Software Requirements Specification

Version 1.2

for

LOTUS Calendar

Prepared by

Alexander Rosser	27543069	arosser95@gmail.com
Costa Papadakos	26665691	cotsop@gmail.com
Philippe Kuret	27392680	philippekuret@gmail.com
Adriel Fabella	27466005	adriel.fab@gmail.com
Saif Mahabub	27392974	rhythmsaif@gmail.com
An Ran Chen	27277385	archen94@gmail.com

Instructor:	Dr. C. Constantinides
Course:	SOEN 343
Date:	23/11/2016

Document history

Date	Version	Description	Author
12/11/2016	1.0	Rough Draft	Alexander Rosser, Philippe Kuret, Costa Papadakos, Adriel Fabella
21/11/2016	1.1	Completion and Revision	Alexander Rosser
22/11/2016	1.2	Revision	Alexander Rosser, Costa Papadakos, Adriel Fabella, Saif Mahabub, An Ran Chen

Table of Content

Table of Figures

Table of Tables

1. Introduction

This introduction provides a description and an overview of the software requirements specifications of our project.

Purpose

The purpose of this document is to provide a report on the requirements for the "Lotus" room reservation system. It will provide an overall description on the development of the system by providing and explaining the system's functions, constraints, assumptions, dependencies and user characteristics. It will also provide specific requirements for development. This document is intended for approval by the stakeholder and for the development team.

Scope

The "Lotus" room reservation system is an online reservation system that allow users to reserve timeslots on a college facility room. Multiple options like create, remove and modify reservation are offered to the user. These options can be performed on every room situated in the directory of rooms. The user can also view all the current reservations made on all rooms to analyze the availabilities. All reservation and wait list information is saved on a database, which will be located on a server.

Definitions, acronyms, and abbreviations

Term	Definition
SSD	System Sequence Diagram

Table 1: Definitions

2. Overall description

Product perspective

The product is a new software tool and not a continuation or adaptation of an existing tool. The product is self-contained and does not rely on pre-existing products to function with the sole exception that it uses a client's pre-existing database to authenticate its user access.
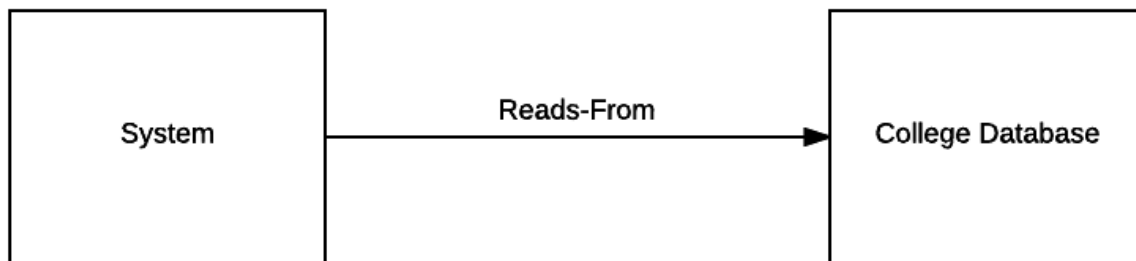

Figure 1: Product Perspective Block Diagram

Product functions

The product allows users to view the current conference room reservations. It also allows them to create new reservations for available time slots, as well as modify and cancel existing reservations. The system permits users to add themselves to a waitlist if the desired room's time slot is already reserved. If a time slot is freed, the system will allocate the time period to the user at the top of the wait list. It shall then remove the user from all other wait lists with the same time slot.

User characteristics

The product is intended to be used by college faculty members. It is expected that the educational level of the average user will be above-average. Their technical expertise is expected to vary greatly and it cannot be assumed that each will have average to above average technical experience. Each user on the system will have the same security, privilege, and accessibility levels.

Constraints

The software is required to allow a multitude of students to view the room reservations simultaneously. The system also has the constraint of providing safety for a write function. Only one student at a time may access a room to create a reservation to allow for mutual exclusion. Furthermore, the product must provide liveness and fairness. The last constraint placed on the software is that it must place a maximum on the number of active reservations per user.

Assumptions and dependencies

The product will authorize access to faculty members registered to the college. To comply, the software depends on access to the school's existing database to authorized personnel access. It is assumed that the product has no requirement of creating or modifying user profiles to the database as these functions will be handled by the school's existing systems.

3. Specific requirements

Functionality

Functional requirements capture the intended behavior of the system. This section contains the *Actor Goal List* and the *Use Case view*.

Actor goal list

Actor	Goal
User	To log in
	To log out
	To view all bookings of a specific date
	To change and view the bookings of a different date
	To make a reservation
	To cancel a reservation made by the user
	To modify a reservation made by the user.

