Software Requirements Specification

Version 1.9

for

QuickBook Conference Room Reservation System

Prepared by

Piratheeban Annamalai	27755708	piratheeban95@hotmail.com
Andrew Laramee	27050925	zephix109@gmail.com
Jacqueline Luo	26938949	luo.jackie@outlook.com
Michael Mescheder	27202202	meschedermichael@gmail.com
Hoang Khang Nguyen	27079427	alphakennyn@gmail.com
Eric Payette	27008058	eric.payette24@gmail.com

Instructor: Dr. Constantinos Constantinides

Course: SOEN 344

Date: 04/04/2017

Document history

Date	Version	Description	Author
10/10/2016	1.0	Initial SRS	
11/18/2016	1.1	Modified SSD, contracts	
11/20/2016	1.2	Modified Domain Model	
3/25/2017	1.3	Modified section 1,2,4 (new reqs)	Piratheeban
3/26/2017	1.4	Modified section 3 (new reqs)	Eric Payette
3/26/2017	1.5	Modified section 2.4	Jacqueline
3/27/2017	1.6	Added state diagram and modified critical use case	Eric, Jacqueline, Michael
3/27/2017	1.7	Updated SSD and operation contract for makeReservation	Eric, Jacqueline, Michael
3/29/2017	1.8	domain model update	Eric, Michael
4/03/2017	1.9	Testing section	Piratheeban, Hoang Khang

Table of contents

List of figures

1. Introduction

1.1. Purpose

This document provides a detailed description and specific requirements of the *QuickBook* conference reservation system. The nature of this product is an academic one and therefore, the document is intended for developers of the application and will be submitted to the professor for review and assessment.

1.2. Scope

This application will be used by members of the Engineering and Computer Science (ENCS) faculty from Concordia University to reserve a conference room. The number of rooms in the facilities are limited, thus a reliable and efficient reservation system is needed in order to guarantee a room is available for use. Users of the system will be able to reserve available rooms for their team via an online system, accessible everywhere where an internet network is available. The reservations can also be canceled should the user change their mind. By providing students and other members of the ENCS faculty with an accessible way of reserving rooms, the application will not only make it convenient for them, but it will also lessen the responsibilities of the manager or administrator of the facilities to manually manage the reservations. Moreover, this application will have imposed restrictions on how many rooms a single user can reserve per week as well as the number of consecutive time slots occupied. Students who are doing their capstone will have priority over other students when trying to book a room. Also, users will be able to reserve equipments such as projectors, computers, etc. This will place a constraint when users are trying to reserve a room. If the room is available for a specific time slot and the equipment isn't, then the user will be placed on the waiting list until the equipment becomes available (another student canceled his reservation).

1.3. Definitions, acronyms, and abbreviations

Term	Definition
User	A user who can create, modify and delete reservations
Guest	A person viewing the application. A viewer can only view and thus cannot modify or create a reservation.
Reservation	A reservation is made by a user for a room. It includes the room number, a timeslot and the name of the user who made the reservation.
Waiting List	A list of users who are waiting for a room to become available for a certain timeslot
Registry	List of all the reservations

