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

May 29, 2015

Contents

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, hence the main use cases that this document will be focussing on are placing orders and managing a user profile. These will be explored in quite some detail. The agile method involves an interactive and incremental 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, executing orders from the cafeteria, generating bills and sending these to the appropriate parties and facilitating payments for access cards (or the use of unique access card numbers).

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 carry around cash if they want to purchase anything from the cafeteria. Hence, this is equivalent to purchasing from an external food outlet where they can also pay with their preferred method of payment. The employees have to hence use up fuel and time and lastly 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, with an amount being deducted from their salary at the end of the month.

Resolve proposed the Cafeteria Management System to assist with this problem. After our first meeting with the client, they brought to our atten-

tion that at times the cafeteria does not even have enough stock to provide some of the menu items, thus the managing of inventory or 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 such a problem. At the end of each month, the bill for the 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 for control purposes. The system will have its own maximum, such that users cannot set a limit that exeeds this.

4 Functional Requirements and Aplication 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:

- Login
- 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

- Login
  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 own preferences as well as keep track of monthly purchases.

- Manage Inventory
  This use case is considered important since this 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 and users must be able to send billing reports to Pay-Roll to have the amount spent per month 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 casher or cafeteria manager that they are low on certain inventory. The system is not dependant on this functionality.

4.3 Use Case/Service Contracts