Table 2: Actor Goal List

Use case view

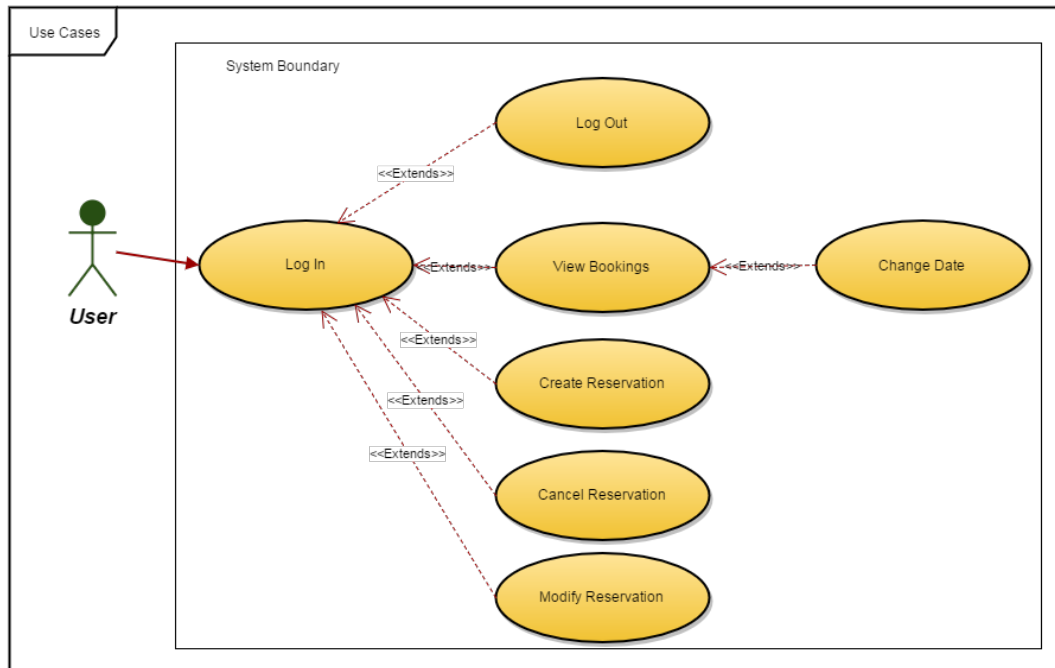The use case model is shown in Figure 2.

Figure 2: Use Case View

Reliability

The system shall be able to accommodate multiple users simultaneously. The system will ensure that all data from a user cannot be accessed or modified by other external parties. The system shall allow only registered users to access the functionality of the software.

Usability

The System shall be understandable without any explanations required. The User shall need no longer than 5 minutes to become familiar with the System. The User shall not need any training in order to operate the system effectively.

Efficiency

The System shall take no longer than 3 seconds to perform any operation. The System shall use no more than 70% resources when operating below the maximum user base.

Maintainability

Maintainability factor is decomposed into the 4 criteria: Analyzability, Changeability, Stability and Testability. This evaluation is taken from the standard quality models of ISO 25010. The components of the System shall not be highly coupled. The System shall be reusable for the implementation of any reservation or waitlist systems. Impacts or defects initially present or possibly caused due to recent modification of the system shall be easily assessed and analyzable. The System shall be relatively modifiable and be able to be modified for a variety of different reservation systems and needs. The System shall be easily testable for all critical scenarios.

Portability

The system shall be highly adaptable. The system shall be easily installable across multiple environments without much modification or support needed.

Design constraints

The architectural style of our project is Object oriented. The possible constraints of the program include:

- **Size**: the program will contain a great amount of lines of code
- **Effort and Time**: time will be greatly spent on design before the actual implementation
- **Speed**: due to the size of the program, the performance of the program may be reduced

4. Analysis Model

This section contains analysis models to straighten the understanding of previously mentioned requirements and constraints.