2. Overall description

The QuickBook conference room reservation system will allow ENCS students and faculty members to reserve rooms that have available time slots. It will be accessible and usable by users already registered in the system. Among the users, capstone students will have priority over others. Users will also be able to reserve an equipment when reserving a room. However, they will only be able to reserve a room with an available time slot if the desired equipment is also available; otherwise, the student will be placed on the waitlist. QuickBook is a new self-contained product that will be created in the form of a web application. The user will interact with the displayed graphical interface in a web browser to make use of the application's functions. The web server will handle the logic and the requests received and sent back to the client browser. It will handle all the access controls, constraints and will provide functionality to the application. Since most of the information shown on the interface will need to be stored somewhere, a connection exists between the web server and a database. The information shown to the user on the web application will be a result of the communication between the web server and the database where requests will be sent to retrieve, modify or delete data.
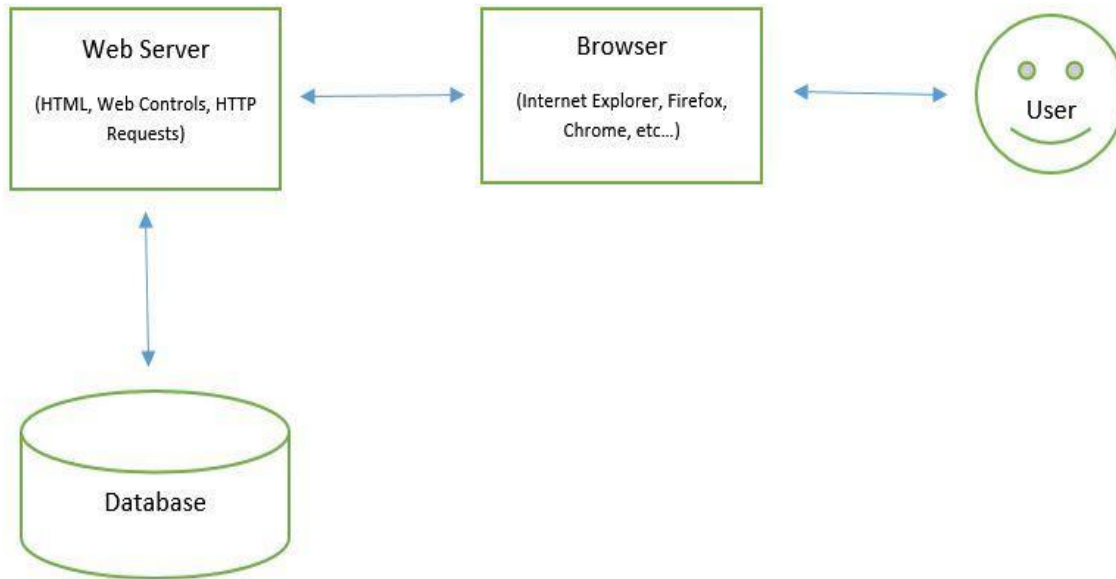
2.1.   Product perspective

Figure 1 Product Perspective Block Diagram

2.2.   Product functions

With this online reservation application, 5 rooms are available on a 24 hour basis. Authenticated users will be able to reserve rooms for a maximum of 3 hours per week. Students who have already reserved a room cannot be placed on a waiting list for any other reservations. Also, the student cannot repeat the same reservation for more than 3 weeks. Creating, modifying or deleting a reservation are the main functions a user can do. Users are also able to view all the rooms in the system. When a user accesses a room, the system will allow other users to access the same room only in read mode as to prevent unfairness and errors. If a room is already reserved for a given timeslot, users can place themselves on a waiting list for that timeslot. Upon room cancellation of the current reservation, the first user on the waiting list can obtain the reservation, in which case he or she is removed from any other waiting list over the same time slots for the same day. Users are restricted by the system on the number of

reservations possible for a week as well as the number of consecutive timeslots for a reservation. The system will also restrict the number of weeks a reservation can be repeated.

Students will be able to reserve equipments (computers, projectors, etc.) when trying to reserve a room. If a room is available at a specific time slot and the desired equipment isn't, then the student trying to reserve the room with the equipment will be placed on waiting list.

Capstone students will have priority to reserve rooms. If a capstone student tries to reserve a room that is already reserved by another student, then he will be placed at the top of the waiting list because he has priority over other students. If there is already a capstone student in the waiting list, then he will be placed right after him.

When viewing the reservations, the time slots of the room are represented in the form of a calendar. The available rooms and reserved rooms can be distinguished on the calendar using different colors. From the calendar, users can select a time slot to reserve the room or enter a waiting list if it is already reserved.

2.3. User characteristics

The intended users of the system are users registered in Concordia University, such as the students and faculty members belonging to the Engineering and Computer Science (ENCS) programs. Users can make use of the system to create reservations, edit reservations, view the status of reservations, or cancel their reservations. They are the intended users for the system in order to reserve rooms for a conference or for working on projects, for instance. Their allowed actions are limited to the CRUD (create, read, update, delete) access to room reservations. They do not have CRUD access to other users' reservations, nor can they view another user's reservations profile. These users are expected to be technically literate and understand how to navigate through a web application.

2.4. Constraints

The system provides several constraints and limitations that affect the users. A couple of them are based on the regulations of the application, while others are more dependent on the technology that supports the system.

