# Functional Requirements Document Specification
# Project:
# Cafeteria Management System:
# Resolve

T-RISE
Rendani Dau (13381467)
Elana Kuun (12029522)
Semaka Malapane (13081129)
Antonia Michael (13014171)
Isabel Nel (13070305)

https://github.com/toniamichael94/MainProjectCOS301

July 7, 2015

# Contents

| | |
|---|---|
| **Document Title** | Functional Requirements Document |
| **Document Identification** | Document 0.0.2 |
| **Author** | Rendani Dau, Elana Kuun, Semaka Malapane, Antonia Michael, Isabel Nel |
| **Version** | 0.0.2 |
| **Document Status** | Second Version - added the Register and Authentication use cases and updated profile |

| 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 |

# 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 incremential method of managing and designing the system.

# 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

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 makes it inconvenient for employees as they have to have cash on hand if they want to purchase anything from the cafeteria. Employees might choose to buy somewhere else where they can use another form of payment. The employees have to use fuel and time, and this does not bring in the maximum amount of income to the cafeteria, hindering its growth and improvement.

Resolve is therefore looking for a means to accept payments from employees for the canteen using their employee access cards or access card numbers. The amount spent at the cafeteria can then be deducted from their salary at the end of the month.

After our first meeting with the client, they brought to our attention that at times the cafeteria does not have enough stock to provide some of the menu items, therefore the managing of inventory and stock will also be part of the system. The system will also predict what inventory/stock needs to be bought for the next week in order to avoid shortcomings. At the end of

each month, the bill for that month will be sent to either payroll or to the employee. This option is configurable from the user's profile. The employee can also set a spending limit for each month. The system will also have a maximum limit that users cannot exceed.

# 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

- Register

- Manage Profile

- Place Order

- Notify

- Manage Inventory

- Report

## 4.2 Use Case Prioritization

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

### 4.2.1 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.

- Register
  This is a critical use case because you can not log into the system if you have not been registered on the system. Furthermore, if you are not logged in, you can not place orders.

- 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 activities relating to that.

### 4.2.2   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. 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 Inventory
  This use case is considered important, because it deals with incrementing stock that has been added and decrementing stock when it is purchased or expired. It also deals with the cafeteria manager marking orders as completed.

- Reporting
  This use case is considered important because users must be able to print a billing report. They should also have the option of sending their montlhly bill to payroll where it will be deducted from their salaries. A history of all the orders that have taken place on the system can also be printed.

### 4.2.3   Nice to have

- Notify
  This is classified as nice to have as it includes notifying a user when their order is ready for collection, as well as other notifications, such as notifying the cashier or cafeteria manager that they are low on certain inventory. The system is not dependant on this functionality.

# 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 integratibility of the system.

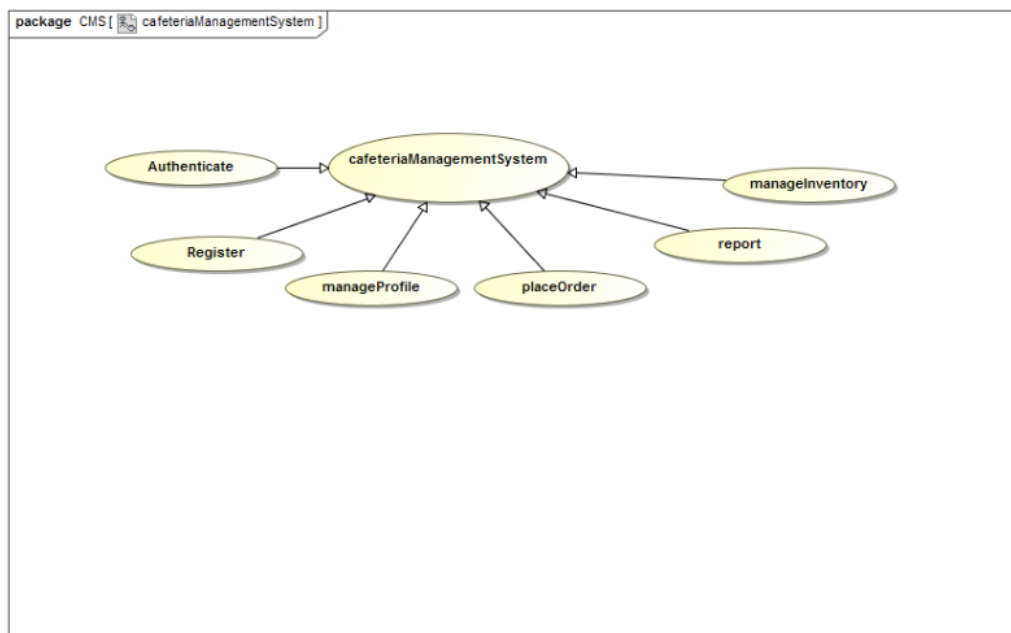## 5.1 High level use case diagram of the CMS



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 Register Module

The register module 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 of the register module.
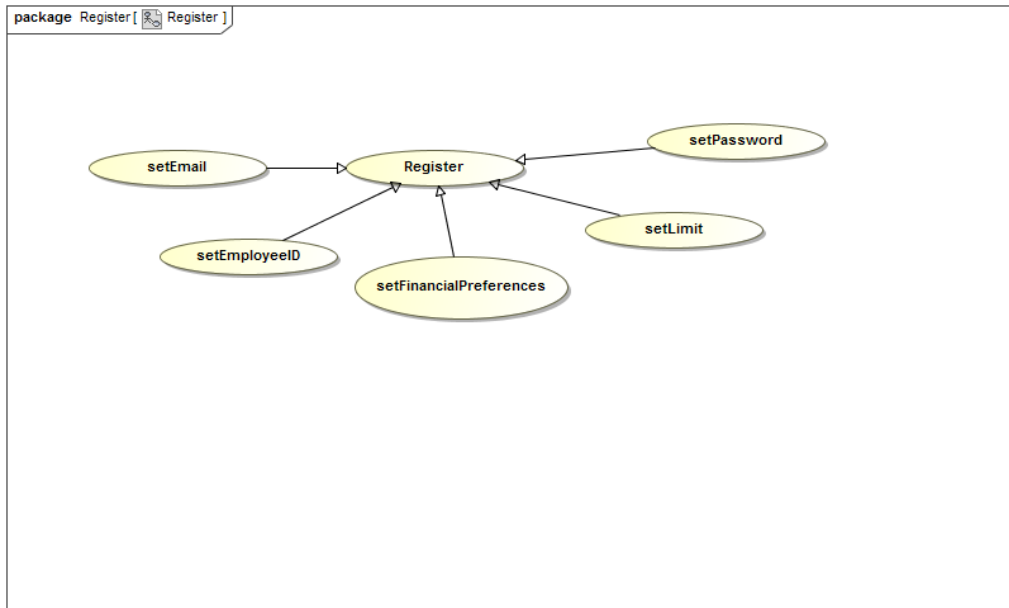
Figure 2: The use case for registering on the system

