

# COSBAS Functional Requirements Documentation

Git: <https://github.com/undecidables/Requirements-Documentation>

**<undecidables>**

Elzahn Botha *13033922*  
Jason Richard Evans *13032608*  
Renette Ros *13007557*  
Szymon Ziolkowski *12007367*  
Tienie Pritchard *12056741*  
Vivian Venter *13238435*

**March 2015**

# Contents

<b>1</b>	<b>Functional Requirements</b>	<b>2</b>
1.1	Introduction . . . . .	2
1.2	Use Cases / Services Contracts . . . . .	2
1.3	Authentication . . . . .	3
1.3.1	Setting of availability . . . . .	4
1.3.2	Alter Appointment . . . . .	5
1.4	Required Functionality . . . . .	7
1.4.1	Authentication . . . . .	7
1.4.2	Setting of Availability . . . . .	8
1.4.3	Alter Appointment . . . . .	8
1.5	Process Spesification . . . . .	9
1.5.1	Authentication . . . . .	9
1.5.2	Alter Appointment . . . . .	9
1.6	Domain Model . . . . .	9

# **1 Functional Requirements**

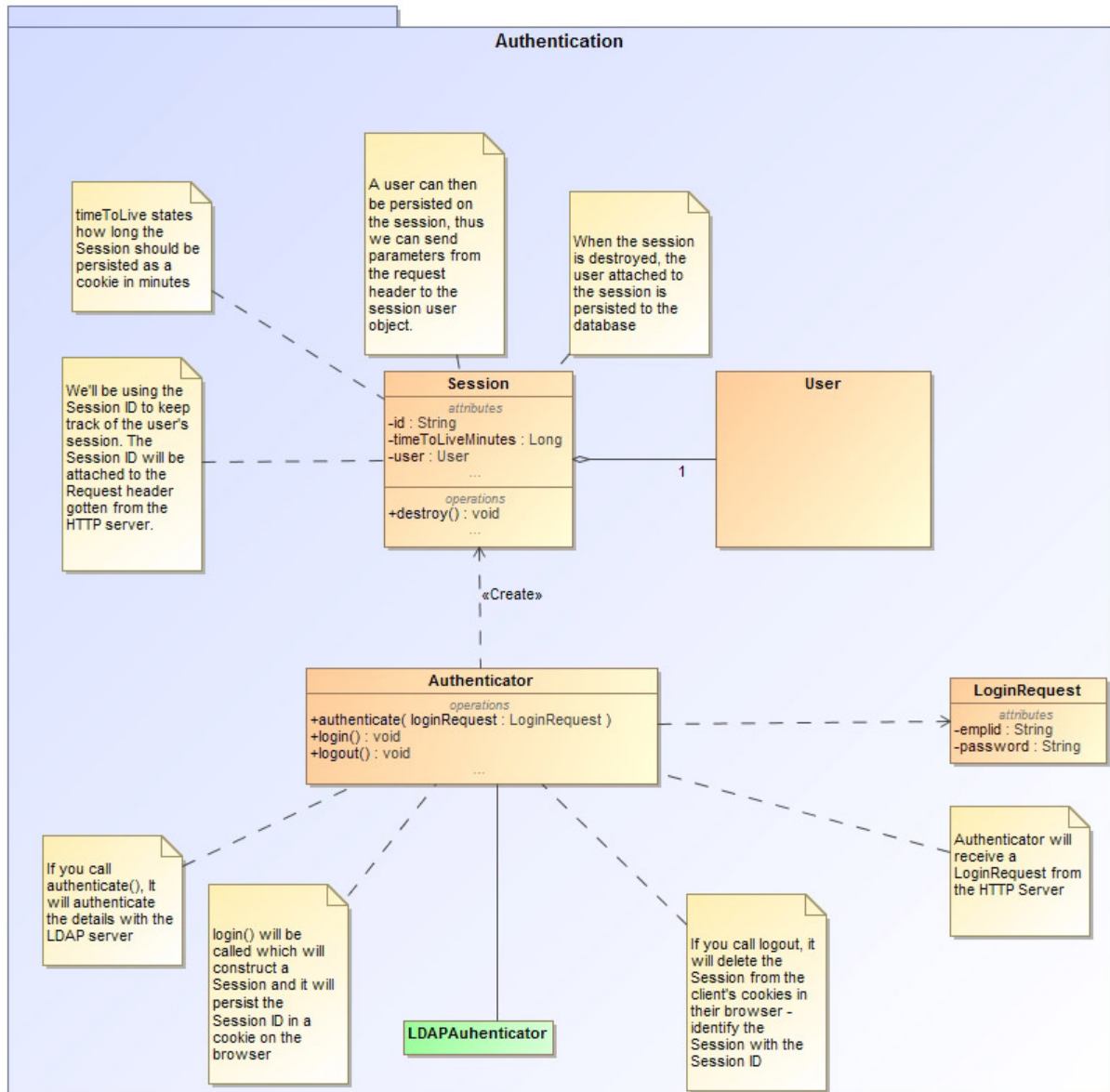
## **1.1 Introduction**

In this section of the document we will identify and address a high level overview of the COSBAS (Computer Science Biometric Access System) system.

## **1.2 Use Cases / Services Contracts**

## 1.3 Authentication

The following section will describe functionality around logging staff members in and out.