- A regulation based on the system's requirements is that each room may be reserved for, if space permits, a maximum of 3 consecutive hours per user. Once a student has reserved a room for 3 hours, he cannot be placed on the waiting list for another reservation. In addition, a user may only create one reservation per week to prevent overbooking and give all users a chance to find a time slot suitable for themselves. In the case where a user wants to reserve a time slot that is already booked by another user, they will be placed on a waiting list designed as a queue, which in case of the original user deleting their reservation, will secure the time slot.

- To ensure safety and fairness in the system, limitations are placed on the user's operations. Only one user can access a certain room at a given time to create, edit or delete a reservation. Another user attempting to access that room will be prohibited until the current user is finished with their operation. Safety of the system is maximized by placing constraints on the users accessing the application. The users of the system are all registered with the ENCS faculty, and this is guaranteed by authenticating them with ENCS usernames and passwords to allow them access to the system console.

- The performance of the system is also affected by constraints. The database is used to hold user information, reservations, users on waiting lists and a list of rooms. The time of an operation being performed can be affected by the databases fetch requests, and therefore provides limitations in the system performance. Hardware constraints are limited to internet connection. A user

with no internet connection cannot connect to the application's web server, which means they cannot access the system.

● A constraint is placed on the user when he tries to reserve a room that is available at a specific time slot, but an equipment desired by the user isn't. In this case, the user will be able on the waiting list until the equipment becomes available to him.

● To prevent a person from booking a room forever, a constraint exists that prevents a person from making the same booking for more than 3 weeks. Same booking in this case refers to the same room and time for more than 3 weeks.

2.5.   Assumptions and dependencies

In order for the reservation system to function as intended, there are a couple of assumptions and dependencies placed on the situation.

- As stated in the constraints section, an assumption for proper use of the system is that the user is assumed to have a strong and stable internet connection. No connection means no way of accessing the server and the console, a weak internet connection means that user operations can be slowed down leading to inefficient use and possible misuse of the system, and an unstable connection leads to possible network crashes or difficulty accessing the database on the client-side.

- Certain web browsers may not handle the website and web server in the same way as it was intended. The system is optimized to run with perfect performance on the Google Chrome web browser, therefore it is assumed that other possible web browsers are able to handle the server and console in a similar way as it was intended.

- The system's intended use depends on pop-up windows being enabled. This is because a user wanting to create or modify a reservation opens a JavaScript pop-up window that displays all available options. If a user has pop-ups disabled, the user won't be able to view the reservation creation or editing window.

3. Specific requirements

This section contains all requirements in detail: Functional as well as non-functional requirements (quality attributes and constraints). The quality attributes are listed according to the *ISO/IEC 25010* standard that classifies software quality in a structured set of characteristics and sub-characteristics.

3.1.   External interfaces

User Interface

A user of web application should see the calendar upon accessing the reservation system. If the user wishes to make a reservation or make changes to an existing reservation (such as canceling or modifying a reservation), the user must log in. Only registered members of the ENCS faculty have credentials to access the system. After logging in, the user will be directed to a page where he/she is able to see a calendar displaying the facilities and their availabilities for a given time period. The user can reserve any one of the available time blocks for any of the rooms. The user has to click on a time block and drag the mouse vertically until their desired time block. Once this action is done, a pop up form will appear on which the user has to submit the request to reserve a room for a certain number of time blocks. A user can also places him or herself on a waiting list for a specific timeslot of a given room that is already reserved.

Hardware Interface

The hardware components are the following: any number of computers, a server and a database. The operating system of the computer and the server will manage the hardware connection to the MySQL database.

Software Interface

The database and the web portal communications consist of every operation that modifies, creates and reads data stored in the database. The web application has to be able to make such communications with the database for every user of the system.

Communications interfaces

Since the system is dependent upon many different hardware and software components, the communication between them is important for the success of the system as a whole. On the other hand, the details of how this communication is made can be negligible since they are handled by the operating system.

3.2. Functionality

This section describes the functional requirements of the QuickBook conference reservation system.