Domain Model

The domain model displays real-world concepts of the system. The model demonstrates the relationships between objects. The domain model is shown in Figure 3.
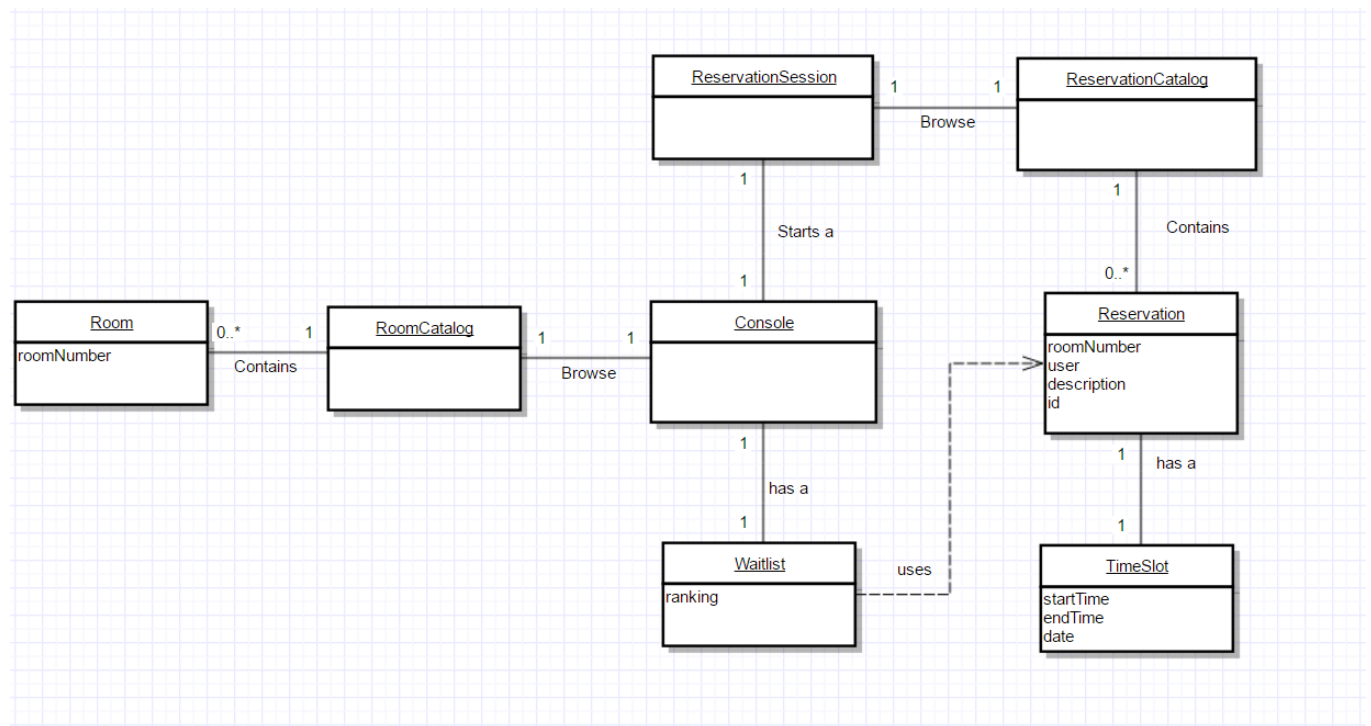

Figure 3: Domain Model

Use Cases

The following are the use cases seen in the use case model of Figure 2. The use cases describe the list of actions between a user and the system to achieve a goal. Use cases with an importance of 5/5 are the system critical use cases.

Name	UC1 – Log in
Importance	2/5
Difficulty	2/5
Primary Actor	Public User
Goal	Securely log the user into the system.
Preconditions	1. Homepage is open. 2. User is not already logged in.
Success Guarantee	User is logged into a secure session.
Summary	User enters their username and password to access the scheduling system with restrictions based on their account type.
Main success scenario	1. User enters username and password. 2. User selects to log in. 3. System verifies the information.
alternative flows	If invalid credentials, indicate error. If user is already logged in, indicate error.

Table 3: Use Case Log In

Name	UC2 – Log out
Importance	2/5
Difficulty	1/5
Primary Actor	Registered User
Goal	Securely log the user out of the system.
Preconditions	User is logged into the system.
Success Guarantee	User is logged out of the system. User is redirected to the home page.
Summary	User logs out of the system.
Main success scenario	User selects to log off the system.

Table 4: Log Out Use Case

Name	UC03 – View Bookings
Importance	5/5
Difficulty	2/5
Primary Actor	Registered user
Goal	View the reservations and availabilities schedule.
Precondition	User is logged in.
Success Guarantee	User is redirected to the current day's schedule.
Summary	User is displayed the bookings for the current day.
Main success scenario	User selects to view the booking's schedule.

Table 5: View Bookings Use Case

Name	UC04 – Change View Booking Date
Importance	4/5
Difficulty	2/5
Primary Actor	Registered user
Goal	Change the viewed date.
Precondition	1. User is logged in. 2. User viewing the booking's schedule.
Success Guarantee	1. User is displayed the selected day's schedule.
Summary	User selects a day to be displayed.
Main success scenario	1. User selects to change the viewed day. 2. User selects the day to be viewed.

Table 6: Change View Booking Date Use Case

Name	UC05 – Create Reservation
Importance	5/5
Difficulty	3/5
Primary Actor	Registered user
Goal	Create a new reservation
Precondition	1. User is logged in. 2. User is viewing the booking's schedule. 3. Selected room is available to modify.
Success Guarantee	1. A new reservation is saved. 2. The rooms availability is set to false for the selected time slot.
Summary	User creates a new reservation
Main success scenario	1. User selects a room to reserve. 2. User selects a timeslot to reserve. 3. User validates his selections. 4. System validates information.
alternative flow(s)	1. If the timeslot is not available, user is asked if they wish to be added to the wait list. 2. If the maximum number of reservations is reached by the user, prompts error.