### Authenticate

Priority: Critical

A user's details need to be authenticated with LDAP.

#### Pre-conditions:

- A user must have an EMPLID from the University of Pretoria
- A user must have an associated password for their EMPLID from the University
- A successful connection to LDAP is important
- A user must be registered as a staff member on LDAP
- A successful validation response after an LDAP authentication is needed to authenticate a user

#### Post-conditions:

- A user is successfully authenticated on the server

## Log In

*Priority: Critical*

A user must be logged into the system once they have been authenticated. A user is logged in by creating a cookie containing the session ID.

### Pre-conditions:

- A user must have been successfully authenticated by the system to be logged in.

### Post-conditions:

- A user is successfully logged in and can thus access features which require authentication.
- A user is taken to the booking management page on the website

## Log Out

*Priority: Critical*

The system must be able to log a user out. A user is logged out by destroying the cookie containing their session ID.

### Pre-conditions:

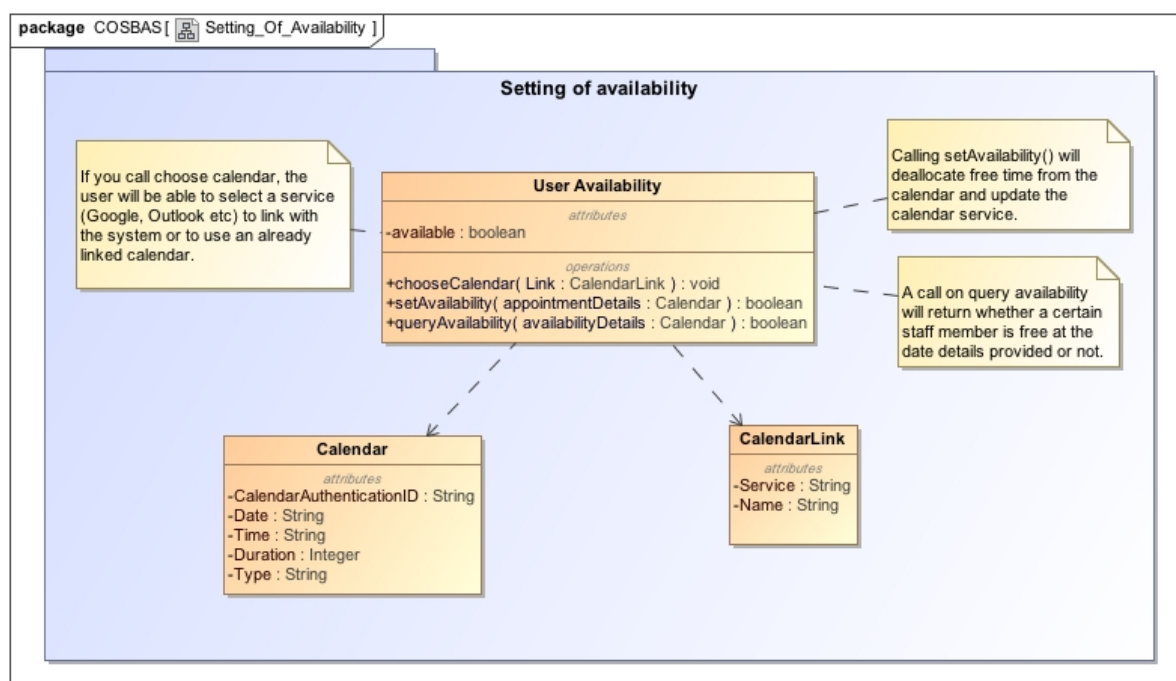
- A user must be logged in before they can be logged out