3.2.1 Actor goal list

Actor	Goal
User	Make reservation
	Cancel reservation
	Modify reservation
	View reservations
	View room availabilities

3.2.2   Use case view
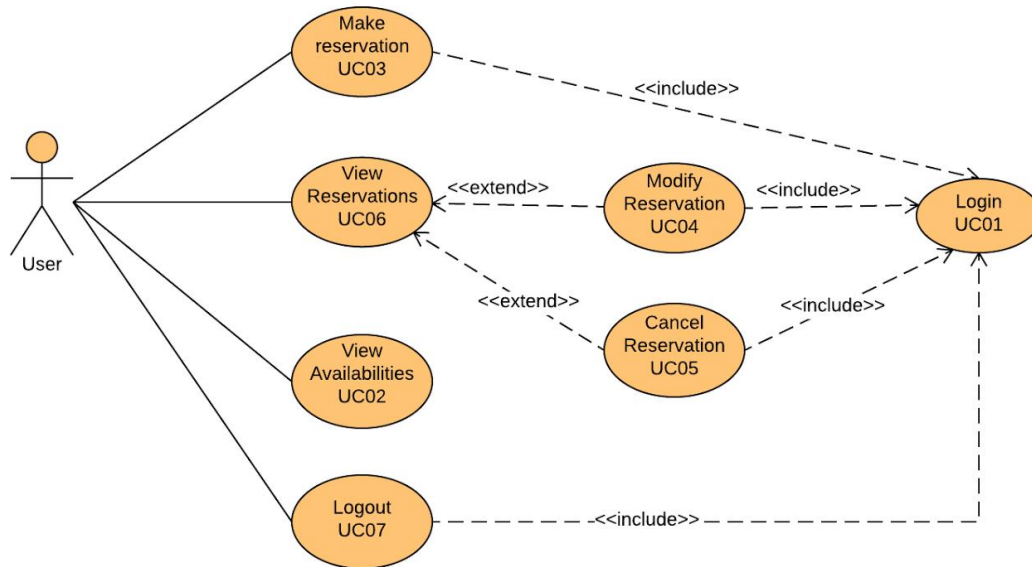
Figure 2 : Use Case Diagram

3.2.3. Use cases

Use Case number	Use case name	Actor	Description
UC01	Login	User	User is authenticated and can access the system, implying he/she is registered with the facility.
UC02	View room availabilities	User	On the index page, user sees facility rooms with their available/taken time slots in a calendar view week by week for the entire academic semester.

UC03	Make reservation	User	User selects an available time slots for room and proceeds to reserve it by entering the required information. Capstone students have priority.
UC04	Modify reservation	User	User chooses to modify the time and day by making an alternative selection from the "view room availabilities calendar". User saves the modification.
UC05	Cancel reservation	User	User cancels one or many of the reservations made.
UC06	View reservations	User	User views all the reservations made with their respective information (location, time, equipment).
UC07	Logout	User	User logs out of the system and is denied permission to make/modify/view or cancel a reservation until a successful login.

Use Case UC01:	Login
Primary Actor:	User
Pre-conditions:	User is registered in the system.
Success guarantee (postconditions):	User is authenticated by the system. User is able to access the main functions of the system.
Main success scenario (basic flow):	1. The user enters their student id and password into their respective text boxes 2. The system validates the user input and redirects them to the welcome page
Extensions (alternative flow):	2.a. If id is not found, indicate error. 2.b. If password entered does not match the password corresponding to the id, indicate error.
Special Requirements:	None
Open issues:	None

Use Case UC02:	View room availabilities
Primary Actor:	User
Pre-conditions:	User is logged in.
Success guarantee (postconditions):	User sees facility rooms with their available/taken time slots in a calendar view week by week for the entire academic semester.
Main success scenario (basic flow):	1. User clicks on "view room availabilities". 2. He can see the rooms in a calendar view week by week for the entire academic semester.