Table 7: Create Reservation Use Case

Name	UC06 – Drop Reservation
Importance	5/5
Difficulty	3/5
Primary Actor	Registered user
Goal	Cancel a pre-existing reservation
Precondition	1. User is logged in. 2. User has at least one reservation. 3. User is viewing their reservation(s).
Success Guarantee	1. Selected reservation is deleted. 2. Timeslot is set to available. 3. System consults the waitlist and if it is not empty: 3.1 Creates a new reservation for the first user. 3.2 Deletes users waitlist reservations with the same timeslot. 3.3 Updates the waitlist.
Summary	User cancels a reservation and system consults the waitlist.
Main success scenario	1. User selects a reservation. 2. User selects to cancel reservation.

Table 8: Drop Reservation Use Case

Name	UC07 – Modify Reservation
Importance	5/5
Difficulty	3/5
Primary Actor	Registered user
Goal	Modify a pre-existing reservation
Precondition	1. User is logged in. 2. User has at least one reservation. 3. User is viewing their reservation(s).
Success Guarantee	Selected reservation is modified accordingly.
Summary	User modifies a reservations information.
Main success scenario	1. User selects a reservation. 2. User selects modify the reservation. 3. User changes reservation information. 4. System validates information.

Table 9: Modify Reservation Use Case

System Sequence Diagrams

The SSDs for the three critical use cases are provided below. They display the interactions between users and the system for the designated use case.
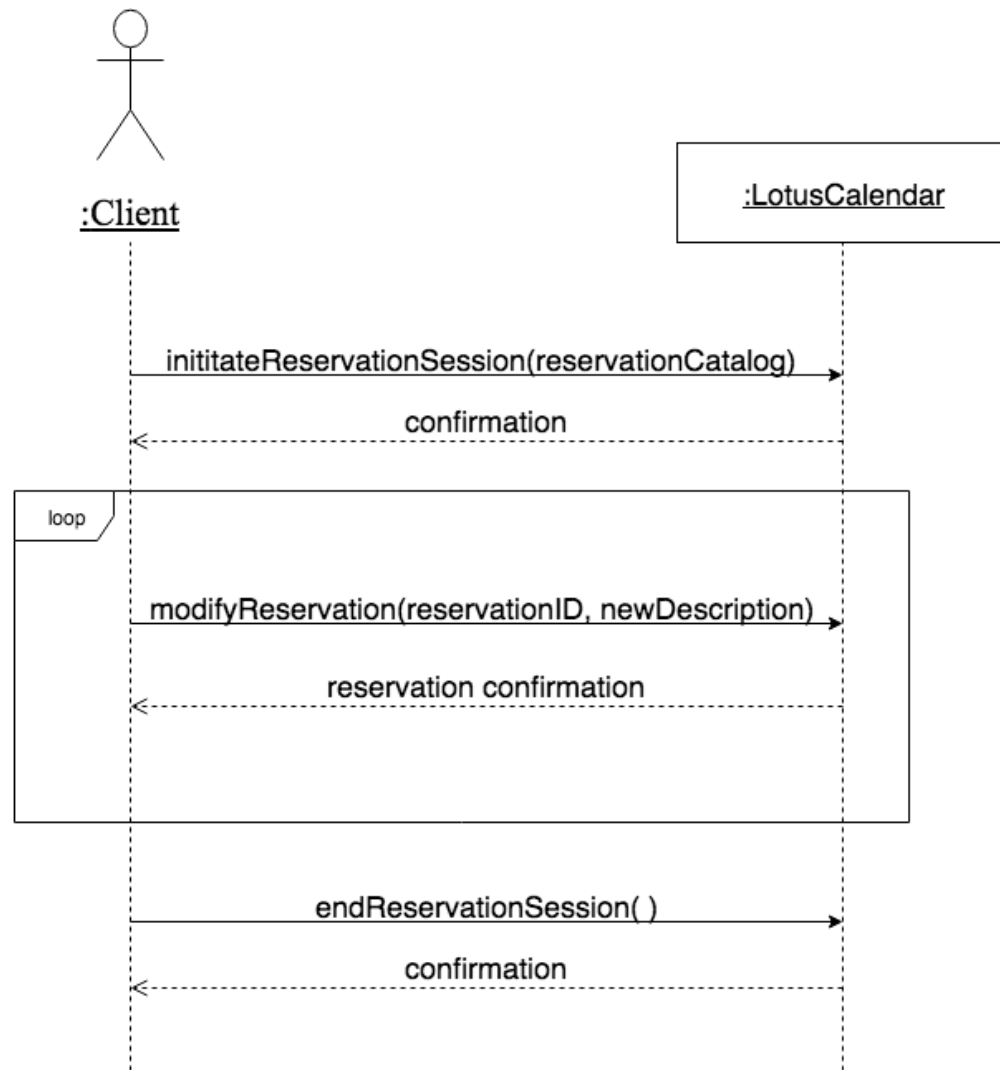
Figure 4: SSD UC07 Success Path
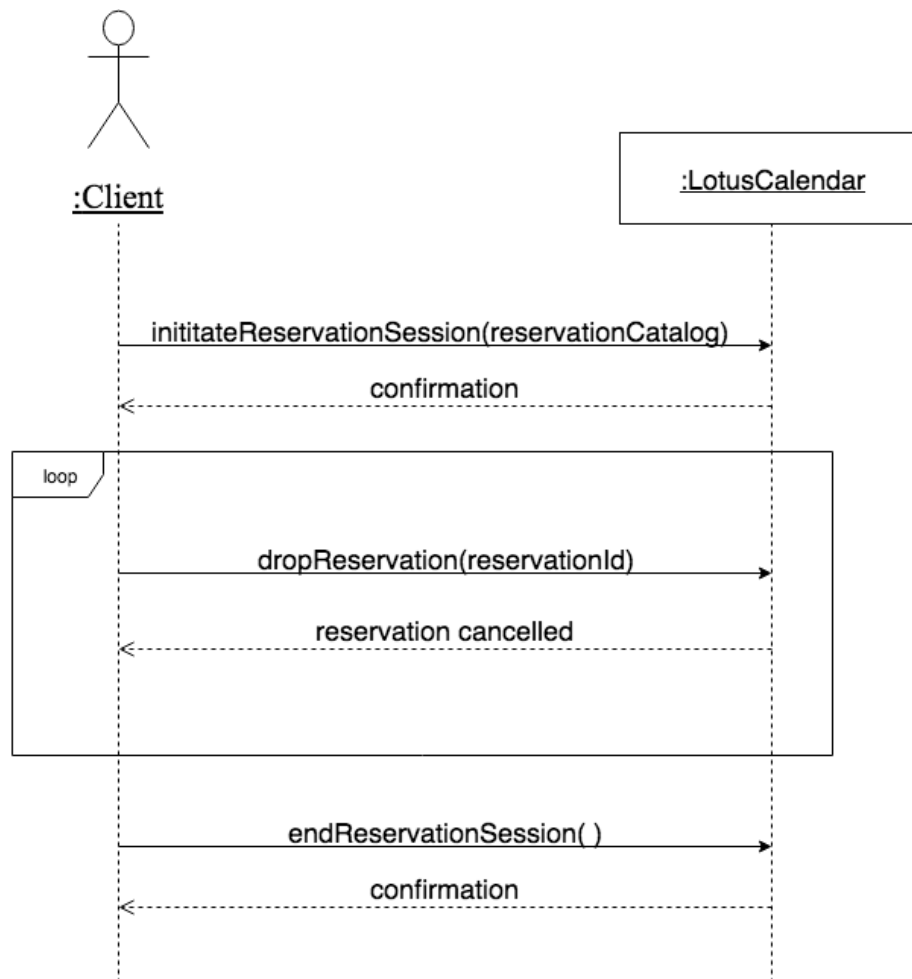
Figure 5: SSD UC06
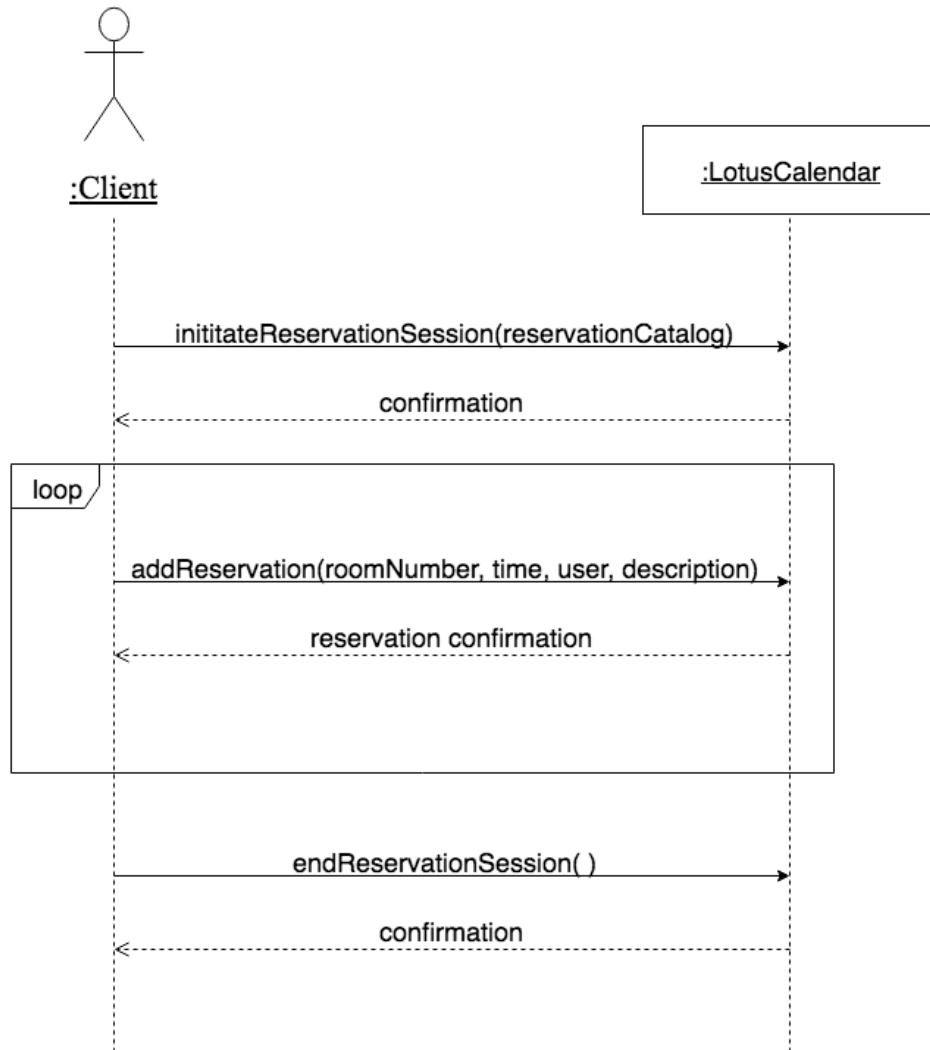
Figure 6: SSD Use Case 05 Success Scenario
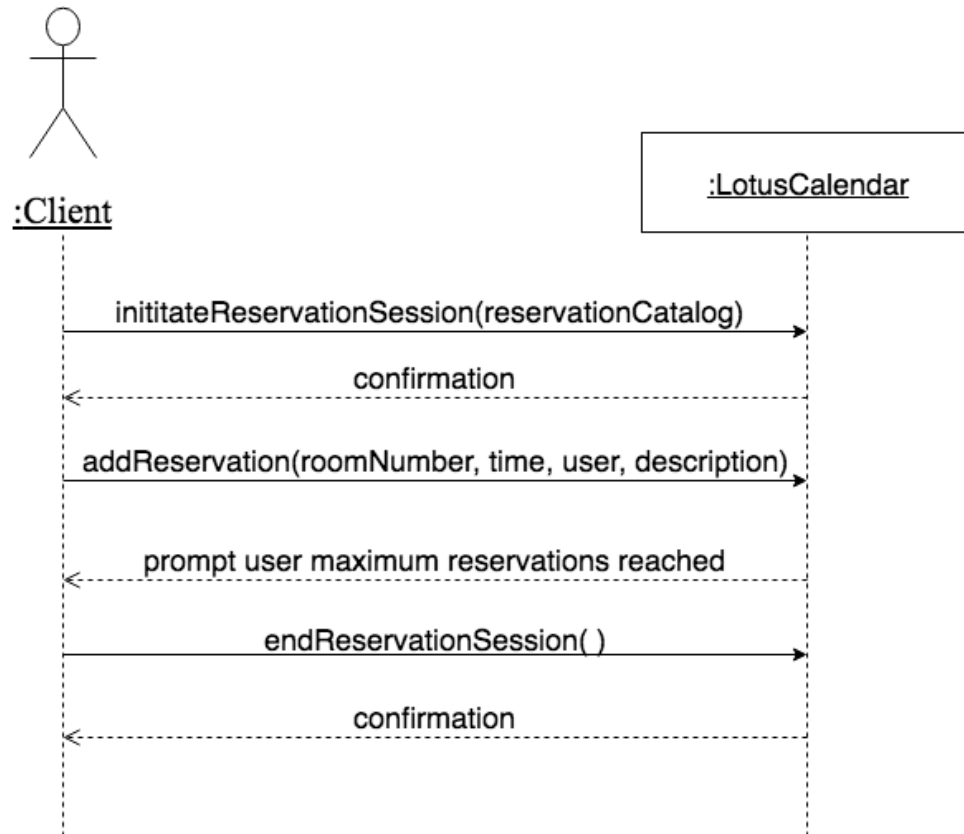
Figure 7: SSD Use Case 05 Alternate Path 1
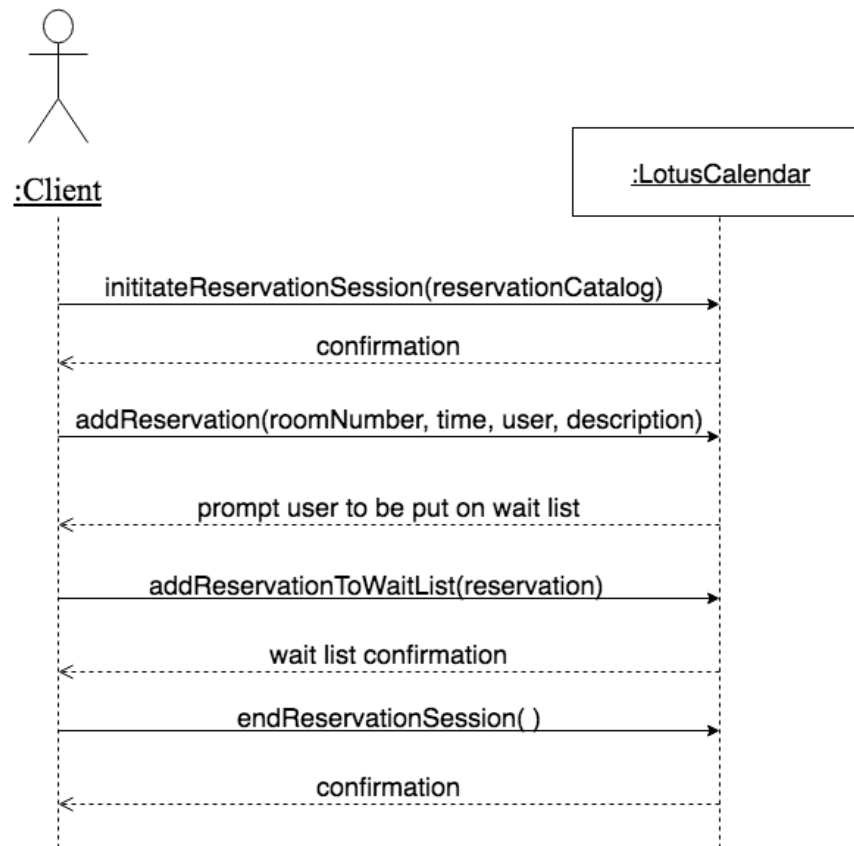
Figure 8: SSD Use Case 05 Alternate Flow 2