### Post-conditions:

- A user's session cookie is destroyed and they are logged out on the system and taken back to the home page.

### 1.3.1 Setting of availability

The following functionality of creating and setting will be available to only signed in users. Users who are not authorised or signed into the system will only be granted read-only privileges.



## Choose Calendar

*Priority: Critical*

An authorised user of the system should be able to select which calendar account to associate with the system. If the user chose a linked service, then that service needs to

be authenticated.

**Pre-conditions:** The user should be logged into the system and already have a calendar account from the linked services (Google & Outlook).

**Post-conditions:** Once the user has chosen the calendar to associate with the system, all updates in availability will be amended in that specific calendar.

### View Availability

*Priority: Critical*

Both the authorised user and a non-authorised user (or guest) should be able to have a read-only functionality of viewing the availability of the associated staff member on their calendar.

**Pre-conditions:** An associated calendar for the staff member should exist.

**Post-conditions:** Be able to see whether the associated staff member has an appointment at a certain date and time or if they are free for appointments.

### Set Availability

*Priority: Critical*

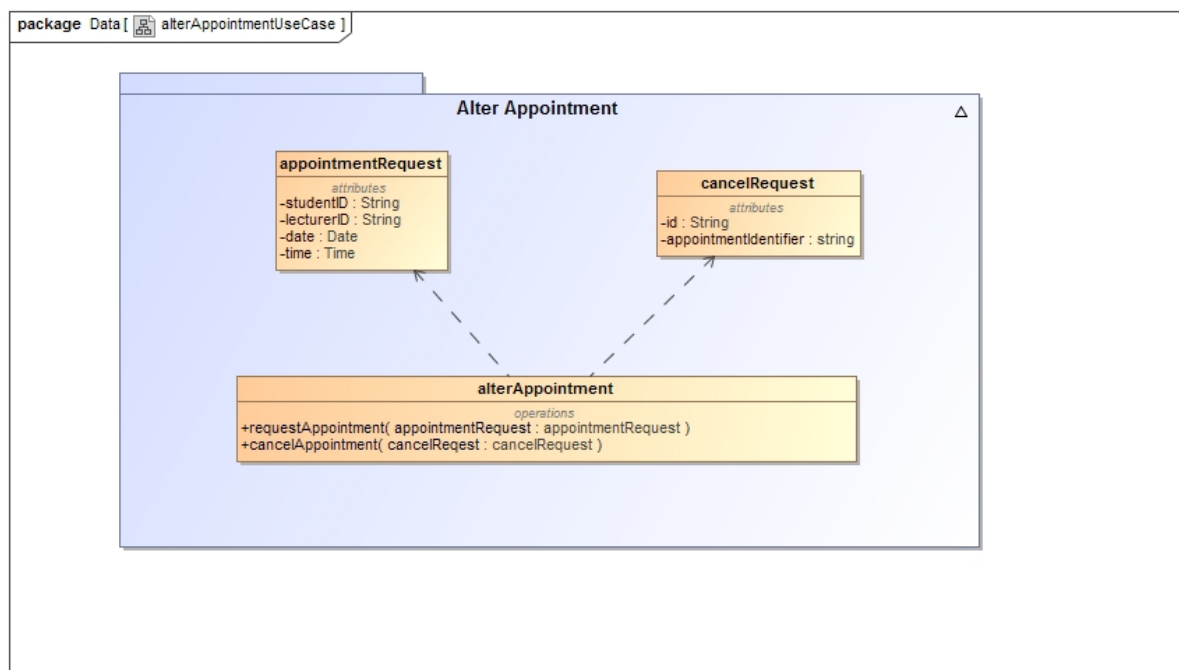
The authorised user should be able to set whether s/he is available at a certain date or time or if they have an appointment at a specific date or time. This can be set by the staff member himself/herself or by the approval of an appointment.