### 5.2.1 Set email

The service contract and activity diagram for setEmail follow. setEmail falls under the use case for Register (refer to page 8 - figure 2 to view this use case diagram). These details, entered by the user will be stored on the system.
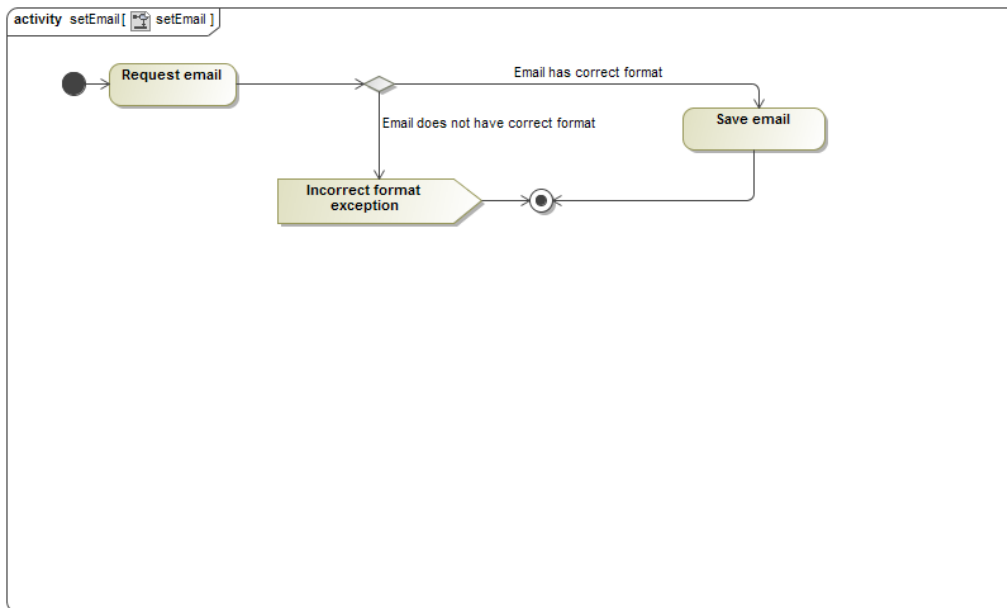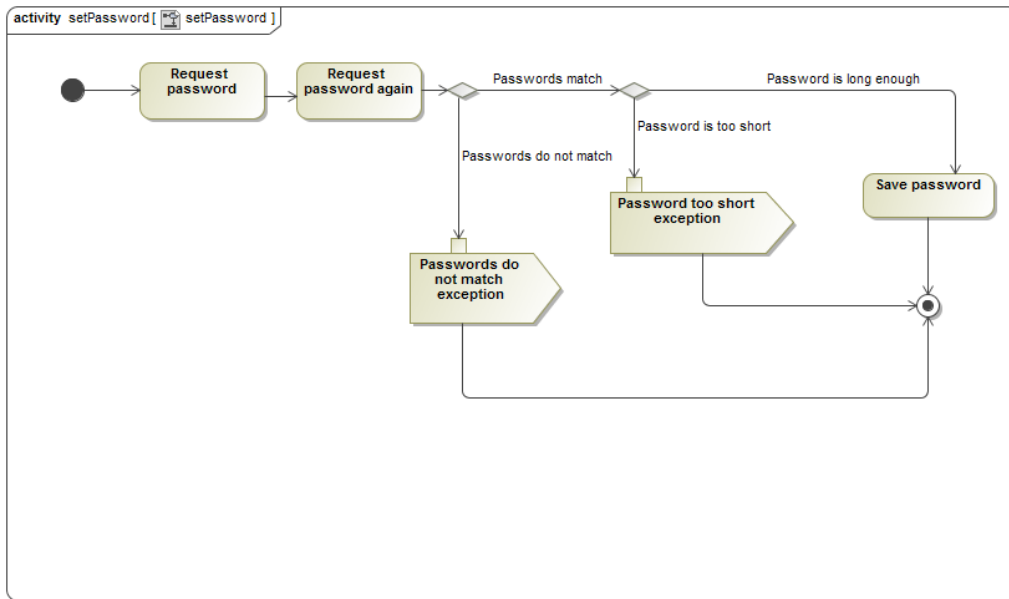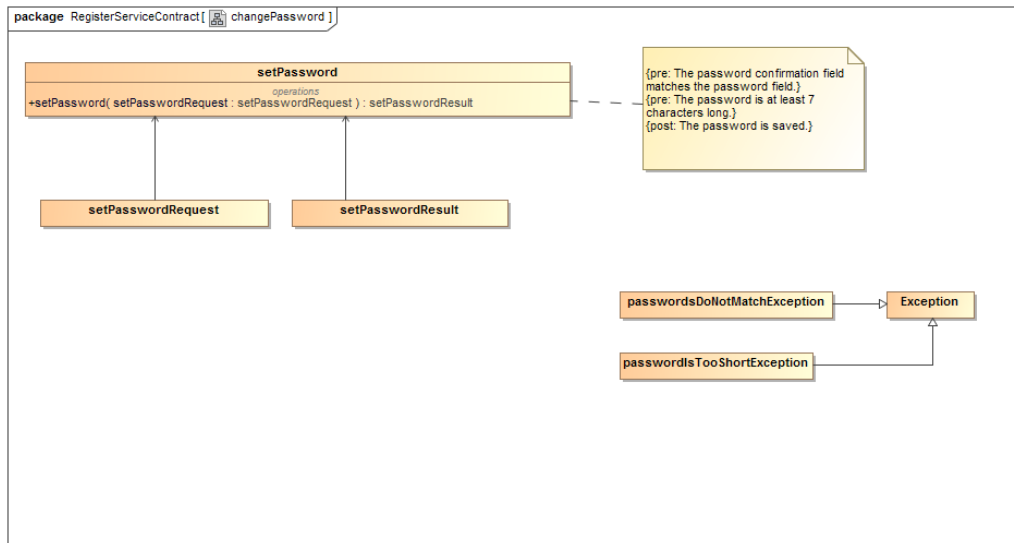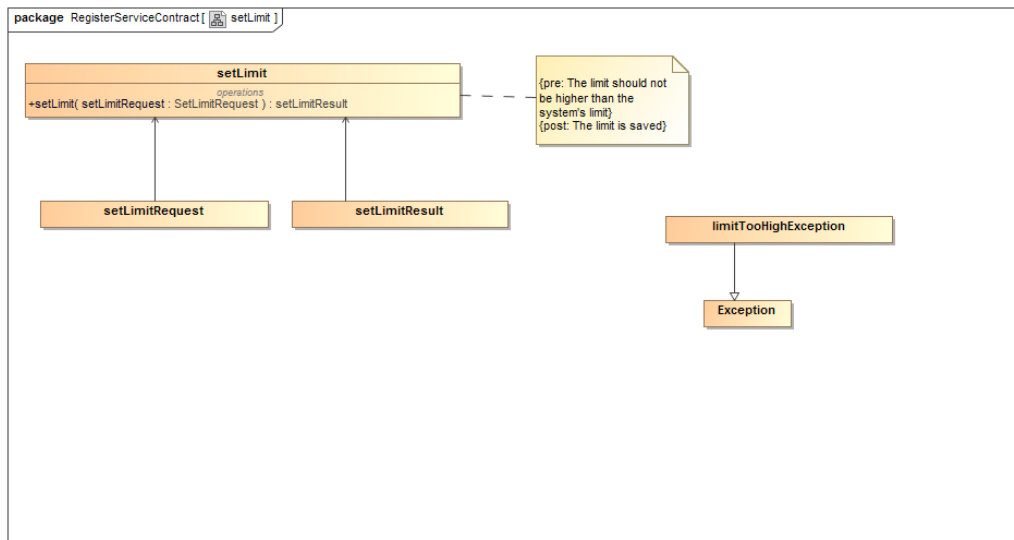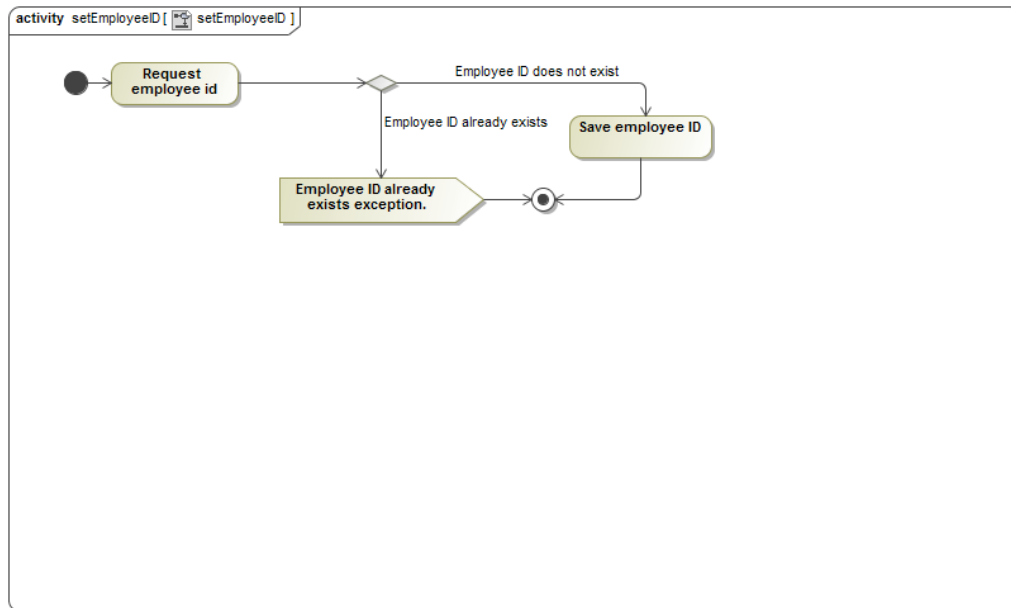
Figure 3: The activity diagram for setting an email address on the system
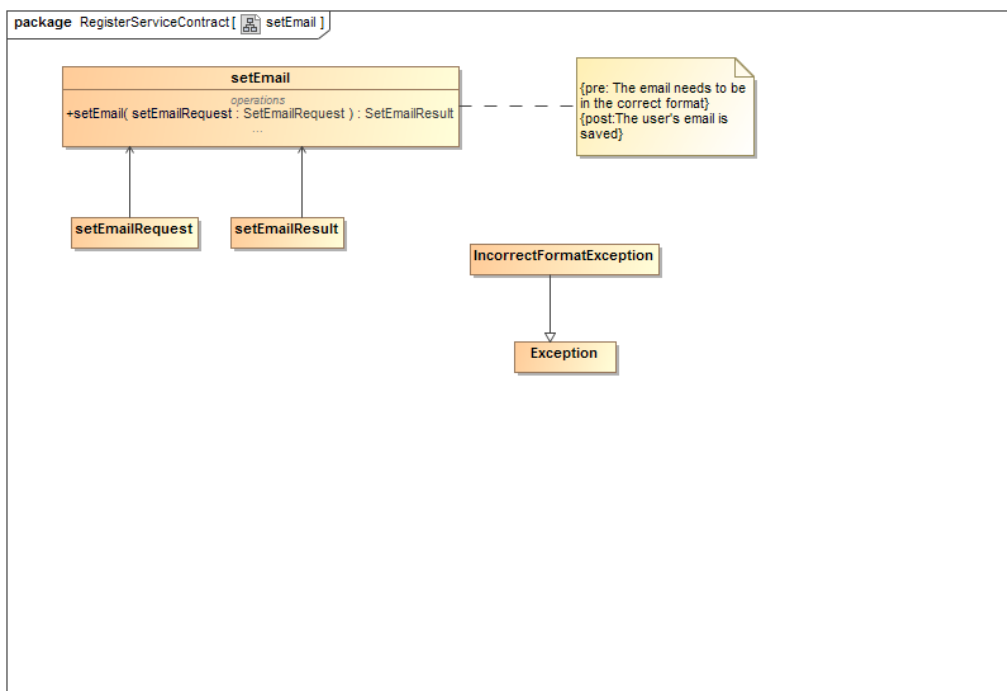


