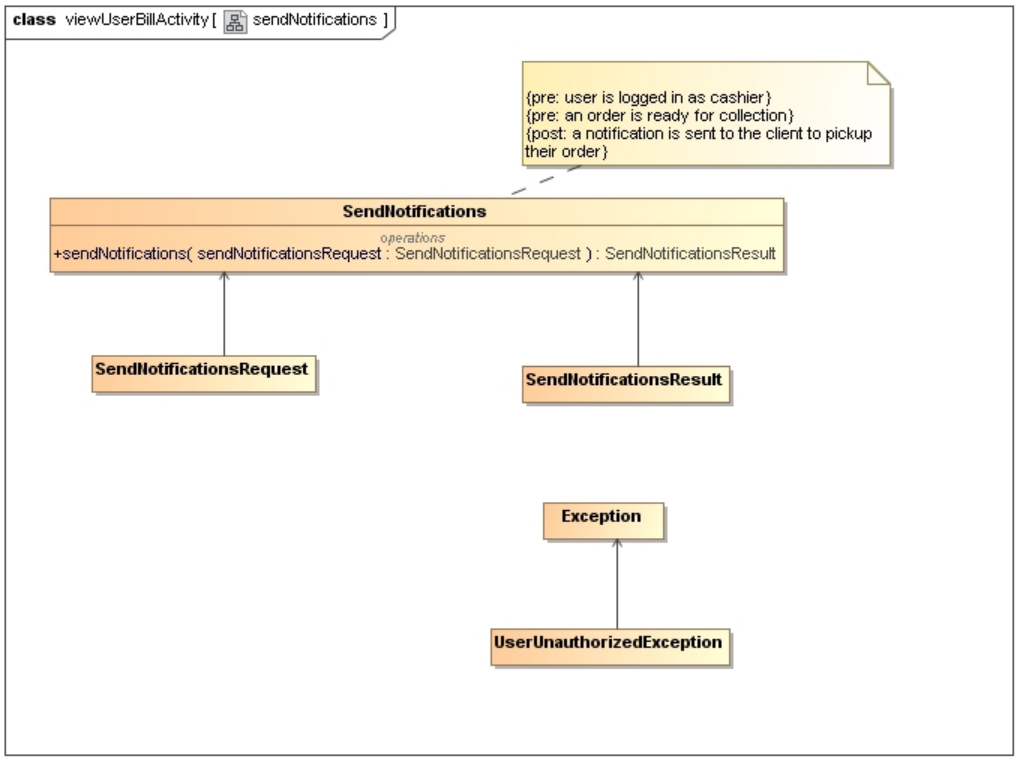

Figure 81: The service contract for sendNotifications

View History

The service contract and activity diagram for viewHistory follow. viewHistory falls under the use case for Reporting (refer to page 59 - figure 73 to view this use case diagram). Here, the user can check on their profile what orders they have placed in the past - for refence when trying to decide what they like and what they don't or just to check what they have ordered before so they don't order it again.