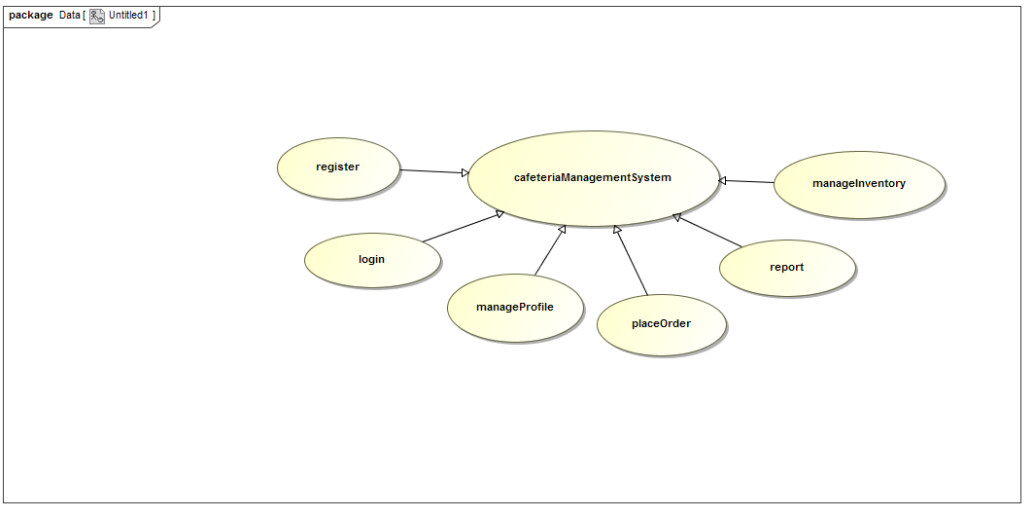

Figure 1: Cafeteria Management System Use Case

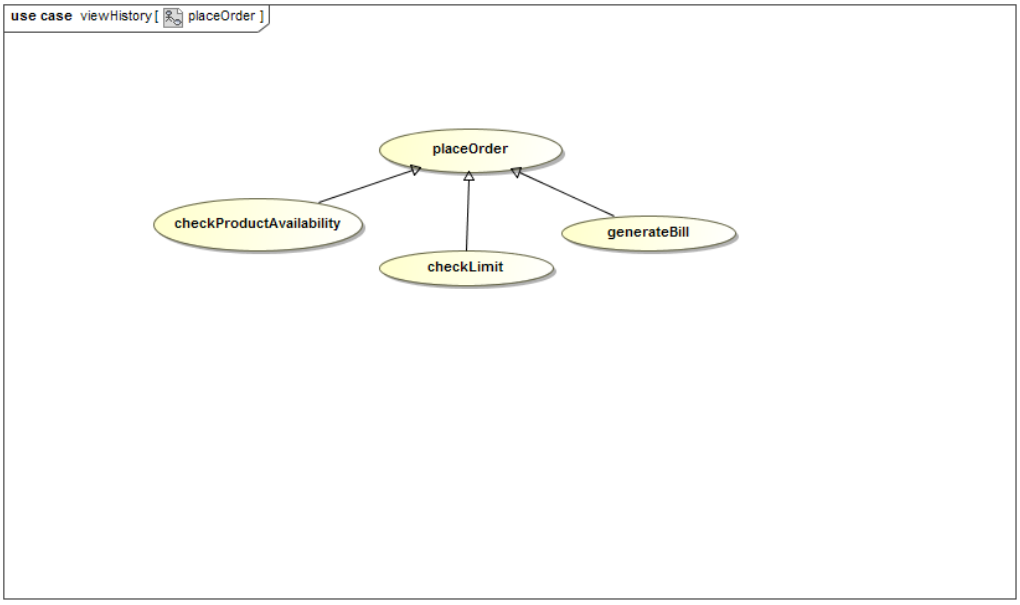

Figure 2: Place Order Use Case

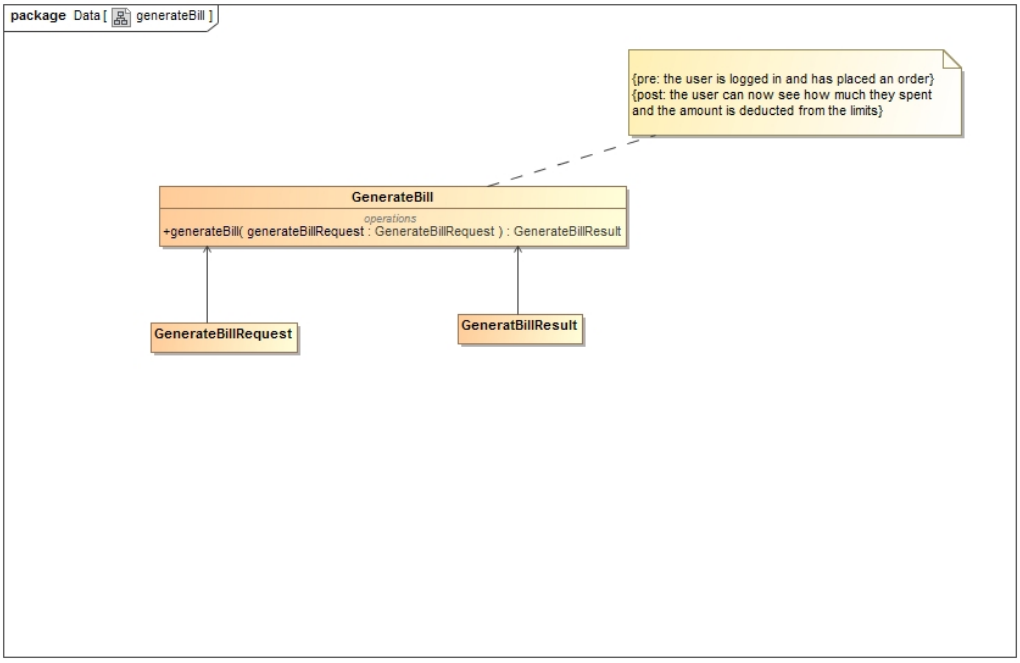

Figure 3: generateBill service contract

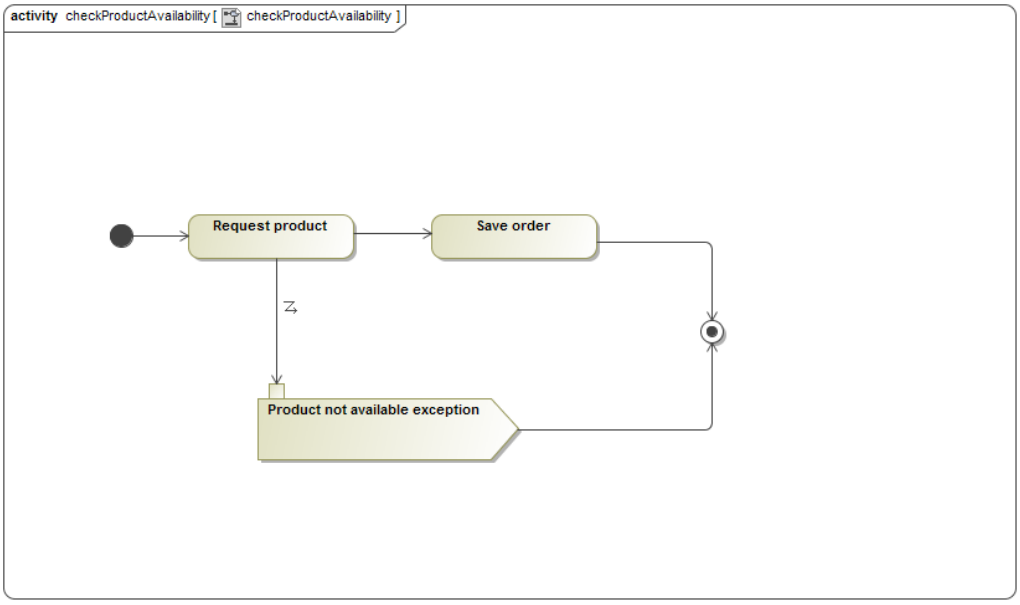

Figure 4: checkProductAvailability activity diagram

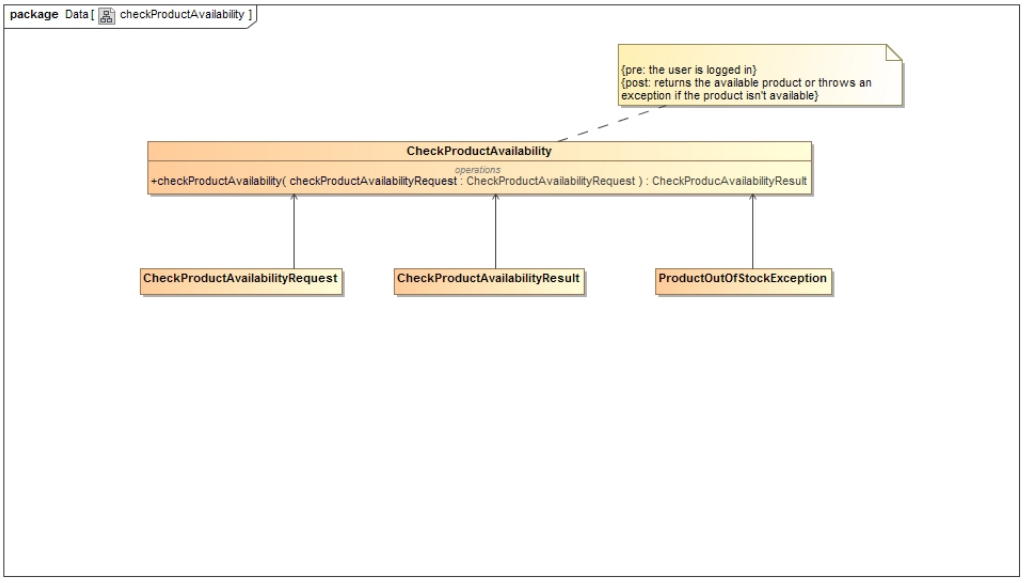

Figure 5: checkProductAvailability service contract

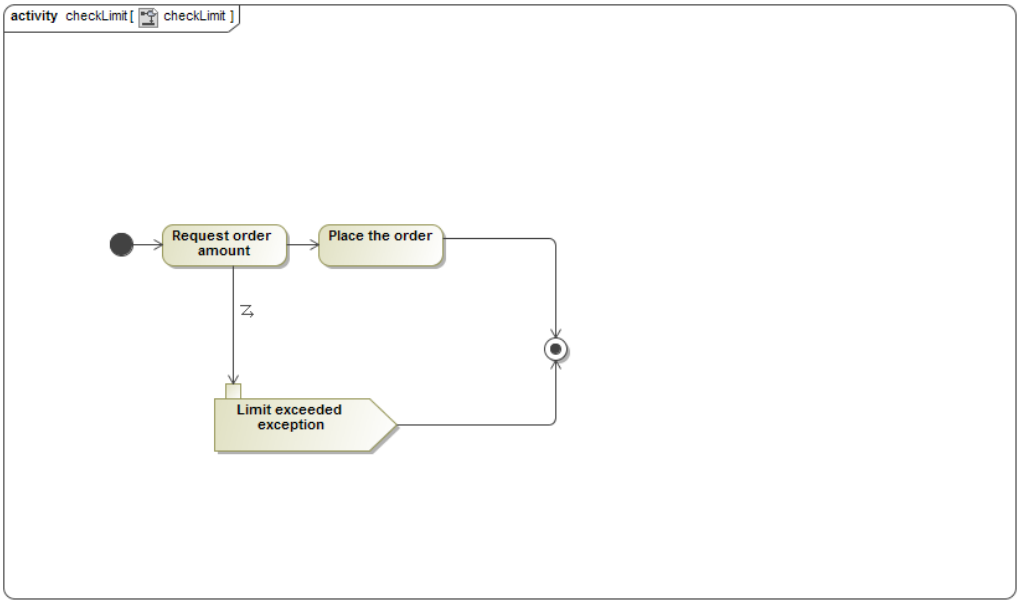