Figure 4: The service contract for setting an email address on the system

9

### 5.2.2  Set Password

The service contract and activity diagram for setPassword follow. setPassword falls under the use case for Register (refer to page 8 - figure 2 to view this use case diagram). These details, entered by the user will be stored on the system.



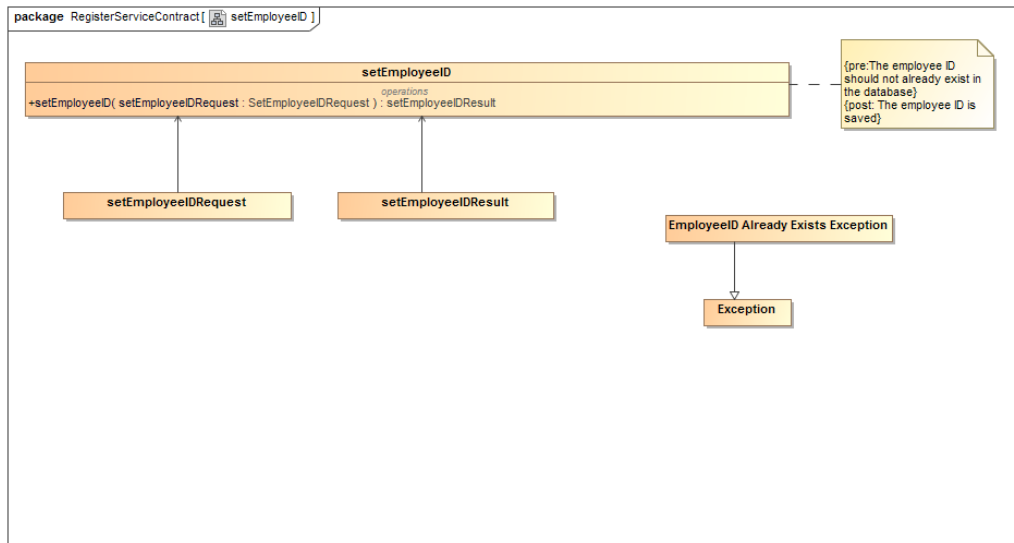Figure 5: The activity diagram for setting a password on the system

Figure 6: The service contract for setting a password on the system

### 5.2.3 Set Limit

The service contract and activity diagram for setLimit follow. setLimit falls under the use case for Register (refer to page 8 - figure 2 to view this use case diagram). These details, entered by the user will be stored on the system.
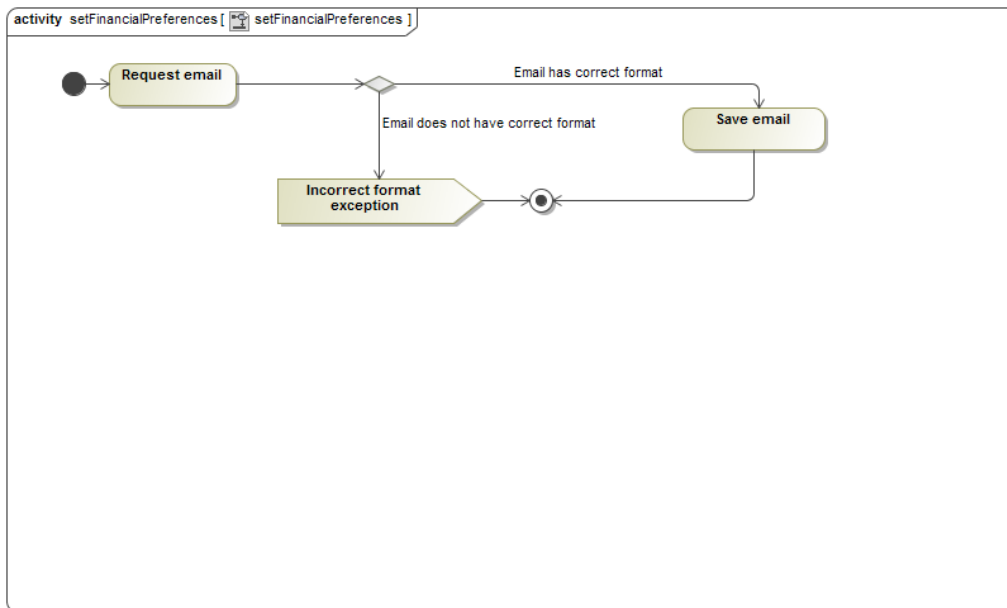


Figure 7: The service contract for setting a limit on the system

### 5.2.4 Set Employee ID

The service contract and activity diagram for setEmployeeID follow. setEmployeeID falls under the use case for Register (refer to page 8 - figure 2 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 employeeId on the system

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

### 5.2.5 Set Financial Preferences

The service contract and activity diagram for setFinancialPreferences follow. setFinancialPreferences falls under the use case for Register (refer to page 8 - figure 2 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 financial preferences on the system
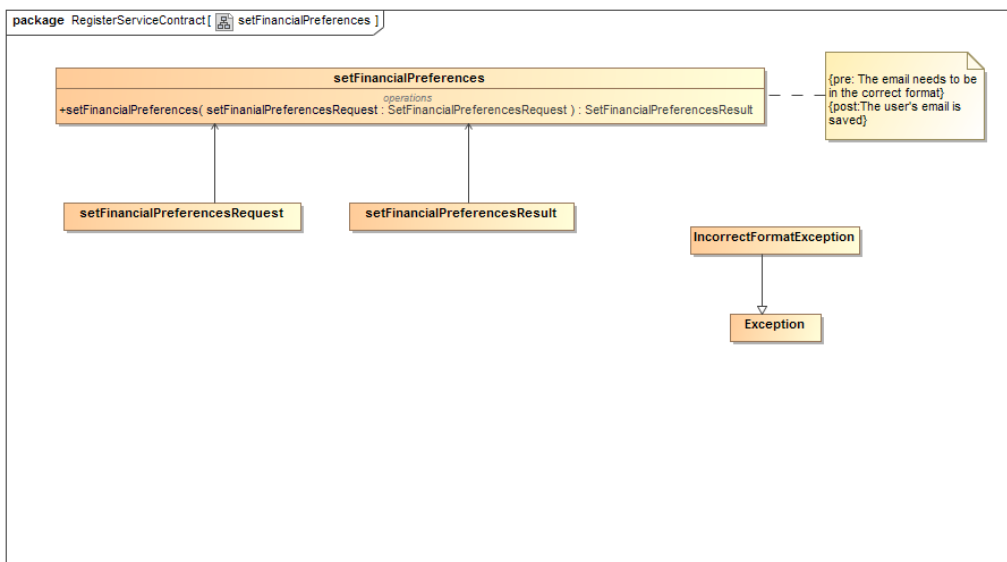