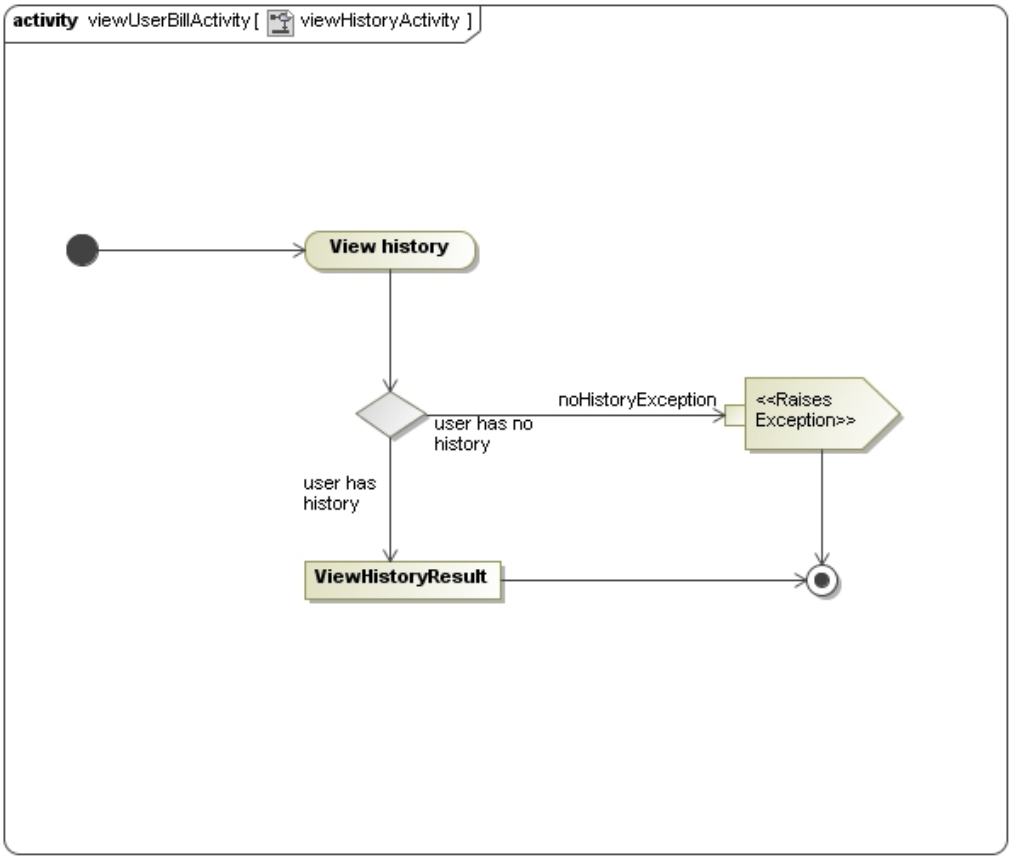

Figure 82: The activity diagram for viewHistory

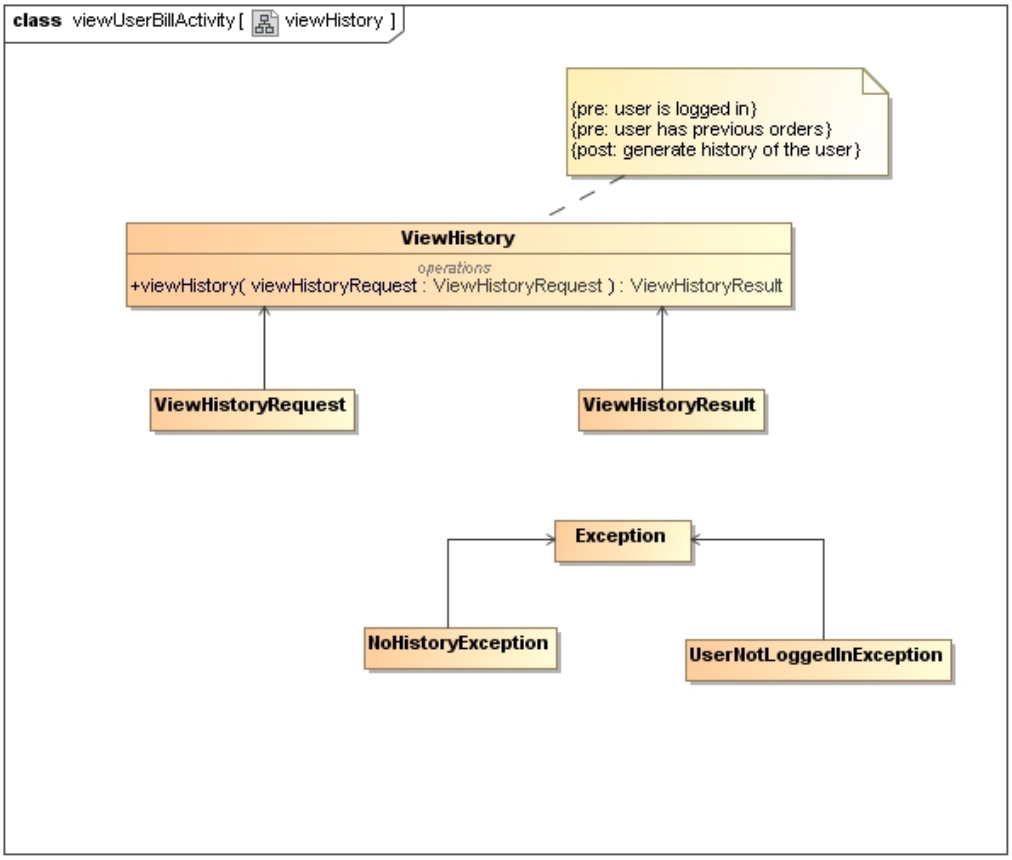

Figure 83: The service contract for viewHistory

View User Bill

The service contract and activity diagram for viewUserBill follow. generateStatistics falls under the use case for Reporting (refer to page 59 - figure 73 to view this use case diagram). Here, the finance person can check a user's bill for a certain time period. They can then see how much the user owes (to deduct it from their salary at the end of the month).