System Operations

The Following figure displays the system operations and their signatures.

Figure 9: System Operations

State Diagrams

The state diagrams for the add, modify and drop reservation scenarios are presented to demonstrate the behavior of the system. These diagrams display the events necessary for the system to enter different states.
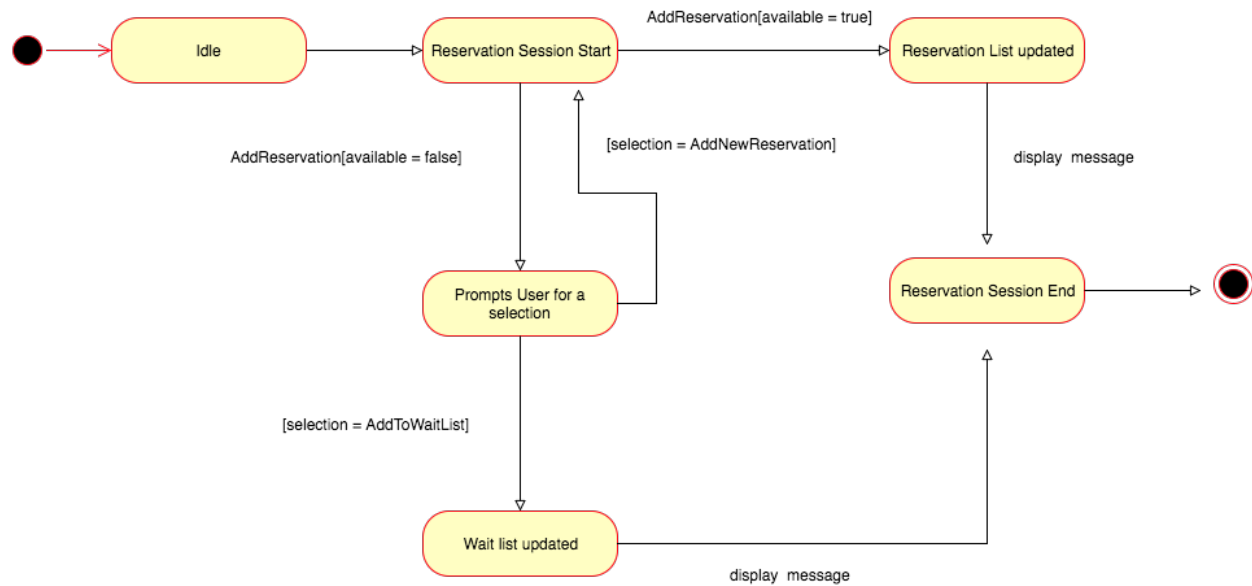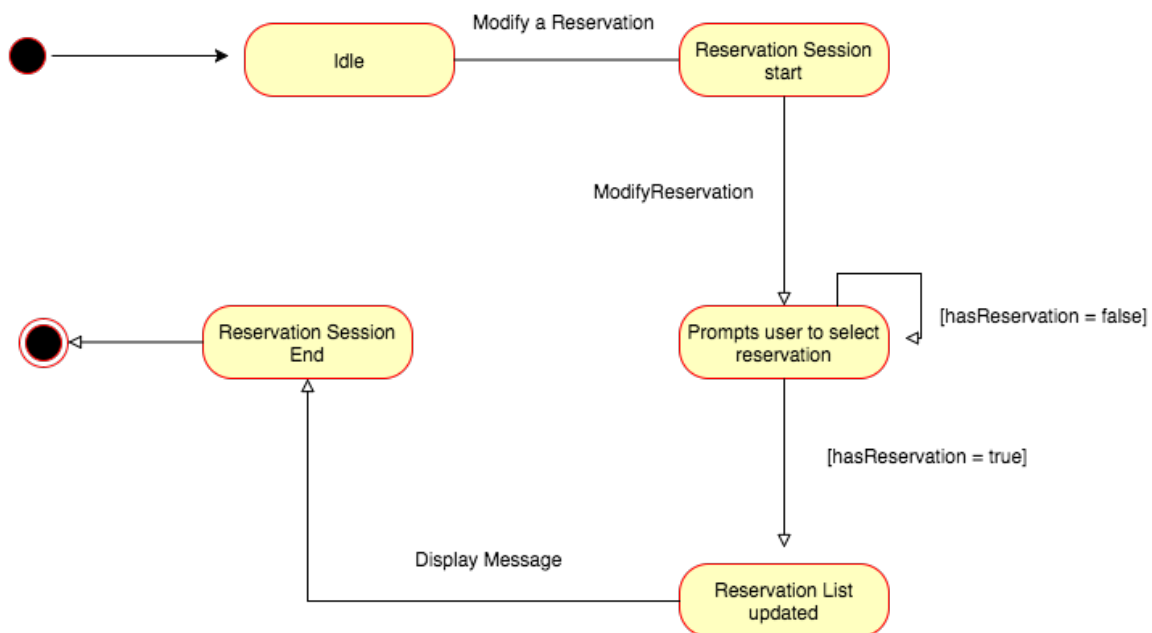