Figure 11: The service contract for setting financial preferences on the system

## 5.3 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.
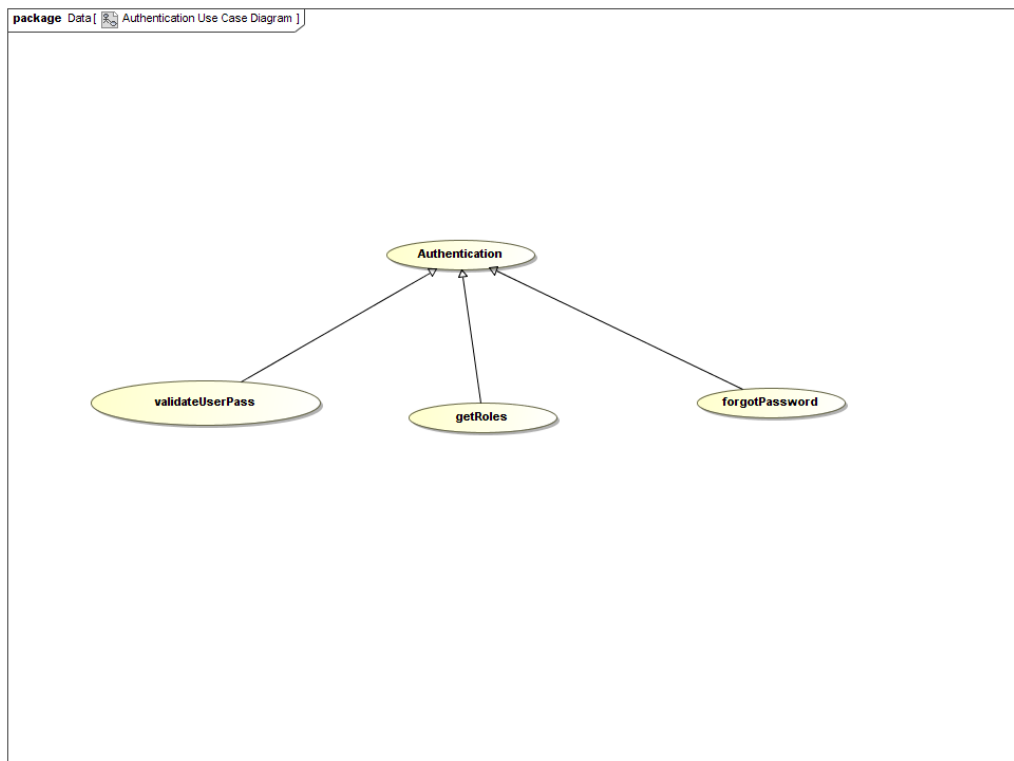


Figure 12: The main use case diagram for Authentication

### 5.3.1 Forgot Password

The service contract and activity diagram for forgotPassword follow. forgotPassword falls under the use case for Authentication (refer to page 15 - figure 12 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.
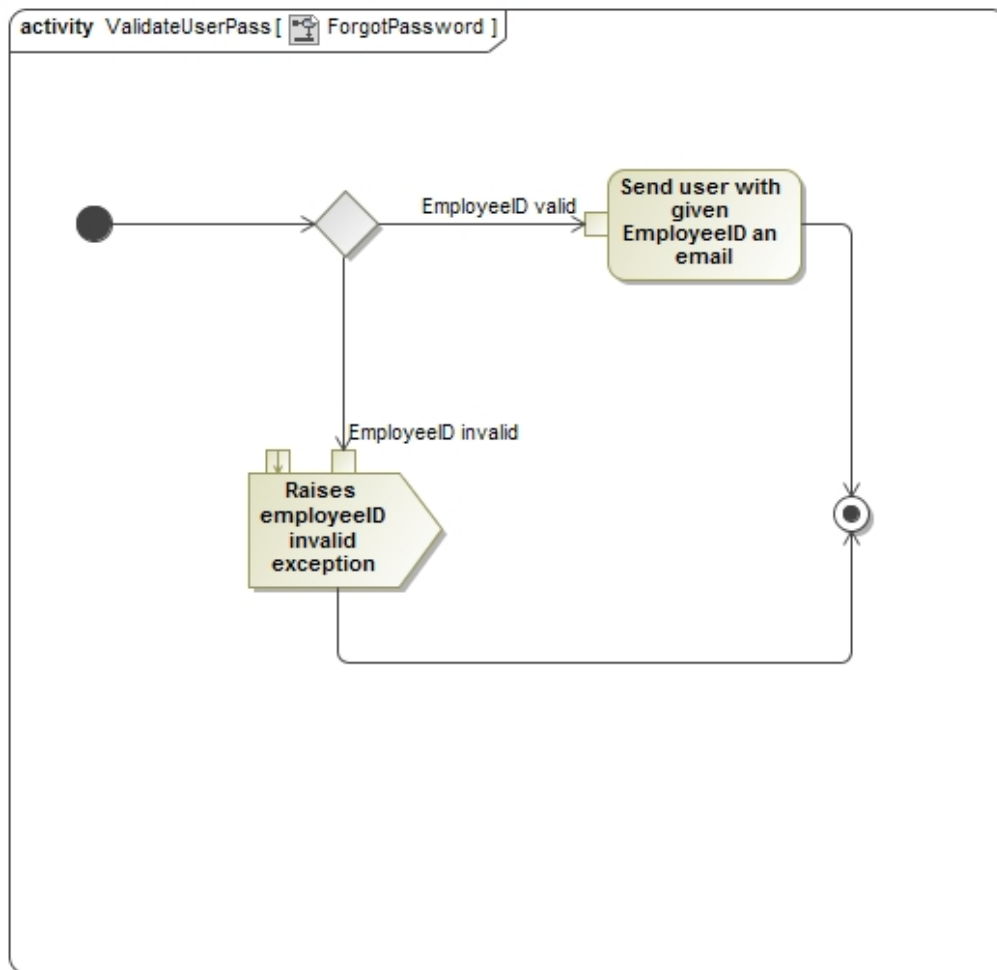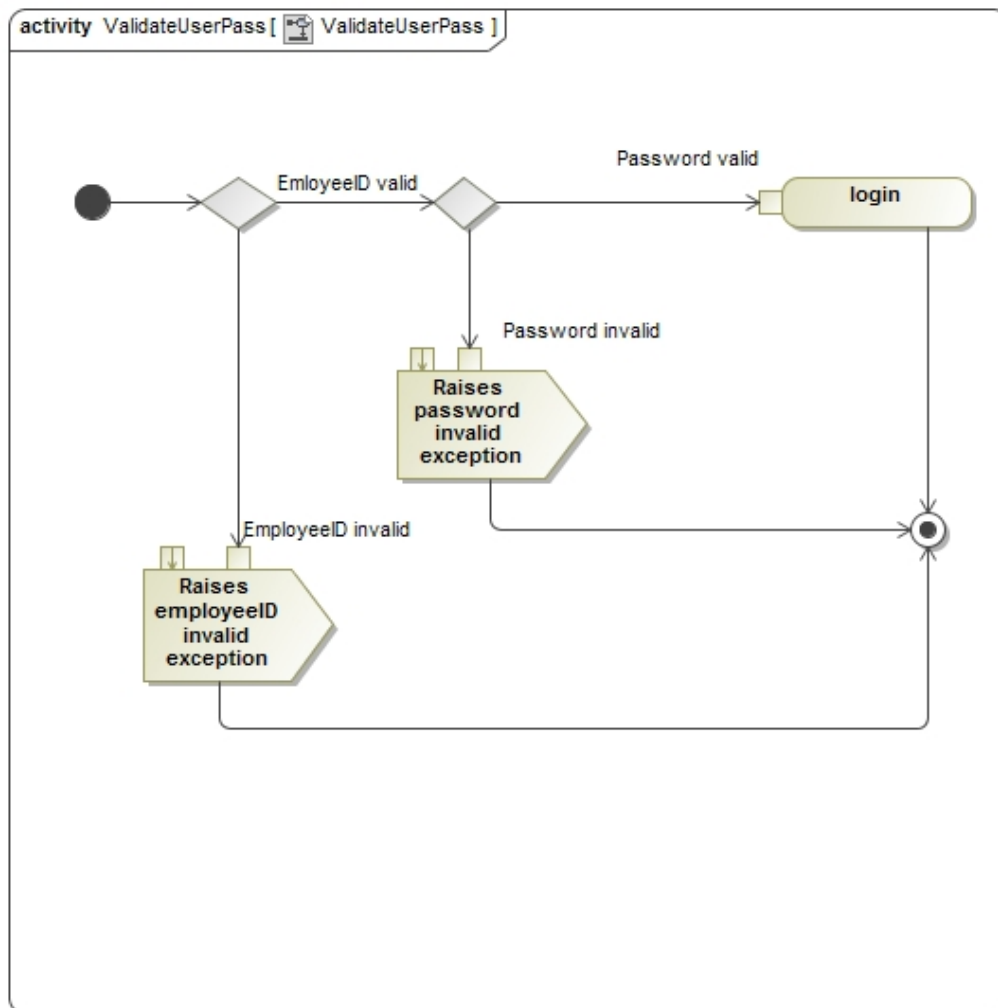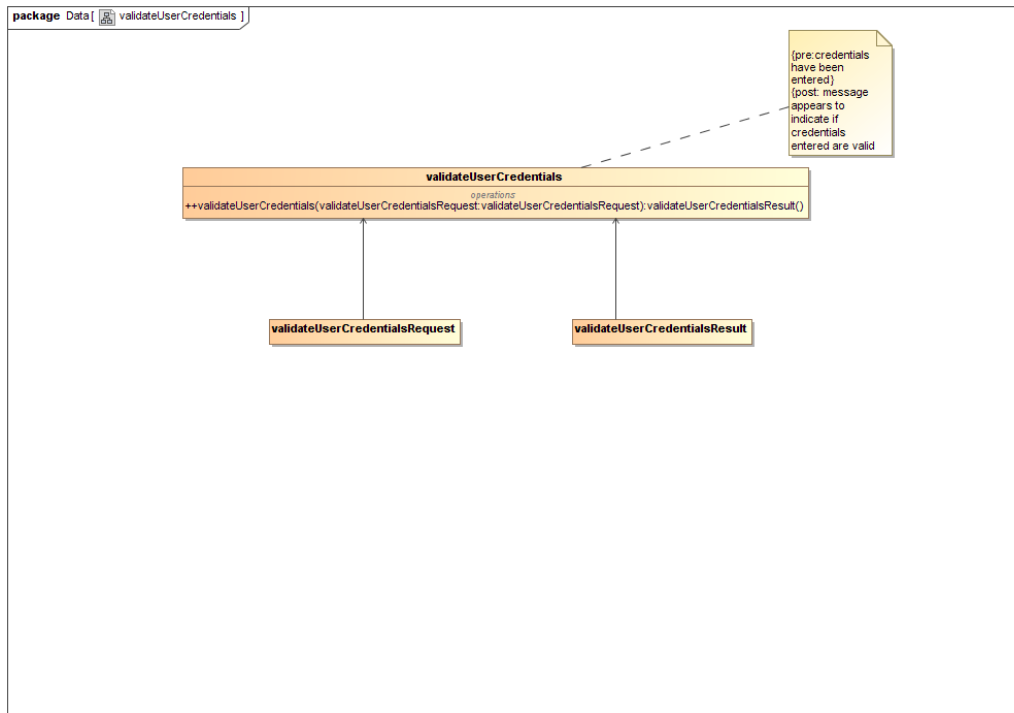
Figure 13: The activity diagram for forgotPassword
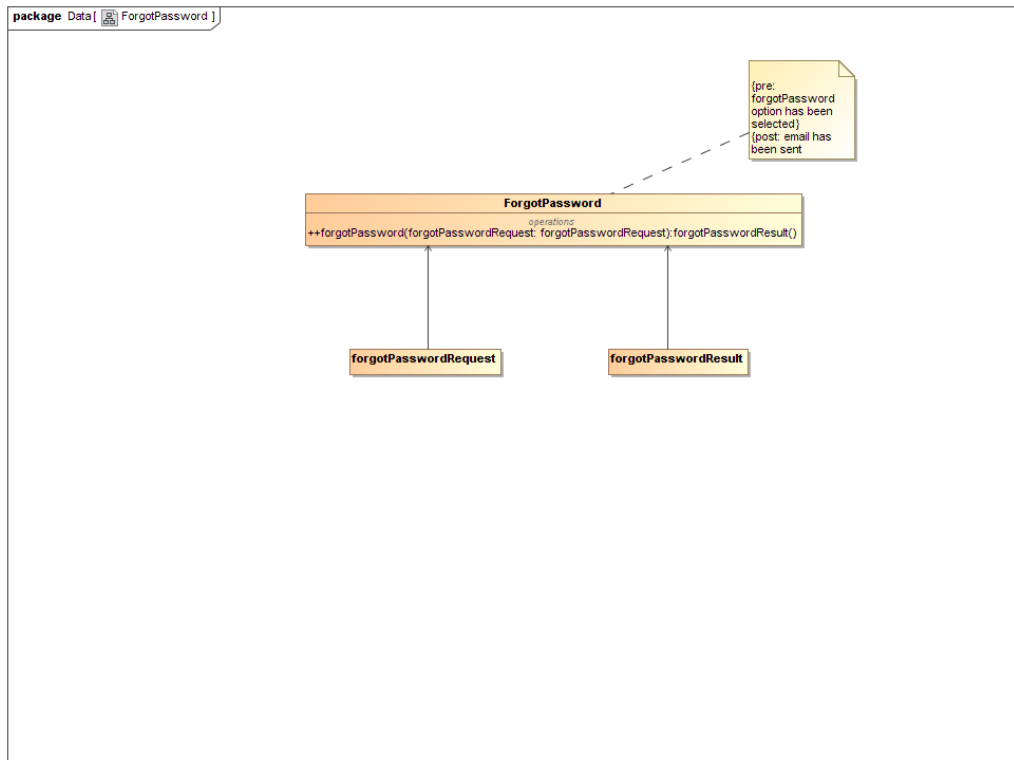
Figure 14: The service contract for forgotPassword