**Pre-conditions:** The user should be authorised and logged in and a calendar should exist.

**Post-conditions:** The linked calendar should be updated with the particulars specified by the authorised user. (CRUD of appointments).

## 1.3.2 Alter Appointment

This section will describe the functionality regarding requesting and cancelling an appointment with a lecturer.



### Requesting Appointment

*Priority: Important*

The requestAppointment function allows a user of the system to request an appointment with a staff member that is also making use of the system.

**Pre-conditions:**

- Lecturer must exist.
- Date and time of the requested appointment must be valid entries.

**Post-conditions:**

- Appointment will be saved for the lecturer to approve or deny later on.
- User gets an appointment identifier back.
- Lecturer is notified of the requested appointment.

**Cancelling Appointment**

*Priority: Important*

The `cancelAppointment` function allows the user to cancel an appointment if the user is either the staff member who's appointment it is or the user who made the appointment.

**Pre-conditions:**

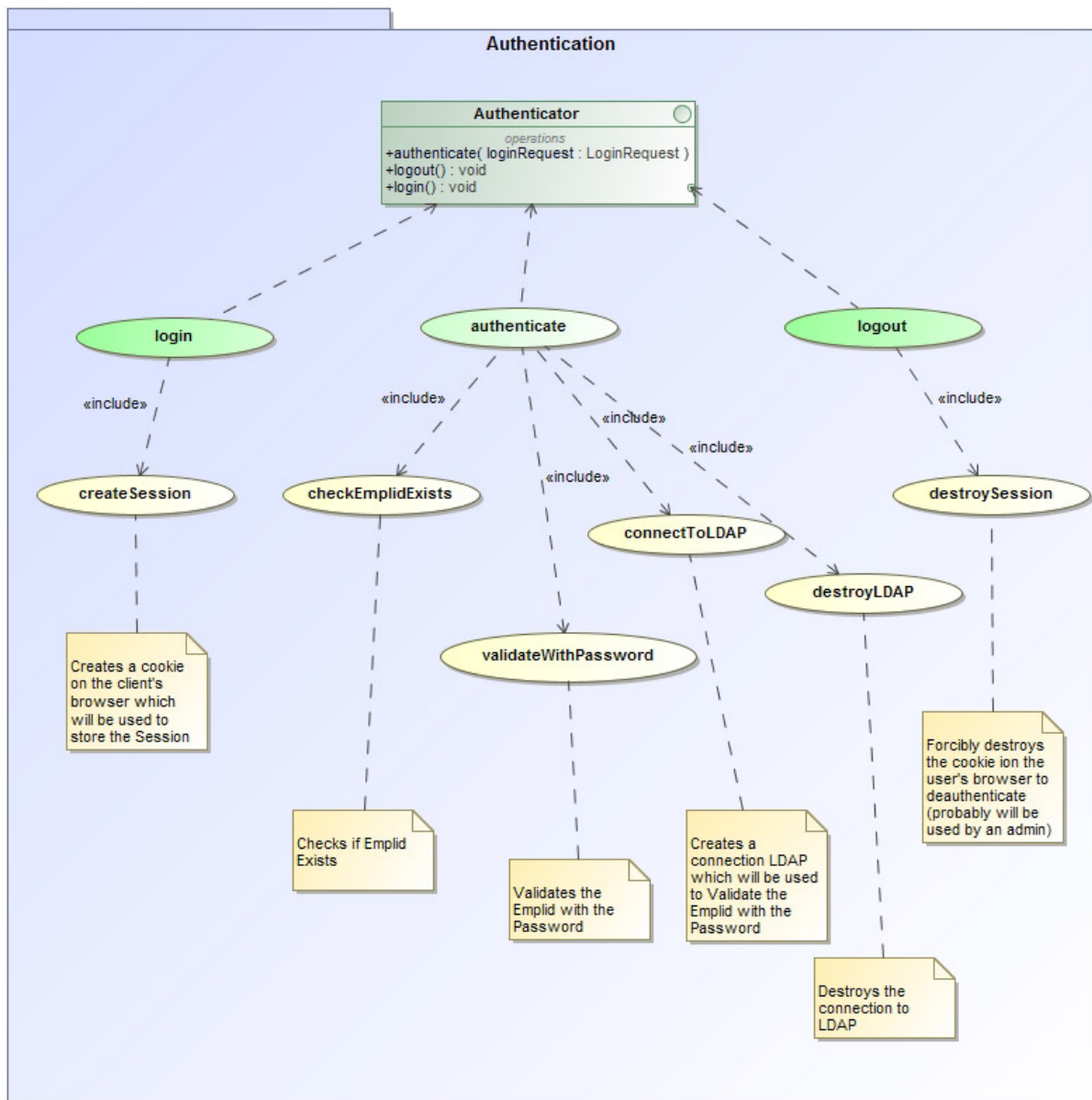
- The appointment must exist.
- The user cancelling the appointment has to be the person that the appointment is with or the person who made the appointment.