Extensions (alternative flow):	None
Special Requirements:	None
Open issues:	None

Use Case UC04:	Modify reservation
Primary Actor:	User
Pre-conditions:	User is logged in.
Success guarantee (postconditions):	Room reservation is modified.
Main success scenario (basic flow):	1. User selects one of their reservations 2. User selects modify option 3. System locks access to the reservation's room 4. User chooses new times on the calendar view 5. The system updates the reservation with the new information and responds with a confirmation. 6. The system moves any non-conflicting reservations from the waitlist to the reservations list 7. System unlocks access to the reservation's starting room
Extensions (alternative flow):	5.a. Time selected unavailable, displays an error
Special Requirements:	None
Open issues:	None

Use Case UC05:	Cancel Reservation

Primary Actor:	User
Pre-conditions:	User is authenticated and logged into the system. User has previously made a reservation. User has selected an existing reservation.
Success guarantee (postconditions):	User's reservation is cancelled.
Main success scenario (basic flow):	1. User requests to cancel his or her reservation. 2. System displays a confirmation for the cancellation.
Extensions (alternative flow):	Having freed their time slot, if there exists a waitlist for it, the next user in line is placed on the reservation of the newly freed time slot.
Special Requirements:	None
Open issues:	None

Use Case UC06:	View reservations
Primary Actor:	User
Pre-conditions:	User is logged in.
Success guarantee (postconditions):	The user's reservations and waitlist information is displayed
Main success scenario (basic flow):	1. User chooses to view their reservations 2. System displays user's current reservations and waitlist positions
Extensions (alternative flow):	None
Special Requirements:	None
Open issues:	None

Use Case UC07:	Logout
Primary Actor:	User
Pre-conditions:	User is logged in.
Success guarantee (postconditions):	User is logged out.
Main success scenario (basic flow):	1. User chooses clicks on the logout button 2. User is directed to the login page

3.2.4. Critical use cases

Use Case UC03:	Make Reservation
Primary Actor:	User
Pre-conditions:	User is authenticated and logged into the system. User has less than 3 hours of reservations. User has not reserved a room on the same day, same time, same room for the previous 3 weeks
Success guarantee (postconditions):	Reservation is saved.
Main success scenario (basic flow):	1. User picks their desired time and room. 2. User selects equipment. 3. User confirms reservation details. 4. System displays reservation summary.
Extensions (alternative flow):	2.a. If entered information is not sufficient or invalid, the system will prompt to complete it after the user confirms the reservation. 3.a. The room and time chosen is already reserved, the user is put on the timeSlot waiting list. 3.b. The equipment desired is not available, the user is put on the EquipmentWaitlist.

Special Requirements:	None
Open issues:	Students from which classes have higher priority to reserve the facilities? How many days in advance can a reservation be made?

3.3 Non-functional requirements

3.3.1 Reliability

The system shall handle more than one users at once without failure. The database shall keep all relevant information as not to allow user loss of his or her reservations made upon system crash. It shall be continuously tested as to avoid potential crashes and to track occurrences of failures.

3.3.2 Usability

Usability, based on the ISO 25010 standard, has the following characteristics: Appropriateness, Recognisability, Learnability, Operability, User Error, Protection, User Interface, Aesthetics and Accessibility. The system shall allow user goals to be encountered, therefore its accessibility shall be at its best. It is important that the system has clear terminology, intuitive placement of buttons and other features. The User Interface (UI) shall be well responsive, and it shall be easy to pattern complex and simple operations (operability). Since the principal user of the system are Students, they will be encouraged to give feedback on the UX (User Experience) they have received upon experimenting with the system. This feedback will be taken in consideration for the future re-evaluation of the UI, making sure that the usability of the system through users has improved (User Interface). The system shall be intuitive enough so that users will not require the use of an instruction manual. The system shall be easily accessible.

3.3.3 Efficiency

The system shall use minimal resources (such as memory) and store in the database only relevant data in an organized manner. The system shall also be efficient for the user in that the

interface shall provide the user to get information quickly without having to perform numerous clicks.

3.3.4   Maintainability

The system should be extendable, in that new functions can be further implemented into the system. Following the ISO 25010 standard, maintainability has multiple quality characteristics that can be monitored throughout the milestones of project using a tool that provides the capability of measuring those quality characteristics. The software *Logiscope* is indeed capable of measuring analyzability, changeability, stability and testability for software written in C#.

3.3.5   Portability

The system should be portable with devices that can support a web browser such as Google Chrome, Internet Explorer or Firefox. It should function on most operating systems, such as Windows, UNIX or LINUX platforms, such that users can access the web application on most computers.