### 5.3.2 Validate User credentials

The service contract and activity diagram for validateUserCredentials follow. validateUserCredentials falls under the use case for Authentication (refer to page 15 - figure 12 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 15: The activity diagram for validateUserCredentials

Figure 16: The service contract for validateUserCredentials

## 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.
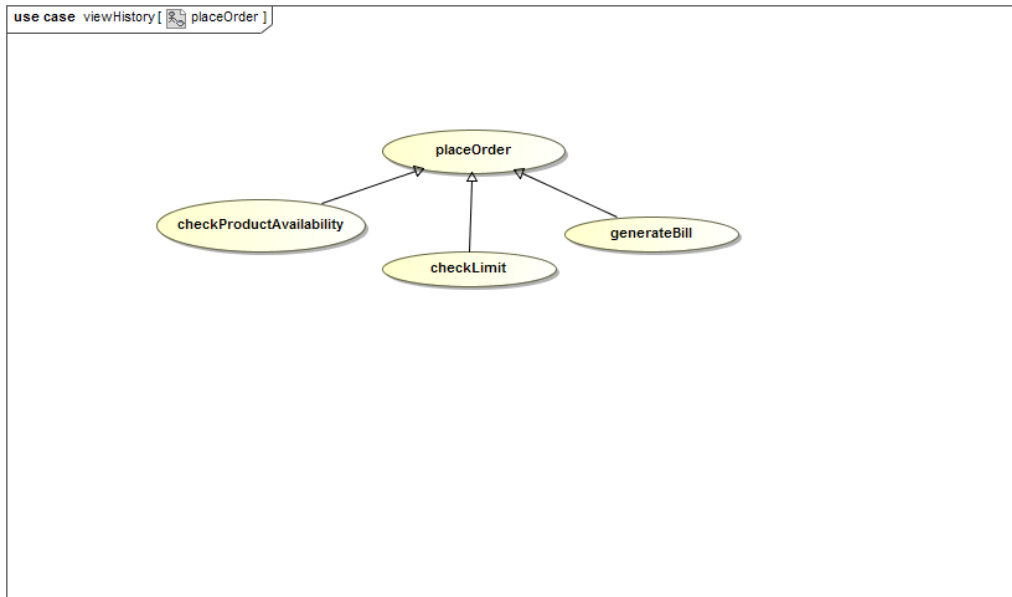
Figure 17: The use case for placing an order