**Post-conditions:**

- The appointment will be cancelled.
- Both parties are notified.
- Access that was granted for the appointment is revoked.

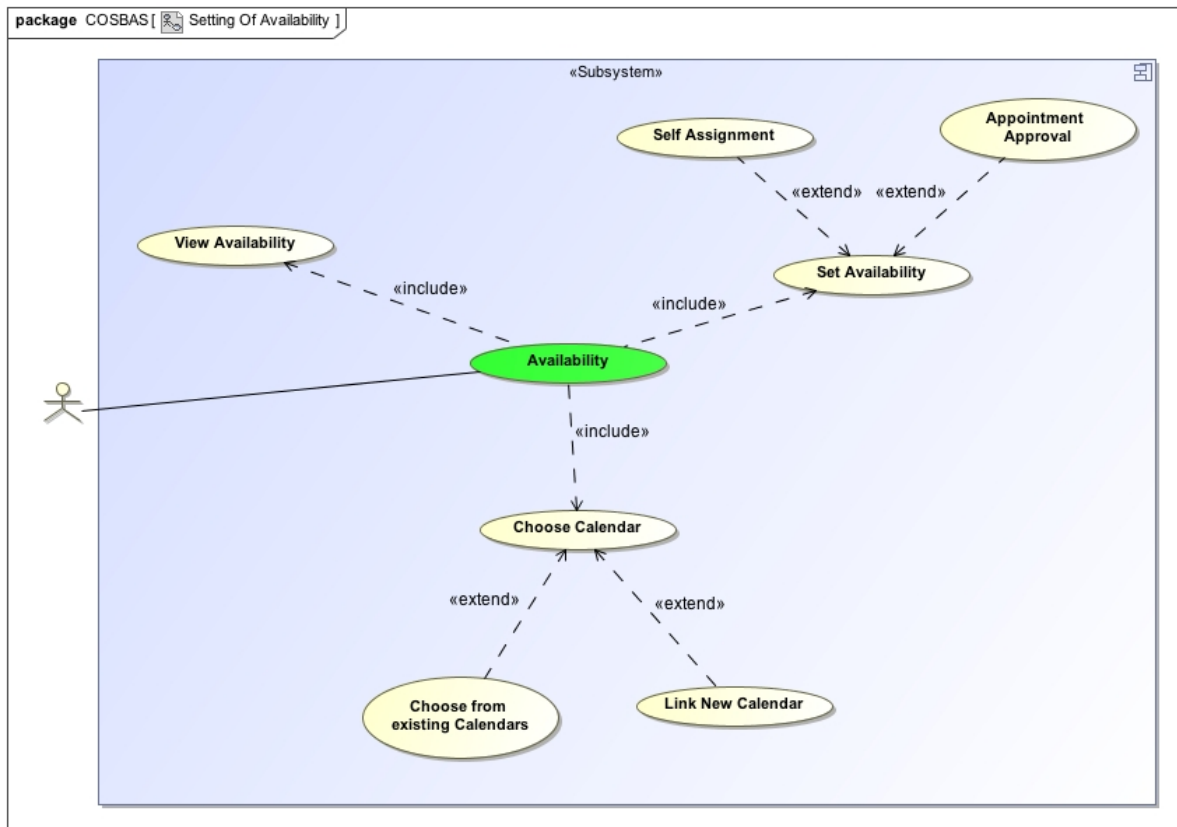