Figure 6: checkLimit activity diagram

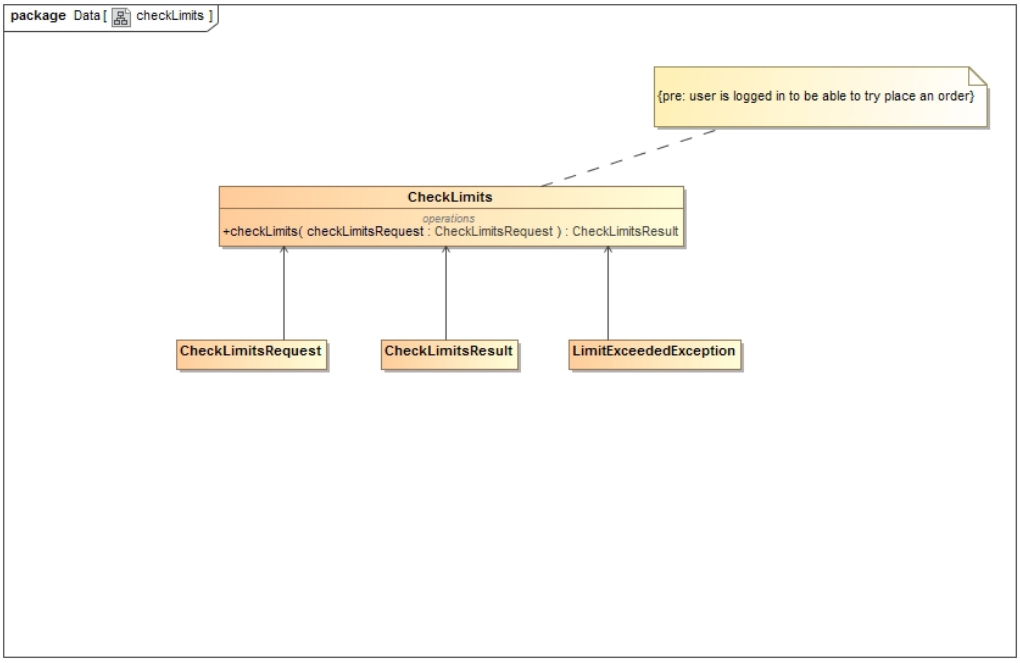

Figure 7: checkLimit service contract

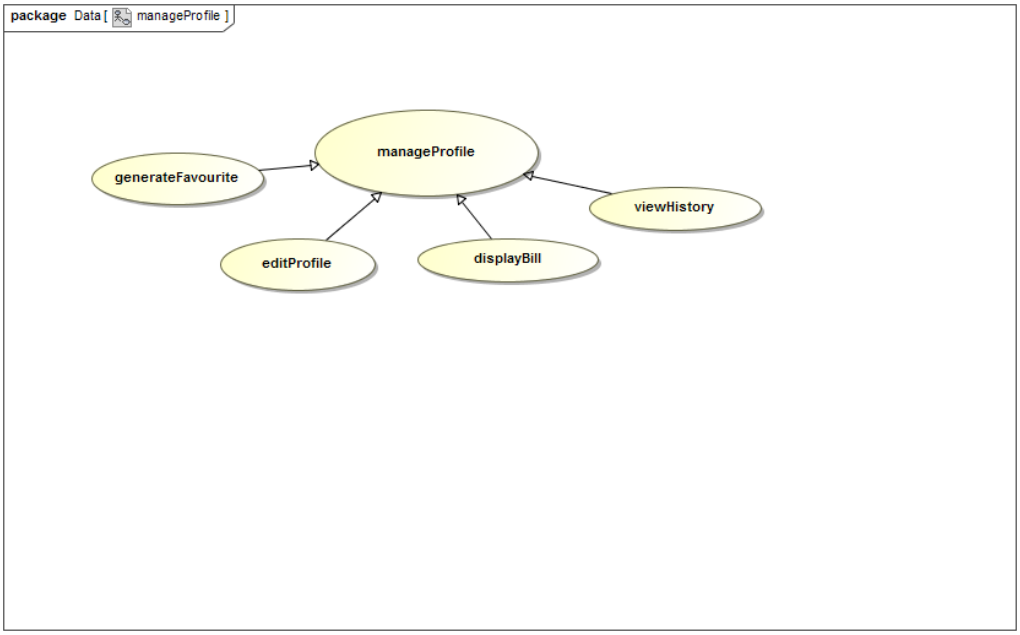

Figure 8: manage profile use case

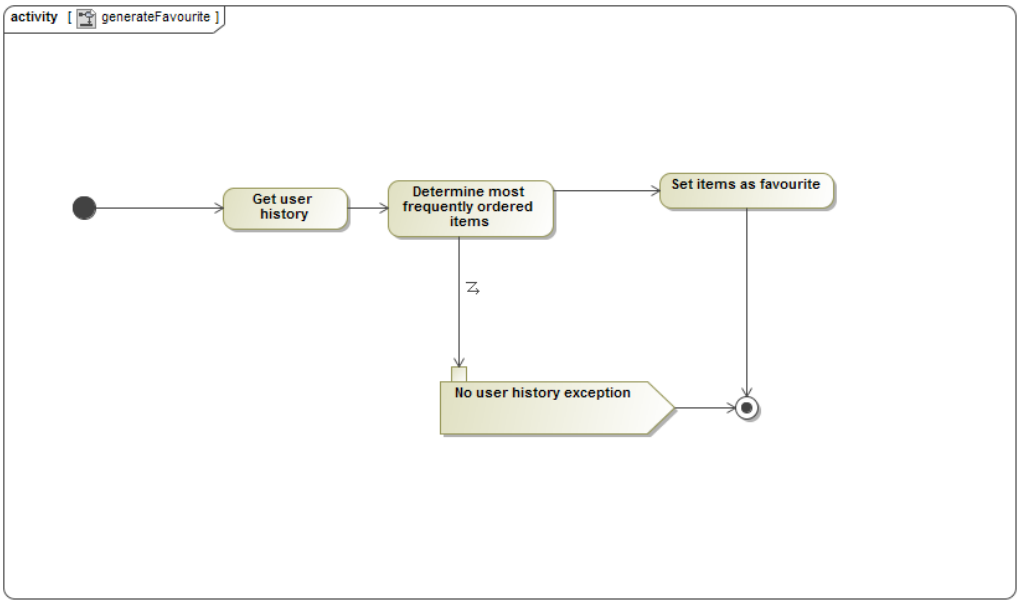

Figure 9: viewFavourite activity diagram

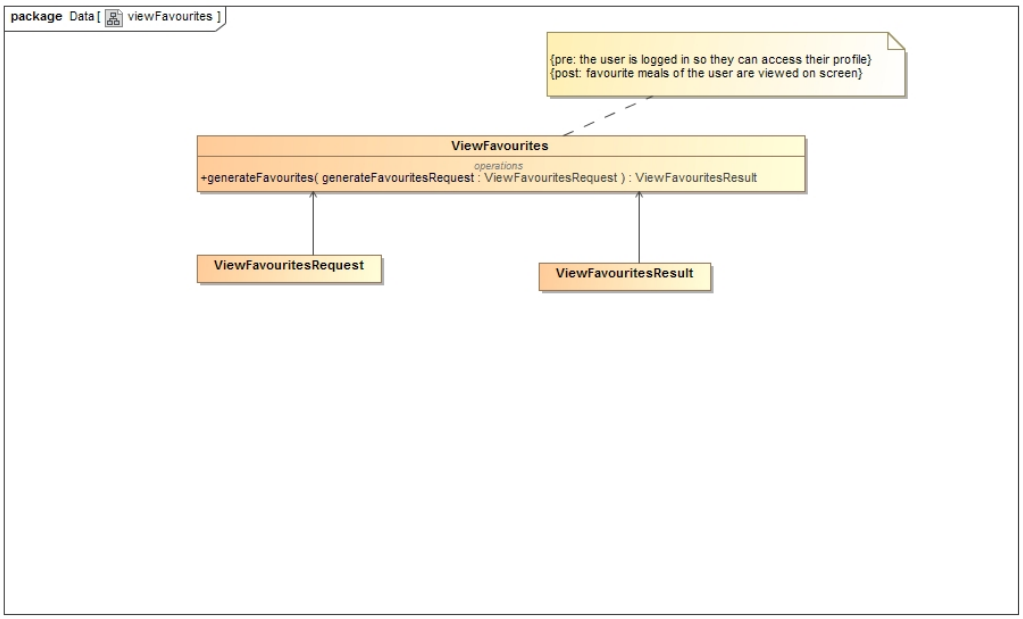

Figure 10: viewFavourite service contract

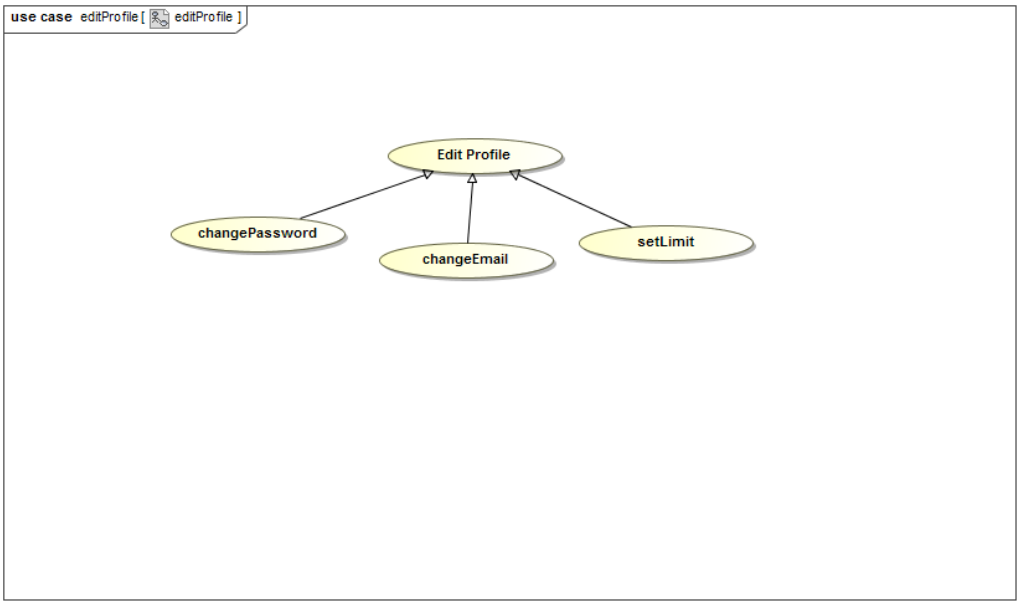

Figure 11: edit profile Use Case

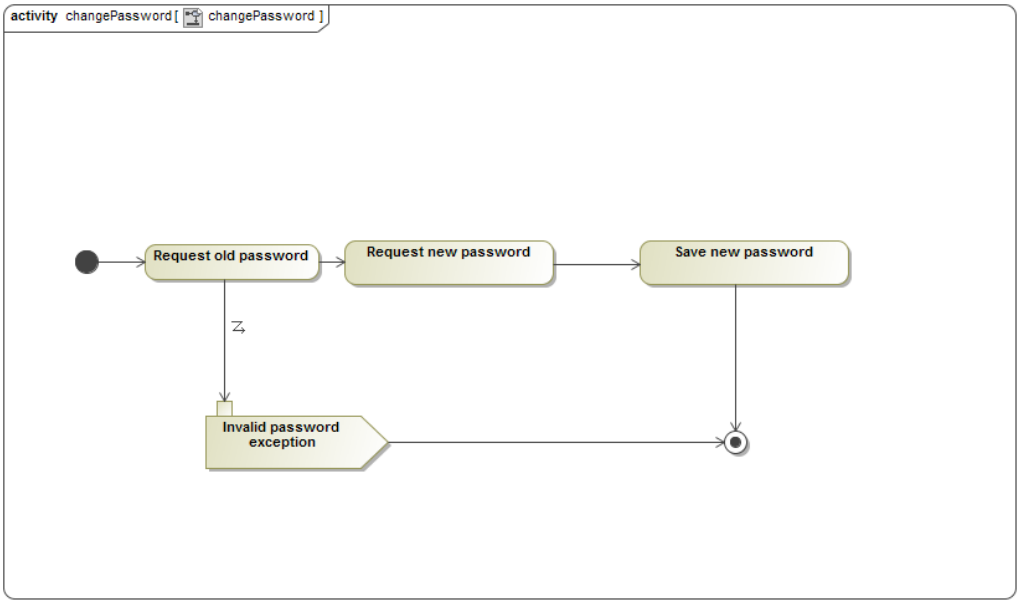