### 5.4.1 CheckProductAvailability

The service contract and activity diagram for CheckProductAvailability follow. CheckProductAvailability falls under the use case for Place Orders (refer to page 20 - figure 17 to view this use case diagram). The system will check whether the product that the user has selected to purchase is currently in stock.
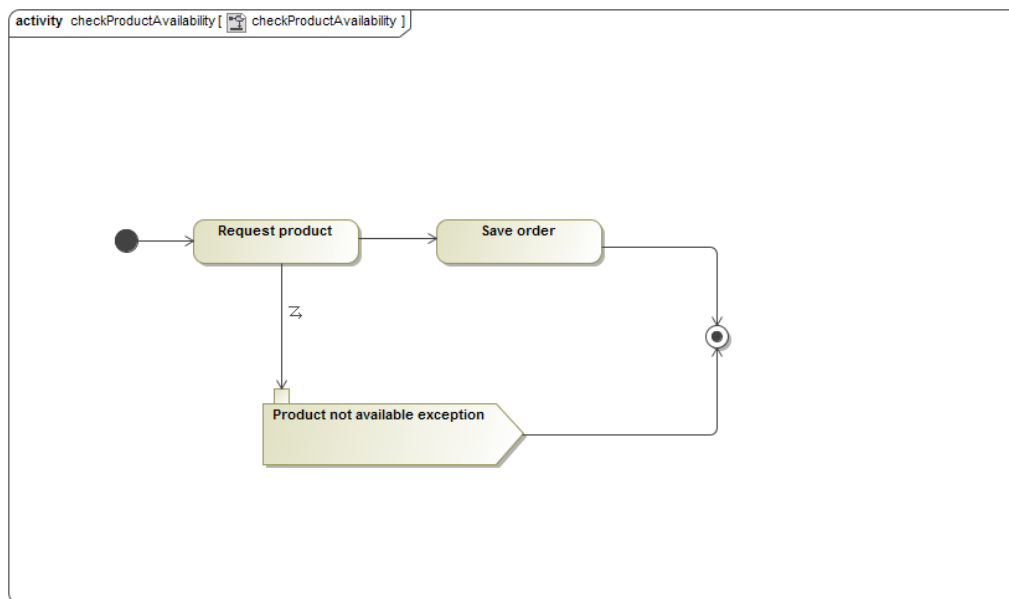
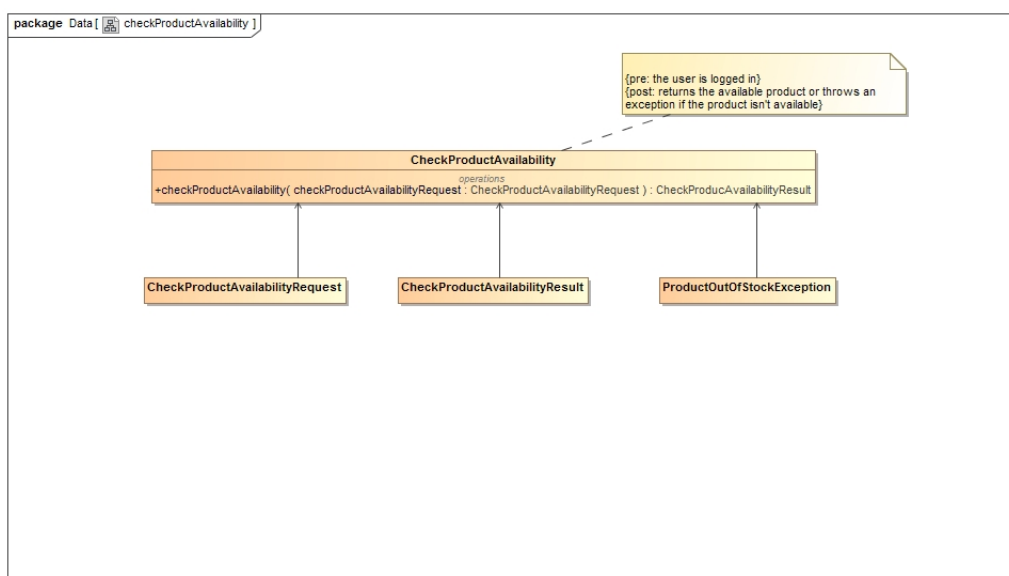Figure 18: The activity diagram for checking product availibilty



Figure 19: The service contract for checking product availabilty

### 5.4.2 checkLimit

The service contract and activity diagram for checkLimit follow. checkLimit falls under the use case for Place Orders (refer to figure 17 page 20 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.
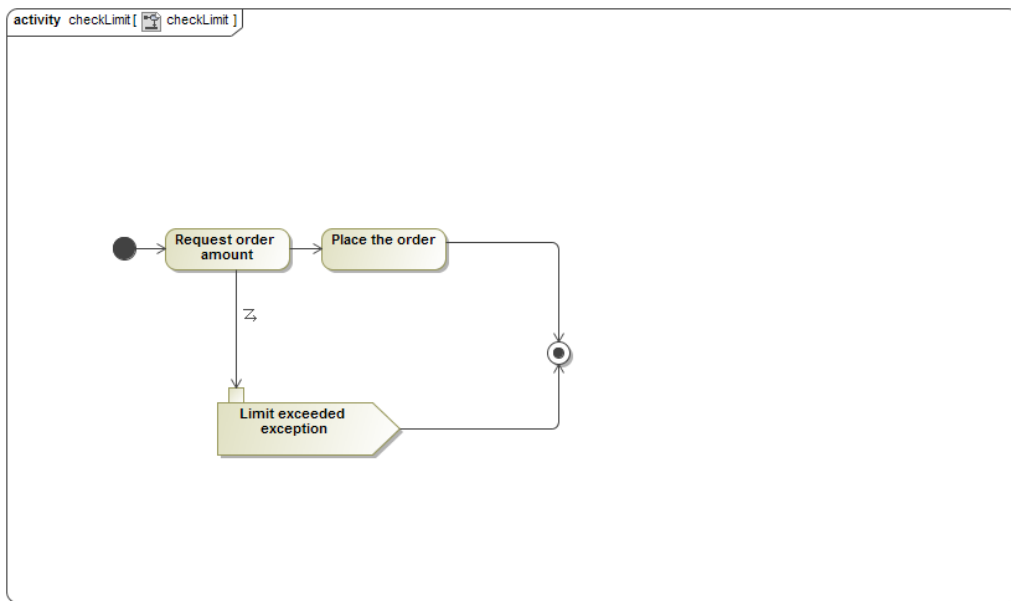


Figure 20: The activity diagram for checking limits
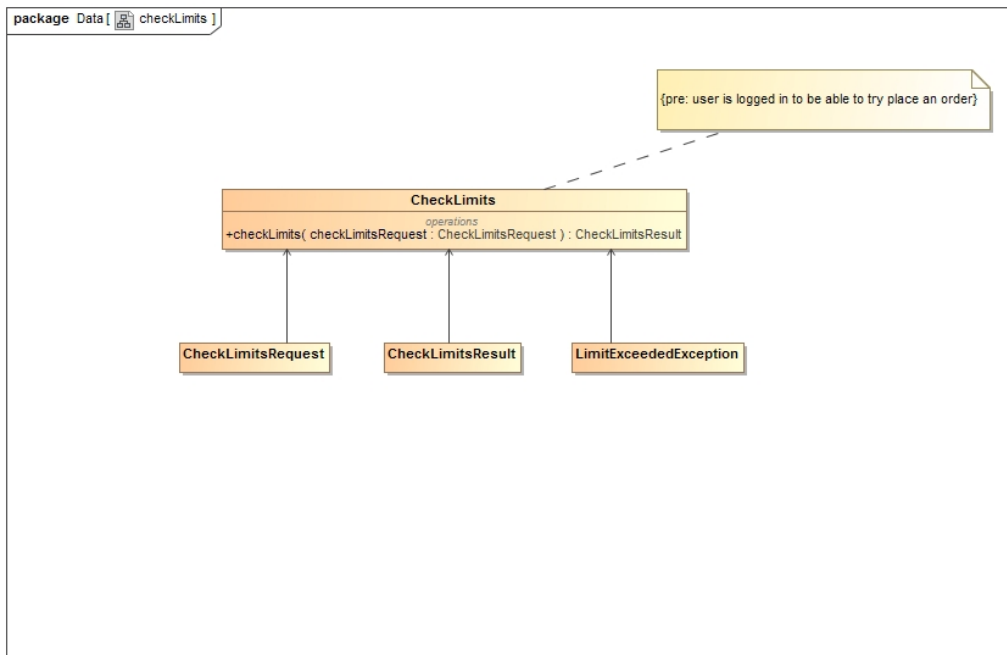
Figure 21: The service contract for checking limits

## 5.5   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 mentioned functionality around managing the profile. Hence, it has the use cases generateFavourite and edit profile.
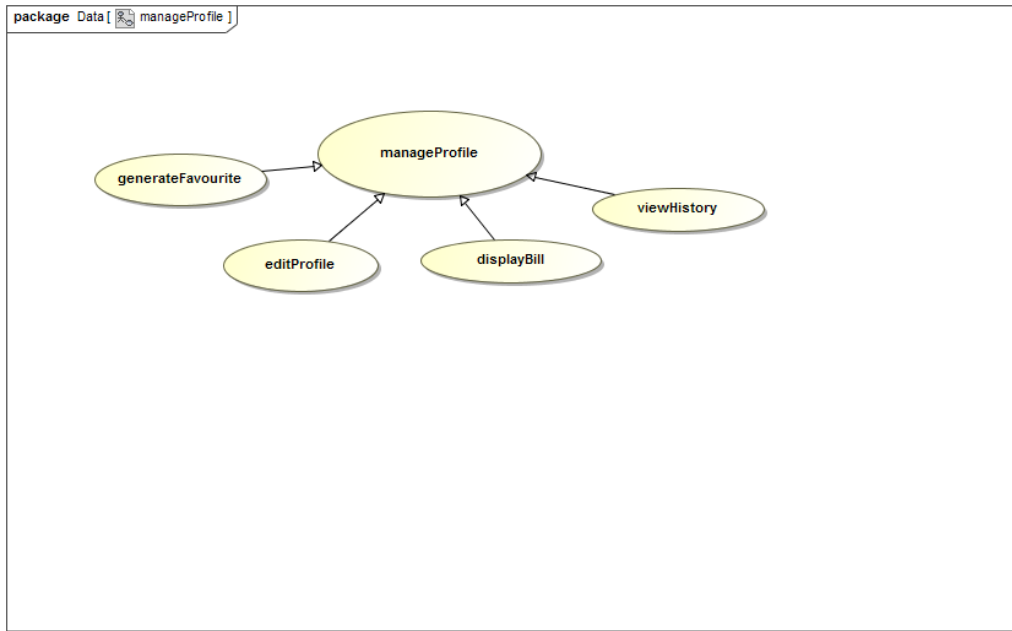
Figure 22: The use case diagram for managing profile