3.4   Design constraints

Languages and tools

The conference room reservation system is developed in C# with the ASP.NET framework using Microsoft Visual Studio Community Edition as the integrated development environment. All artifacts produced during development are found on the password-protected GitHub repository. Those include the Software Requirements Specification (SRS), the Software Architecture Document (SAD), the code, and the test cases. The link to the repository is the following:

https://github.com/alphakennyn/FuriousMonkey_CapstoneRoom_344

Processes

The development life cycle followed for the development of the online conference room

reservation system is an iterative process in which each iteration follows the waterfall development life cycle. More precisely, the requirements are reviewed at the beginning of each iteration. The design and architecture is adjusted if necessary. The code is updated in order to correspond to the design changes, and test cases are created to cover the new requirements.

4. Analysis Models

4.1. Domain Model Diagram

The Domain Model diagram consists of a total of ten domain level objects (DLO). This diagram shows the relationship between conceptual classes for a reservation system.
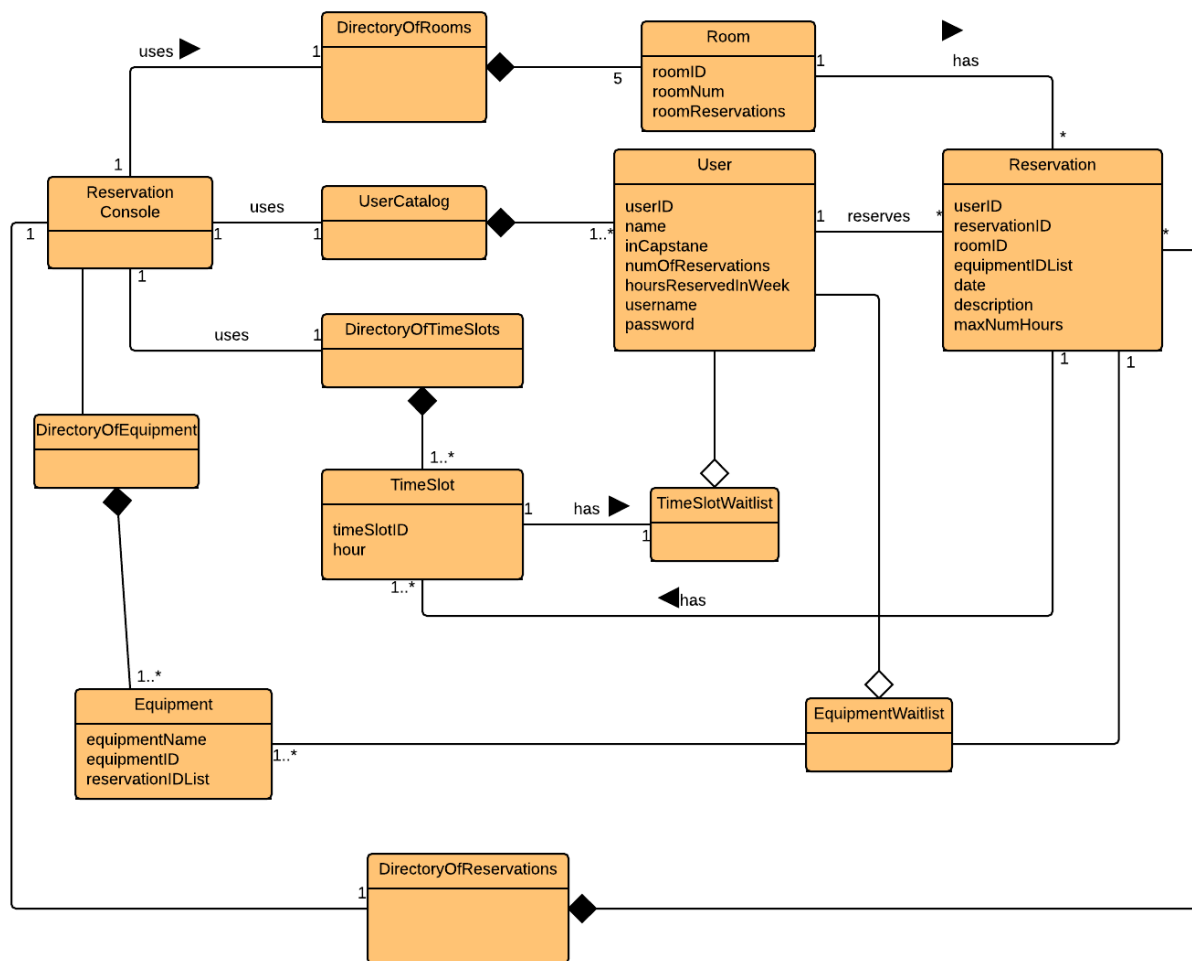

Figure 3 : Domain Model

4.2. State diagram

In this section, the state diagram represents the critical use case Make reservation.