## 1.4 Required Functionality

### 1.4.1 Authentication

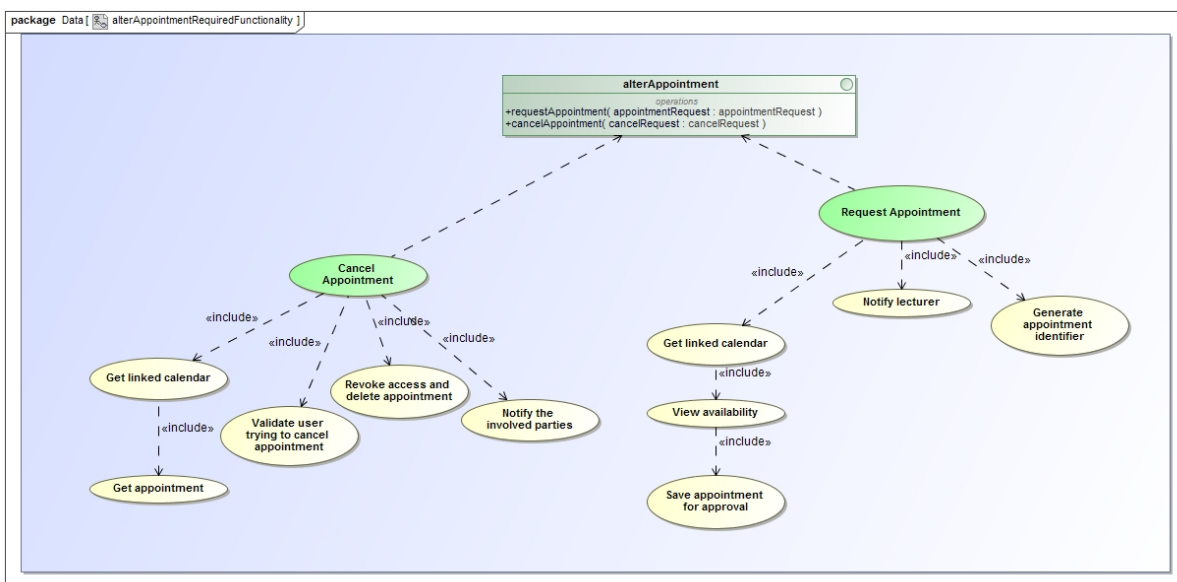




### 1.4.2 Setting of Availability

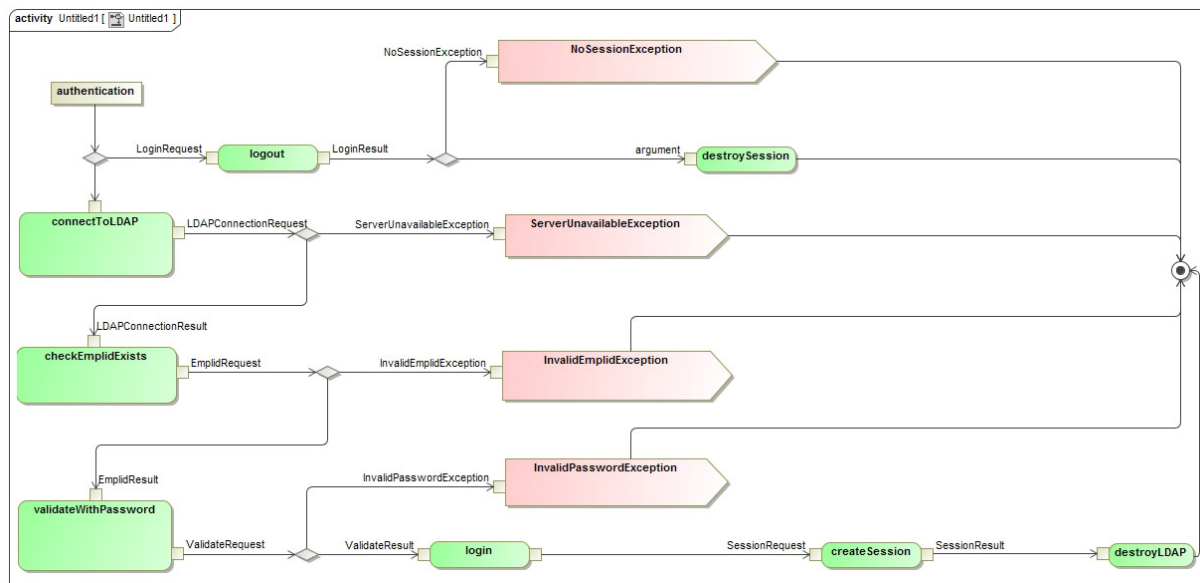