### 5.5.1 setFinancialPreferences

The service contract and activity diagram for setFinancialPreferences to follow. setFinancialPreferences falls under the use case for Manage Profile (refer to page 24 figure 22 to view this use case diagram). The user will be able to choose where their monthly bill is sent to.
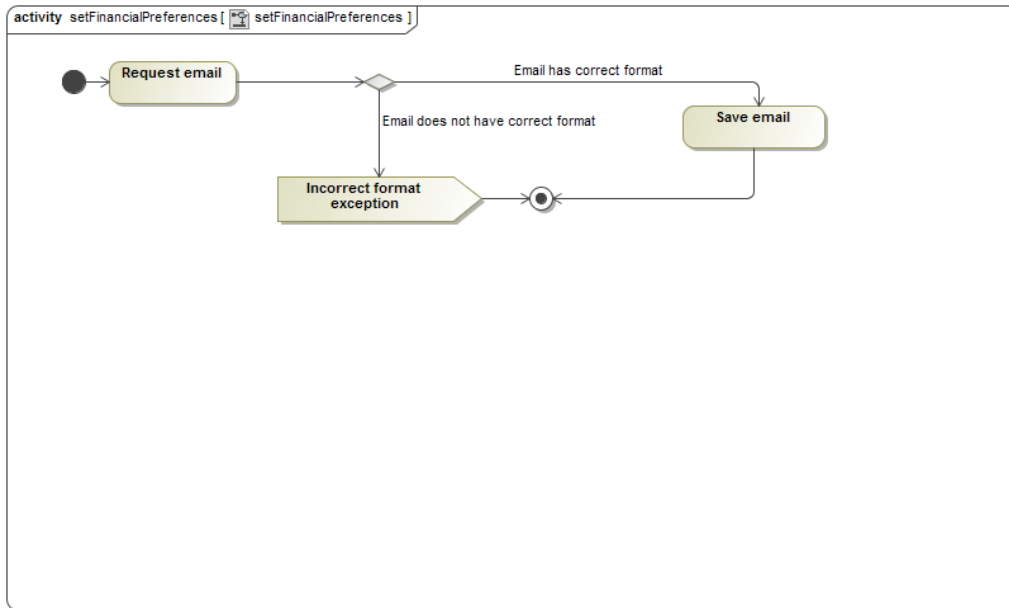
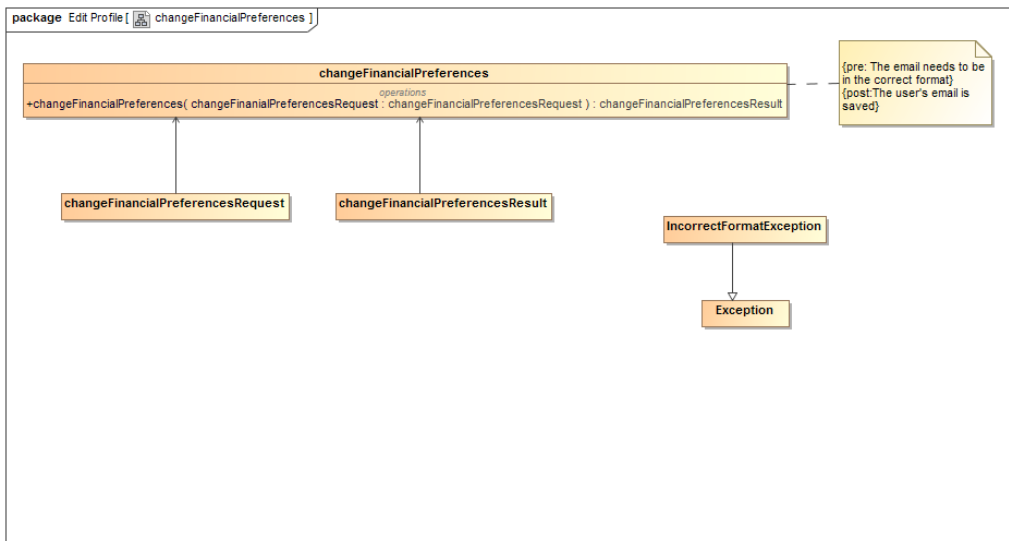Figure 23: The activity diagram for setting financial preferences



Figure 24: The service contract for settting financial preferences

### 5.5.2 changePassword

The service contract and activity diagram for changePassword follow. changePassword falls under the use case for Edit Profile (refer to page 24 figure 22 to

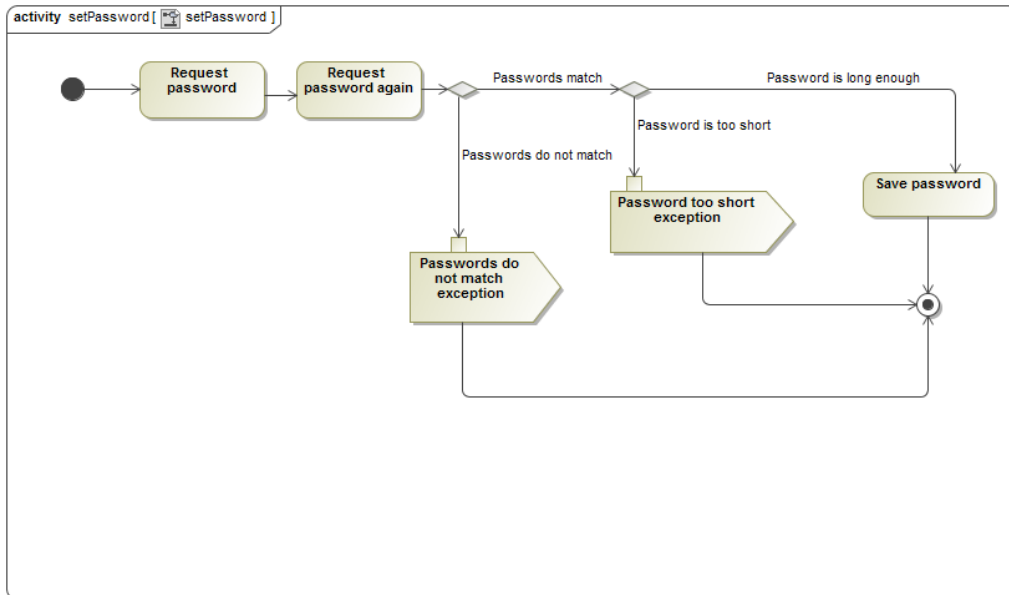view this use case diagram). The user will be able to edit their password.



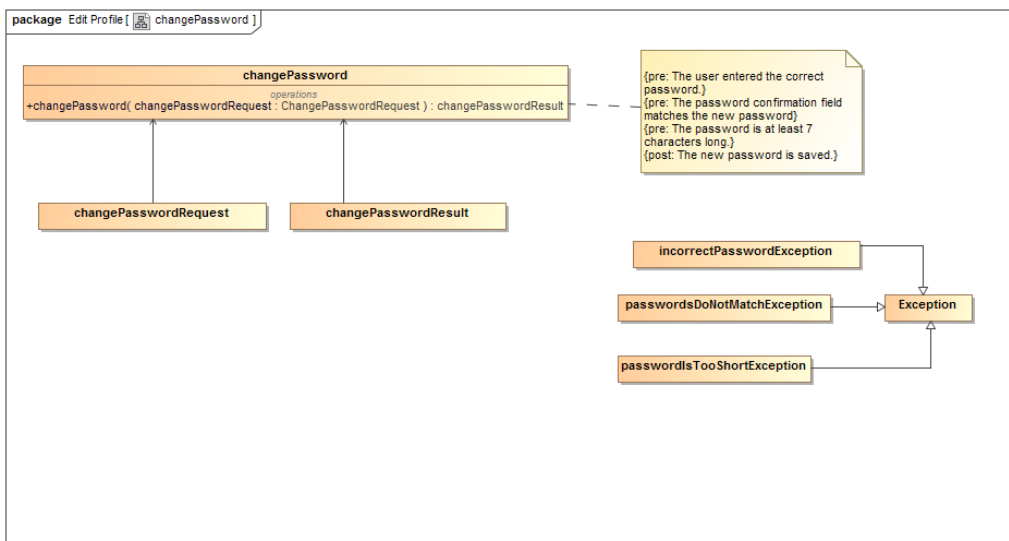Figure 25: The activity diagram for changing password