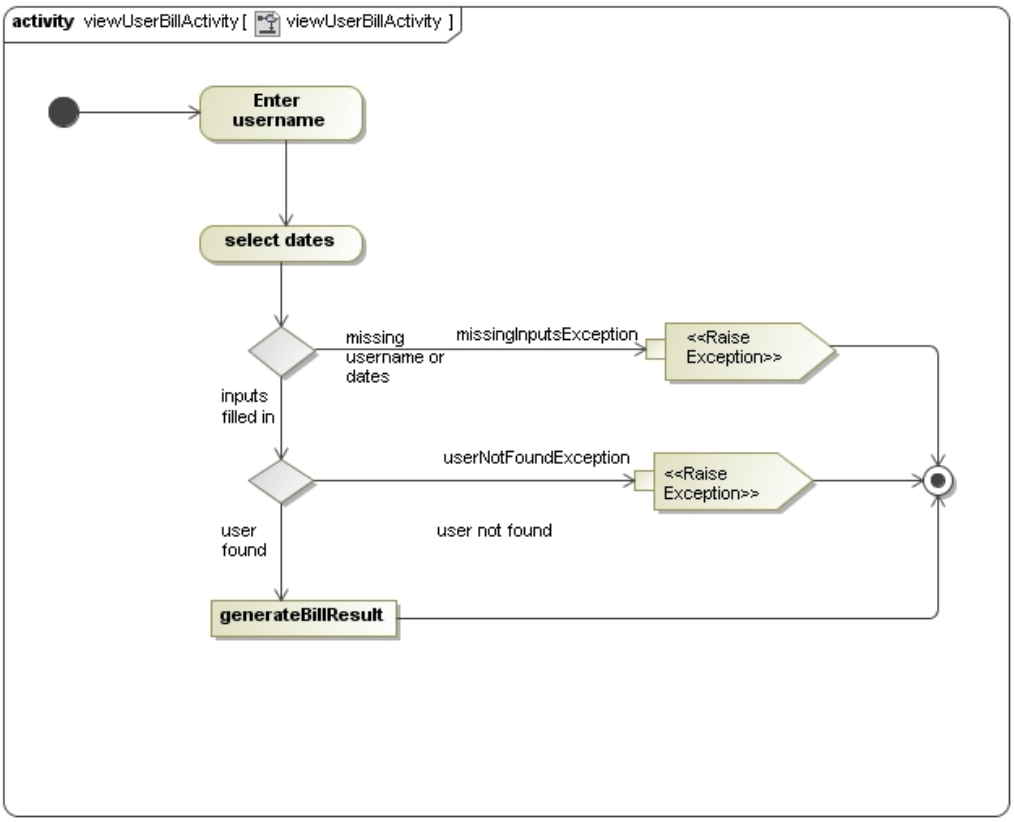

Figure 84: The activity diagram for viewUserBill

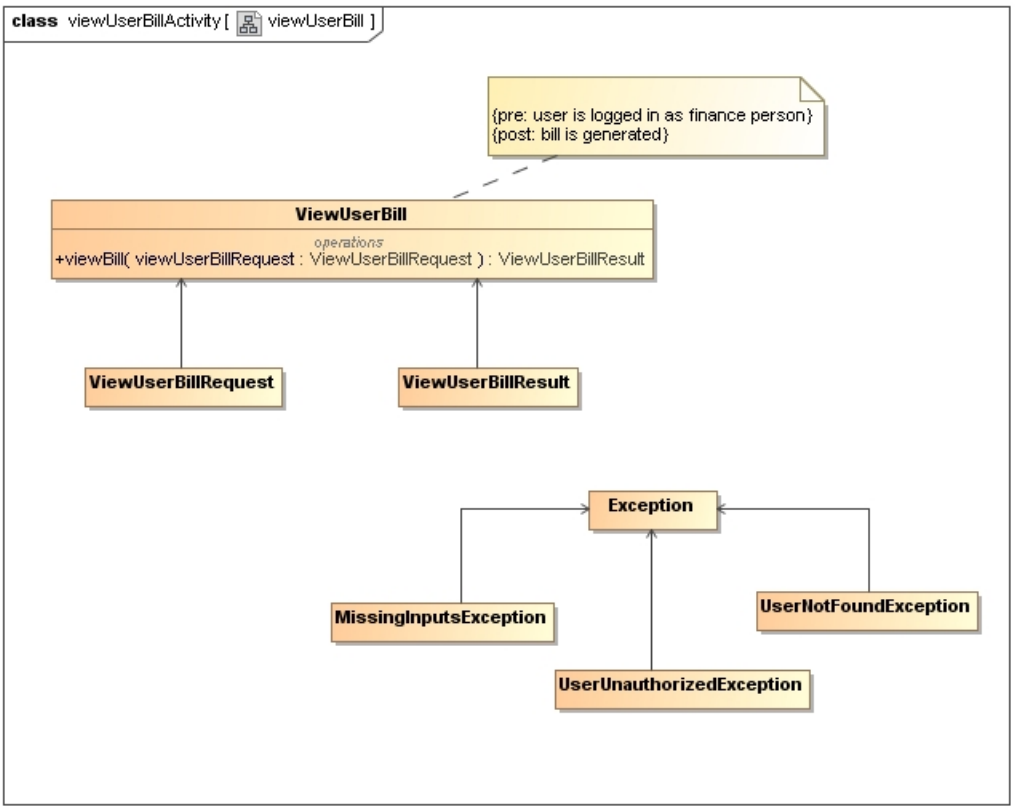

Figure 85: The service contract for viewUserBill

6 Comment

No comment for this version update.

Version	Date	Summary	Authors
0.0.1	29 May 2015	First draft contains first two use cases	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel,
0.0.2	6 July 2015	Second draft adding Register and Authentication use cases and updated profile	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel
0.0.3	22 July 2015	Third draft adding Inventory and Menu use cases	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel
0.0.4	23 July 2015	Fourth draft adding Manage system and editing whole document	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel
0.0.5	25 August 2015	Fifth draft adding and editing Place Order module	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel
0.0.6	25 September 2015	Sixth draft adding and editing Reporting module	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel
0.0.7	28 October 2015	Seventh draft adding statistics, history and auditing module	Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael Isabel Nel