Figure 12: changePassword activity diagram

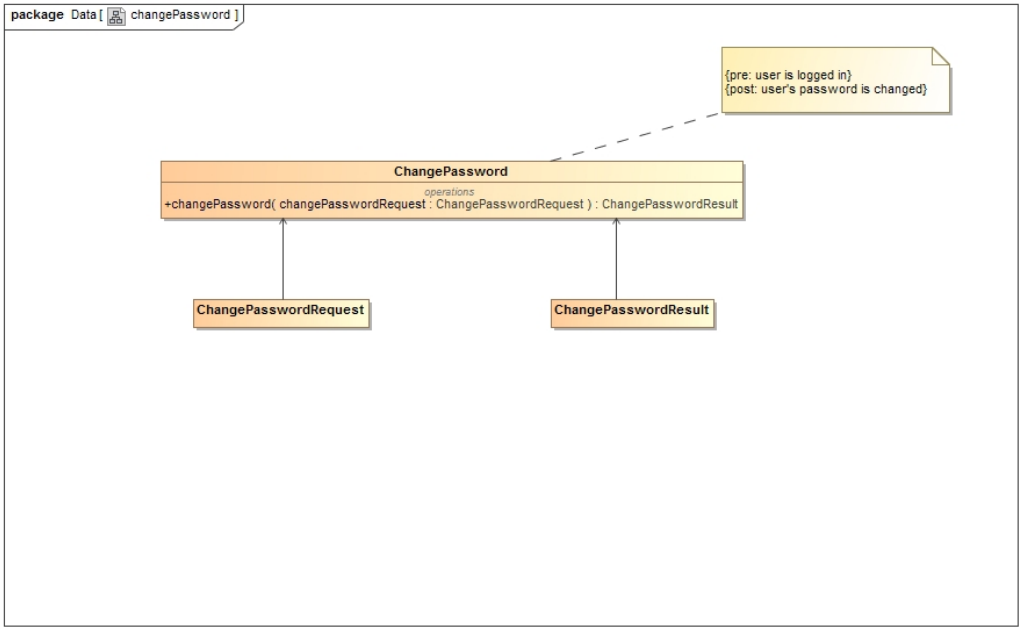

Figure 13: changePassword service contract

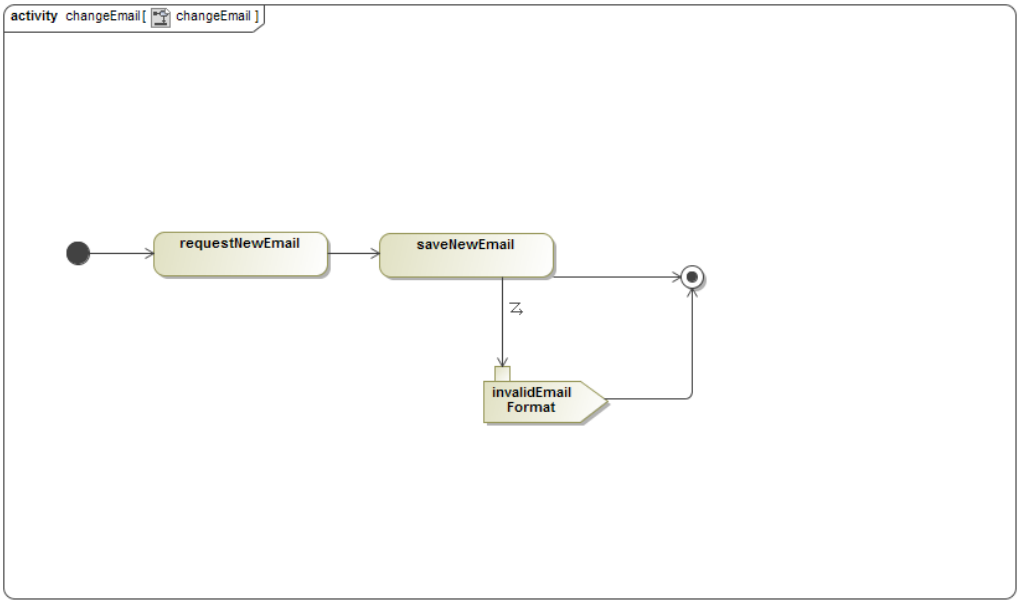

Figure 14: changeEmail activity diagram

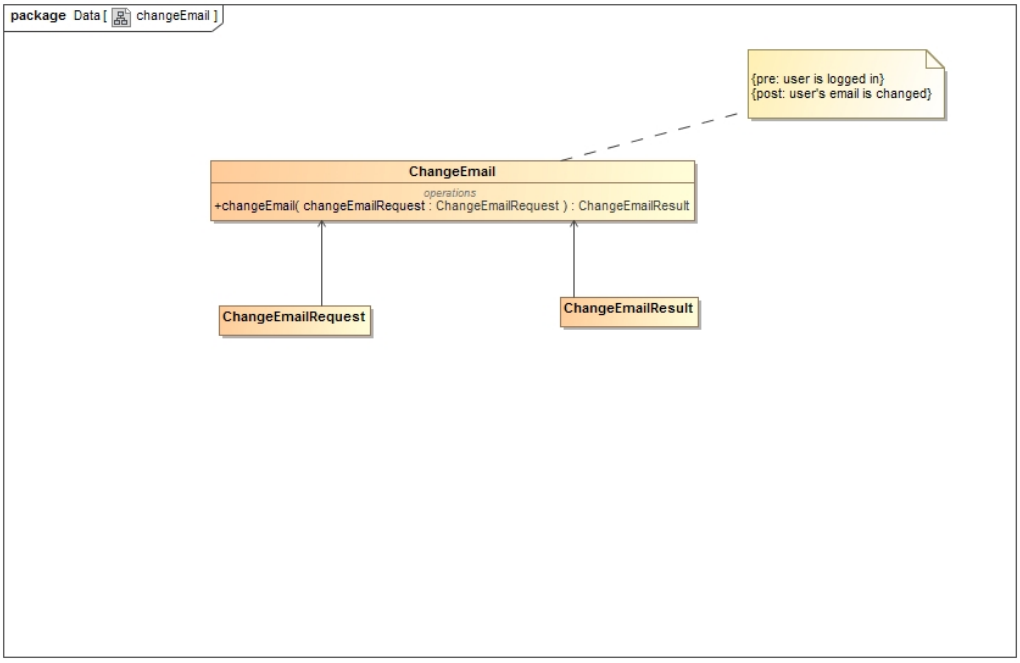

Figure 15: changeEmail service contract

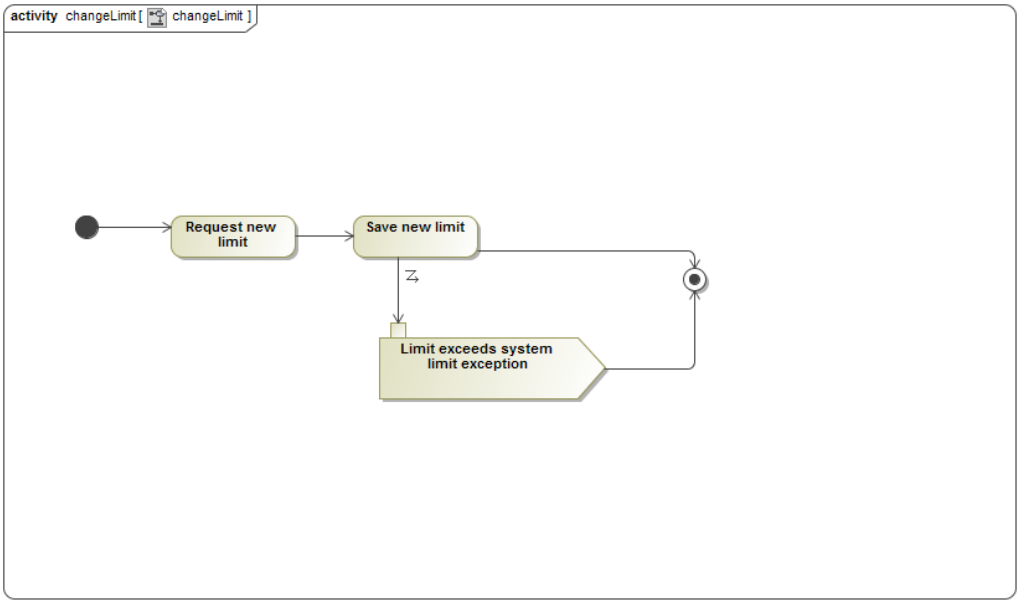

Figure 16: setLimit activity diagram

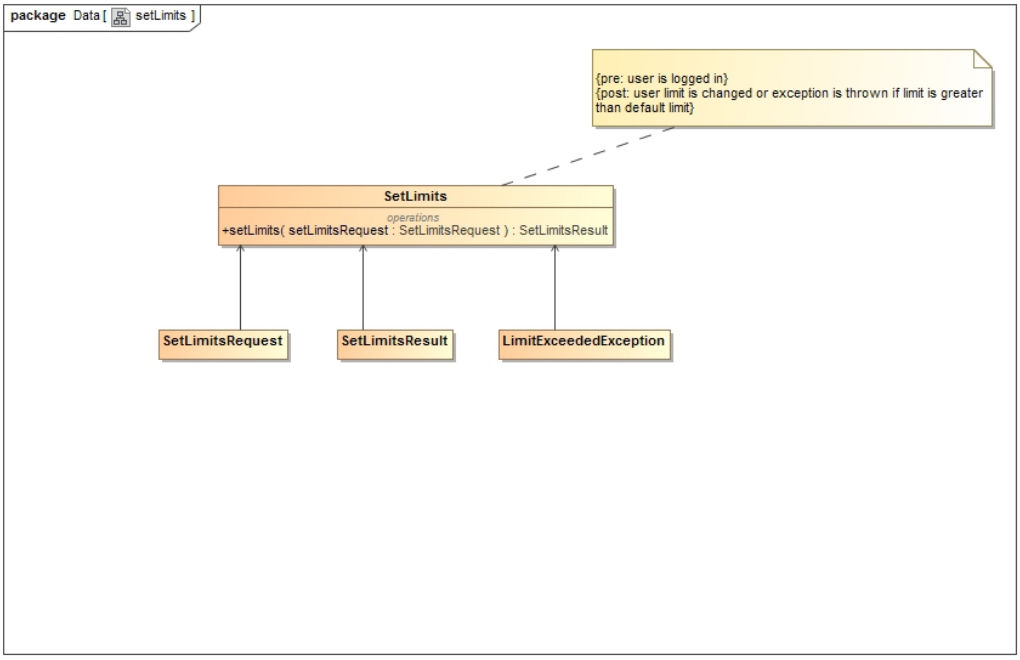

Figure 17: setLimit service contract