Figure 26: The service contract for changing password

26

### 5.5.3   changeEmail

The service contract and activity diagram for changeEmail follow. changeEmail falls under the use case for Edit Profile (refer to page 24 figure 22 to view this use case diagram). The user will be able to edit their email address.
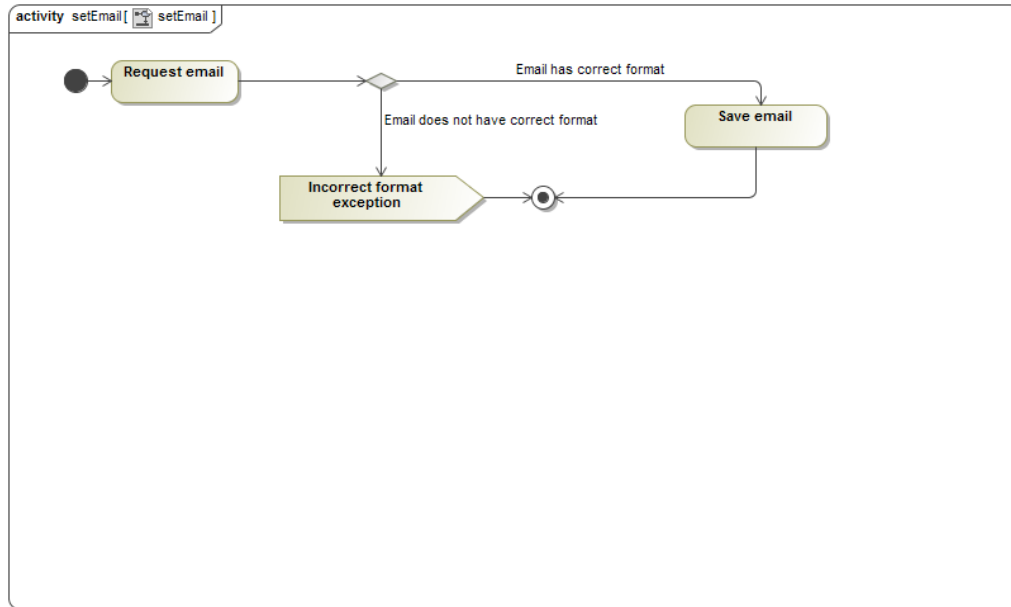


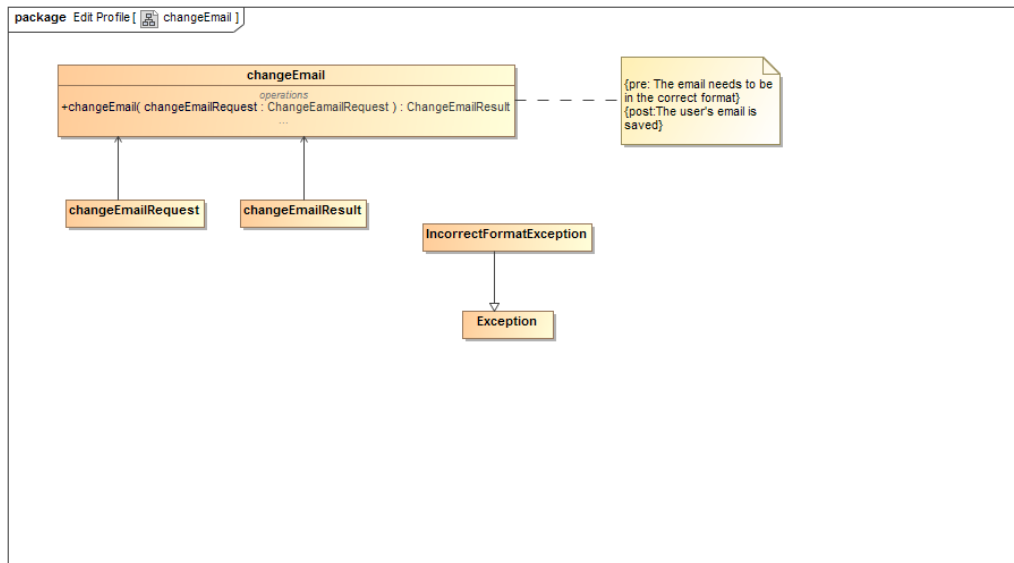Figure 27: The activity diagram for changing email address

Figure 28: The service contract for changing email address

### 5.5.4  changeLimit

The service contract and activity diagram for changeLimit follow. change-Limit falls under the use case for Edit Profile (refer to page 24 figure 22 to view this use case diagram). The user will be able to change their personal spending limit on their profile, however, this must not exeed the system's limit.
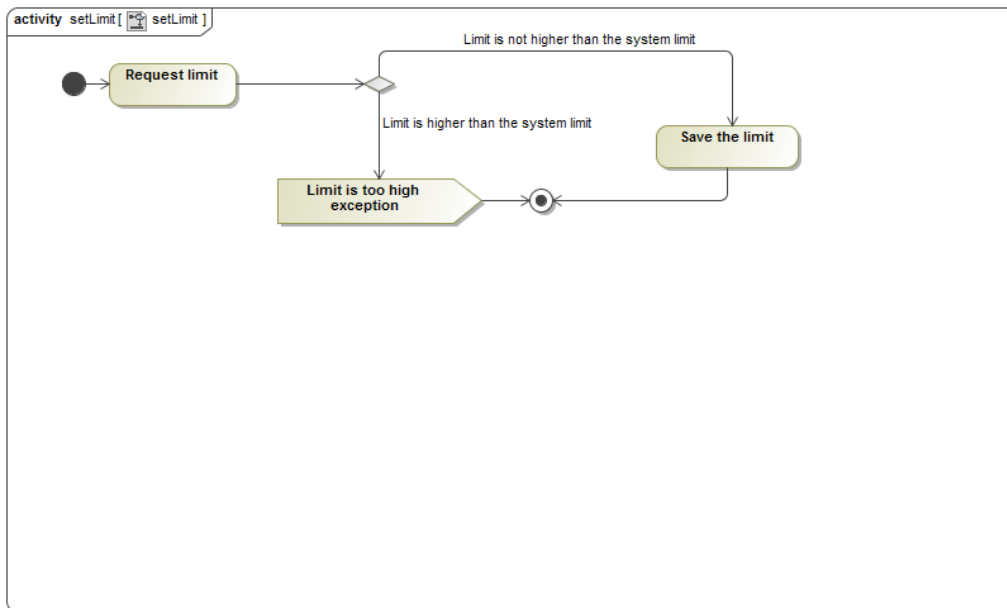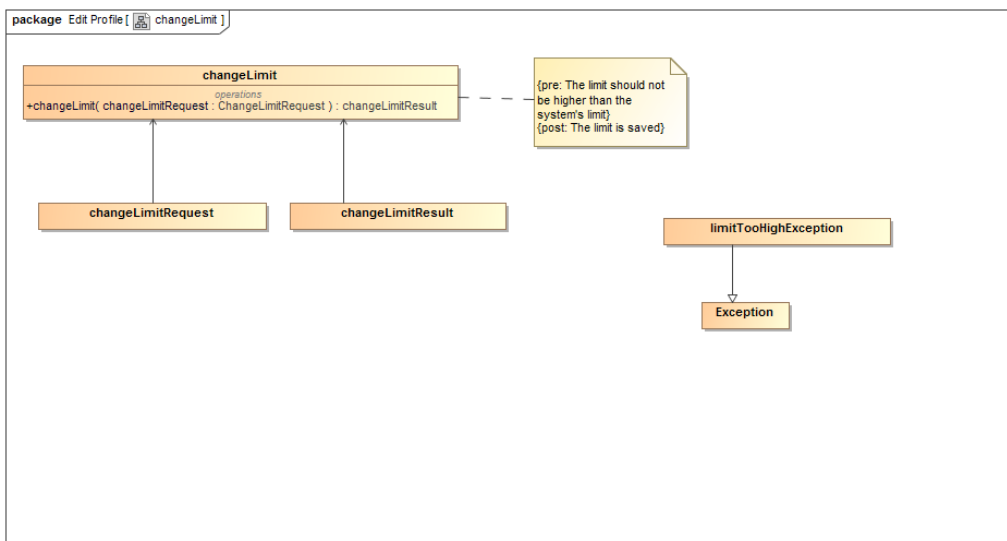
Figure 29: The activity diagram for setting limit



Figure 30: The service contract for setting limit

# 6 Comment

The document has now been updated with the use cases and other diagrams for the modules Authentication and Register.