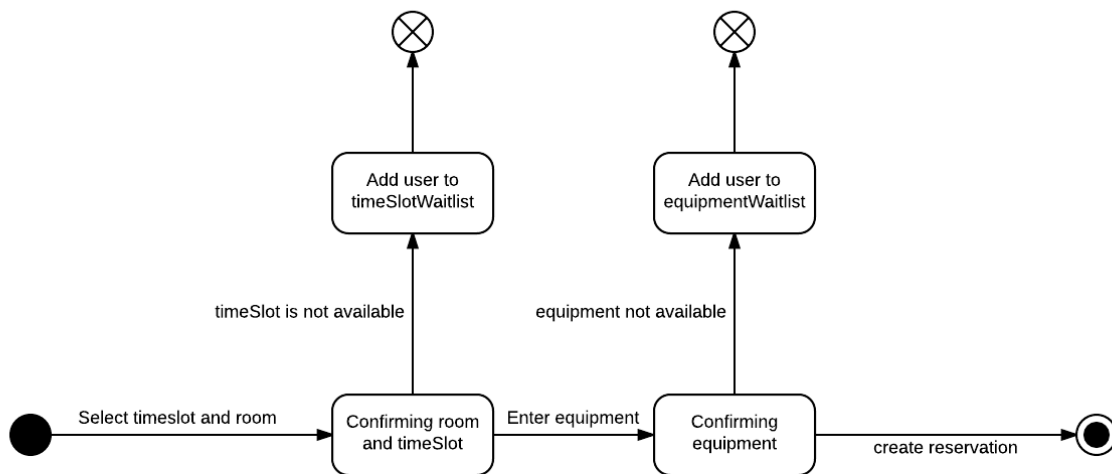### 4.2.1. State diagram for Make reservation

Figure 4: State Diagram for Make Reservation

4.3. System Sequence Diagrams

In this section, system sequence diagrams (SSD) represent the critical use case Make reservation. A system operation is derived and is followed by an operation contract.

4.3.1. *System Sequence Diagram for Make reservation*

Make reservation main (success) scenario: A user accesses an online console for a conference room reservation system to reserve a room for a given timeslot (maximum 3 hours per week). The user enters all required information and submits it. The system displays a confirmation and

a reservation number after completion of the reservation. The user may only reserve a room if they reserved for less than 3 hours during the week. Pre-conditions:

1. User has been authenticated from a database (user catalog).
2. Equipment is available for reservation on a given time slot.
3. Room is available on the same time slot.