Figure 10: Add Reservation State Diagram
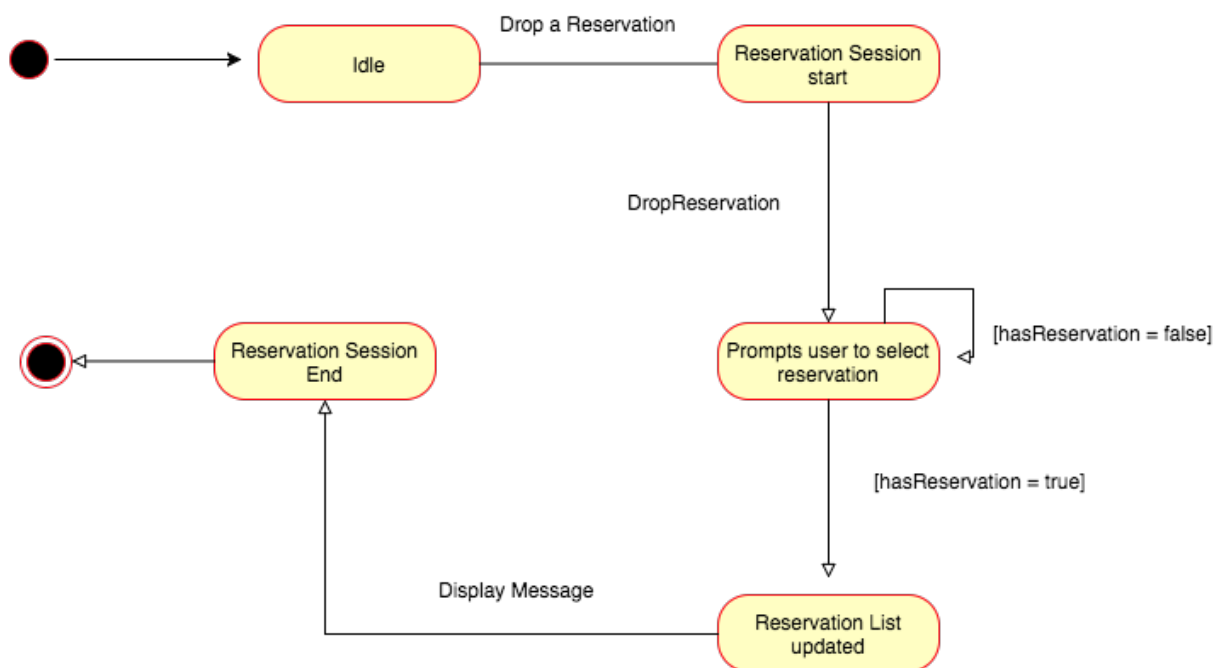
Figure 11: Modify Reservation State Diagram

Figure 12: Drop Reservation State Diagram