### 1.4.3 Alter Appointment

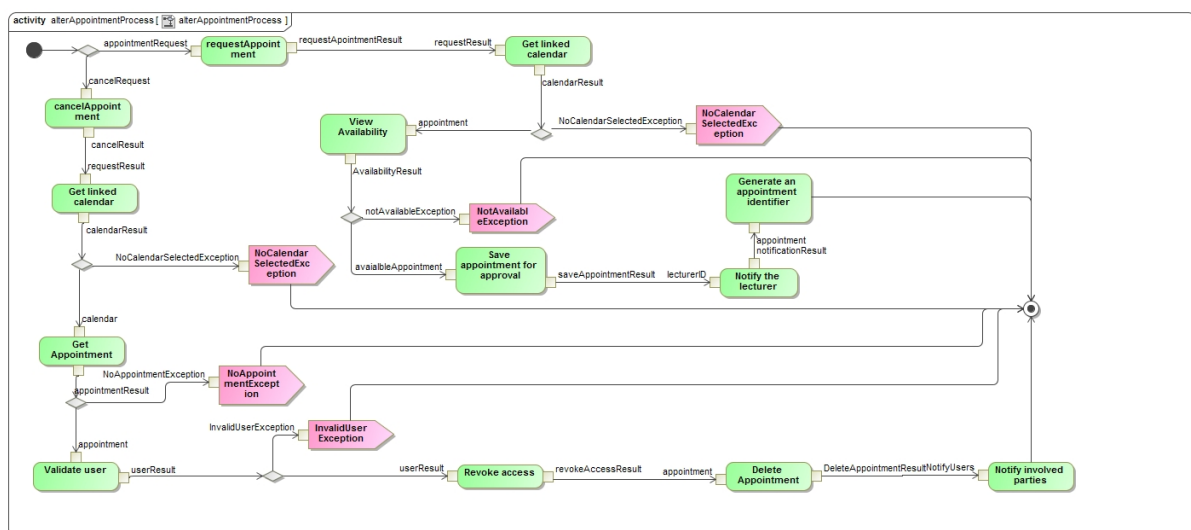


## 1.5 Process Specification

### 1.5.1 Authentication



### 1.5.2 Alter Appointment



## 1.6 Domain Model