Post-conditions:
1. One (or more) reservation has been made on a given user.
2. Equipment is not available for the given time slot.
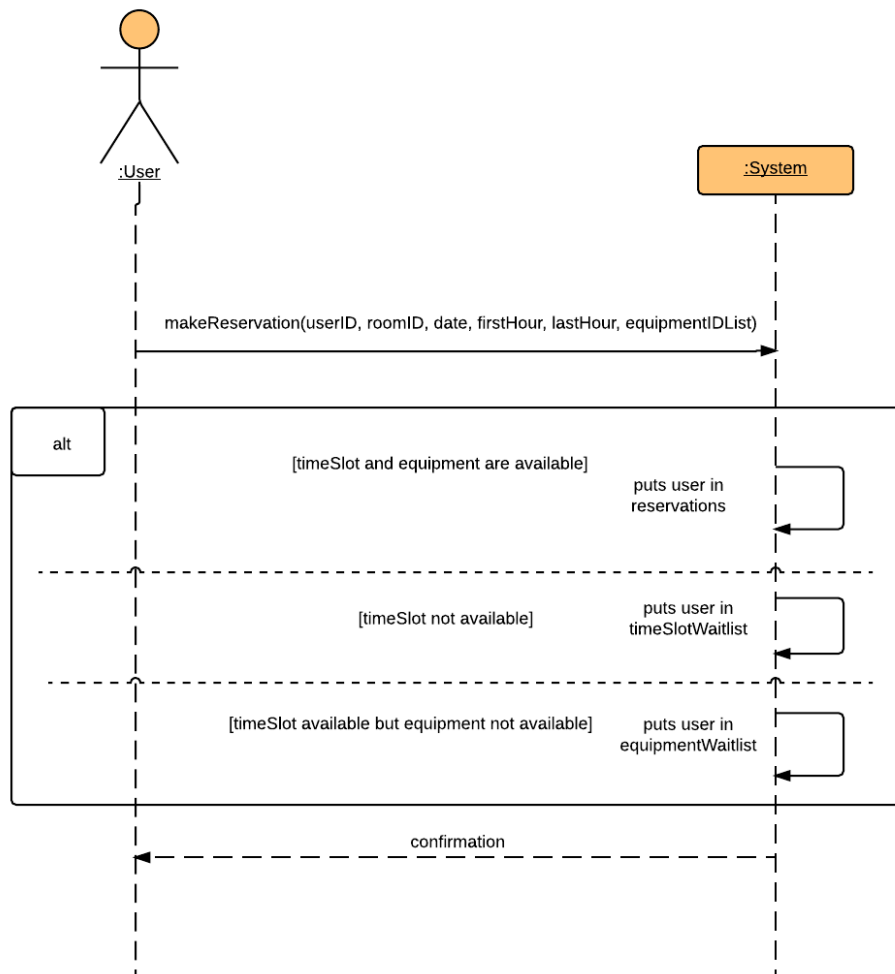3. Room is not available for the given time slot.

Figure 5 : System Sequence Diagram for Make reservation

System operation for Make reservation

System
makeReservation(userID, roomID, date, firstHour, lastHour, equipmentNameList)

Operation contract

Contract CO1: **makeReservation**
Operation: *makeReservation(userID, roomID, date, firstHour, lastHour, equipmentNameList)*
Cross References: Use Case: Make reservation
Pre-Conditions: 1. Room is available on a given timeslot. 2. Equipment is available on the same timeslot. 3. User is logged into the system.
Post-Conditions: 1. An instance of Reservation res was created. (instance creation) 2. Instance res was associated with DirectoryOfReservations (formation of an association) 3. Instance res was associated with DirectoryOfEquipments (formation of an association) 4. Instance res was associated with Room. (formation of an association) 5. Instance res was associated with User. (formation of an association) 6. Instance res was associated with Equipment (formation of an association) 7. An instance of TimeSlot ts was created. (instance creation) 8. Instance ts was associated with Reservation. (formation of an association)

5. Testing

5.1. Unit Testing

Unit Testing Tools

Unit testing will be done using MSTest framework. Which is the default unit test plugin for Visual Studio. We chose to use this because it's simple to use and our tests were not heavy.

Relevant Units to Be Tested

We ran a few tests for the login to see if the credentials would be accepted by the system when a user tries to login. All login tests ran at the same time. We had 2 test cases both consisting of inputs for the method.

We also ran a few unit tests for the constraints when a user tries to reserve a room. The constraint was that a user is not able to reserve a room for more than 3 hours. The first test was written in a way that it would succeed, and it did. The second test was purposely written to fail, and it did.

Unit Test	Result
AccountControllerTest.LoginValidTest()	Pass
AccountControllerTest.LoginTest()	Pass
Summary	2/2 tests passed

Unit Test	Result
ReservationConsoleTest.dailyContraintCheckFailTest()	Pass
ReservationConsoleTest.dailyContraintCheckSucceedTest()	Pass
Summary	2/2 tests passed

Run All | Run... ▼ | Playlist : All Tests ▼

◢ **AccountControllerTests** (2)
  ✓ LoginIsValidTest   8 sec
  ✓ LoginTest   4 ms
◢ **ReservationConsoleTests** (2)
  ✓ dailyConstraintCheckFailTest   < 1 ms
  ✓ dailyConstraintCheckSucceedTest   2 ms

Summary

Last Test Run Passed (Total Run Time 0:00:12)
  ✓ 4 Tests Passed