

# Software Architecture Document

Version 1.0

for

Concordia Capstone Scheduler

Prepared by

Name	StudentID	Email
Lance Lafontaine	26349188	lance.lafontaine92@gmail.com
Jason Tsalikis	25892120	jtsalikis@hotmail.ca
Lenz Petion	26775837	lenzpetion@gmail.com
Rameen Rastan-Vadiveloo	27191863	rameenrastanv@hotmail.com
Benny Zhao	27205104	bennyzhao@live.ca
Simeon Cvetkovic	27430515	bitugos@gmail.com
Lance Lafontaine	26349188	lance.lafontaine92@gmail.com
Sam Alexander Moosavi	27185731	sammoosavi94@gmail.com

Instructor	Professor Constantinos Constantinides
Course	SOEN 343 Software Architecture and Design I
Date:	November, 23, 2016

## Document History

Date	Version	Description	Authors
Saturday October 8th 2016	0.1	Initial template and structure of document	All
Saturday October 23rd	0.2	Define architectural overview, class diagram	- Simeon - Lance - Rameen - Jason
Friday, November, 18th	0.3	Writing body text	- Rameen - Sam - Jason - Zhipeng
Sunday, November, 20th	0.4	Developed layered class diagram/design package diagram, sequence diagrams, writing body text	- Rameen - Sam - Jason - Lenz
Tuesday, November, 22nd	0.5	Finishing body text and diagrams	- Rameen - Sam - Jason - Lance
Wednesday, November, 23rd	1.0	Formatting and editing, refined diagrams	- Rameen - Sam - Jason - Simeon - Benny

## **Table of contents**

### **Introduction 4**

Purpose 4

Scope 4

Definitions, acronyms, and abbreviations 4

Glossary 5

### **Architectural representation 6**

### **Architectural requirements: goals and constraints 7**

Functional requirements (Use case view) 8

Non-functional requirements 9

### **Logical view 11**

Layers, tiers etc. 12

Subsystems 12

Use case realizations 14

Reuse of components and frameworks 15

### **Size and performance 16**

## List of figures

Figure 1	Class Diagram	6
Figure 2	Use Case Diagram	7
Figure 3	Layered Class Diagram	10
Figure 4	Design Package Class Diagram	13
Figure 5	View Reservation Sequence Diagram	14
Figure 6	Make Reservation Diagram	14
Figure 7	Modify Reservation Sequence Diagram	15
Figure 8	Cancel Reservation Sequence Diagram	15

## List Of Tables

Table 1	Use Case View	8
Table 2	NFR table	10

## **1. Introduction**

### **Purpose**

The purpose of the Software Architecture Document (SAD) is to describe the system in terms of its high-level design (the system's architecture), as well as its low-level design (implementation level). Firstly, the document will provide an overview on the architecture of the system. It will proceed to present architectural representations of the system through different architectural views. It will then make reference to architecturally relevant functional and nonfunctional requirements, and will finally address size, performance, and quality attributes of the system. The document is intended for all team members as well as the clients of the product, as the different architectural views provide different abstractions of the system that is intended for different stakeholders.

### **Scope**

This document applies strictly to the software architecture of the system. As such, it details the high-level and low-level design of the system. This has a direct effect on the implementation of the system, as the implementation will be modeled by the system's architecture. Designers and implementers of the system will make reference to the architectural overview presented in this document when making design decisions during the development of this product.

## Definitions, acronyms, and abbreviations

Table 1 . Glossary

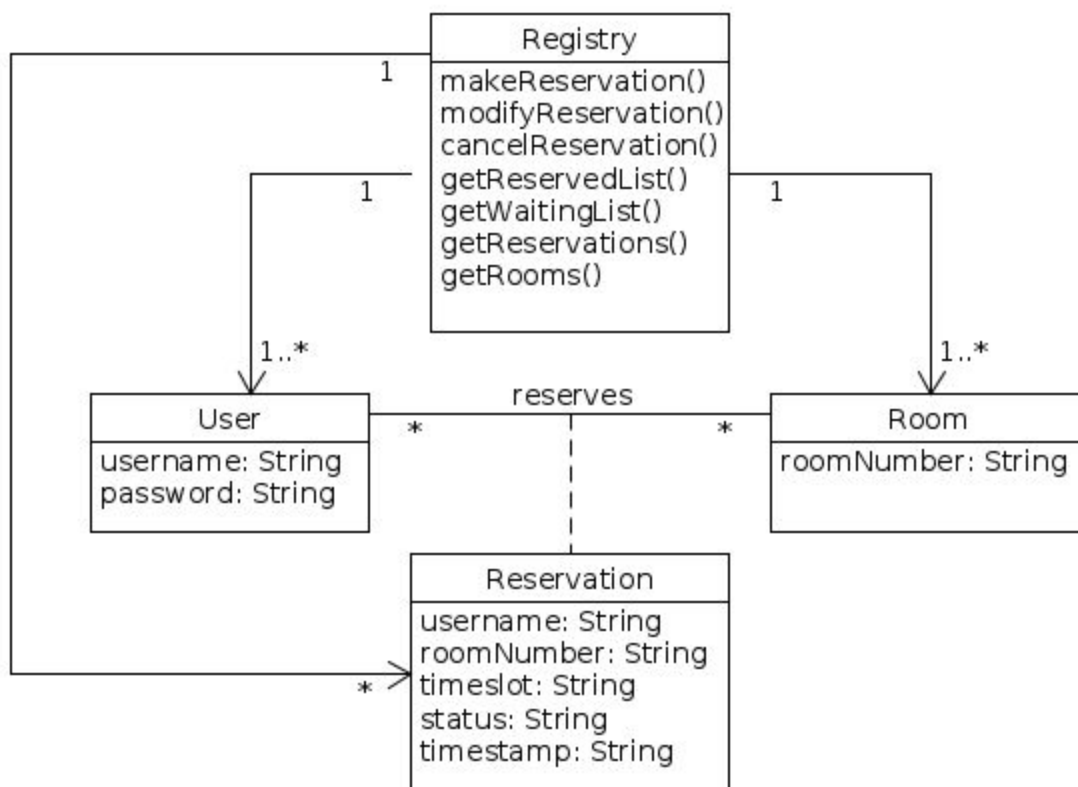
Term	Definition
User	Someone who interacts with the reservation system.
Registry	A system for keeping an official record of room reservations.
Room	A space that students occupy for academic activities
Wait list	An ordered queue of students with a pending reservation at an unavailable time slot.
Reservation	Occupying a study room at a given time slot
Time Slot	A 1-hour time interval.
Weekly Calendar	A table displaying room information (time slots and availabilities) for a given week.
Stakeholder	An individual or group who is affected by the development of this project.
SRS	Software Requirements Specification
SAD	Software Architecture Documentation
Python	Python is a high-level, general-purpose programming language
SSD	System Sequence Diagram
HTML	Hypertext Markup Language
CSS	Cascading Style Sheet
TDG	Table Data Gateway

## 2. Architectural representation

### Logical view

#### Class Diagram

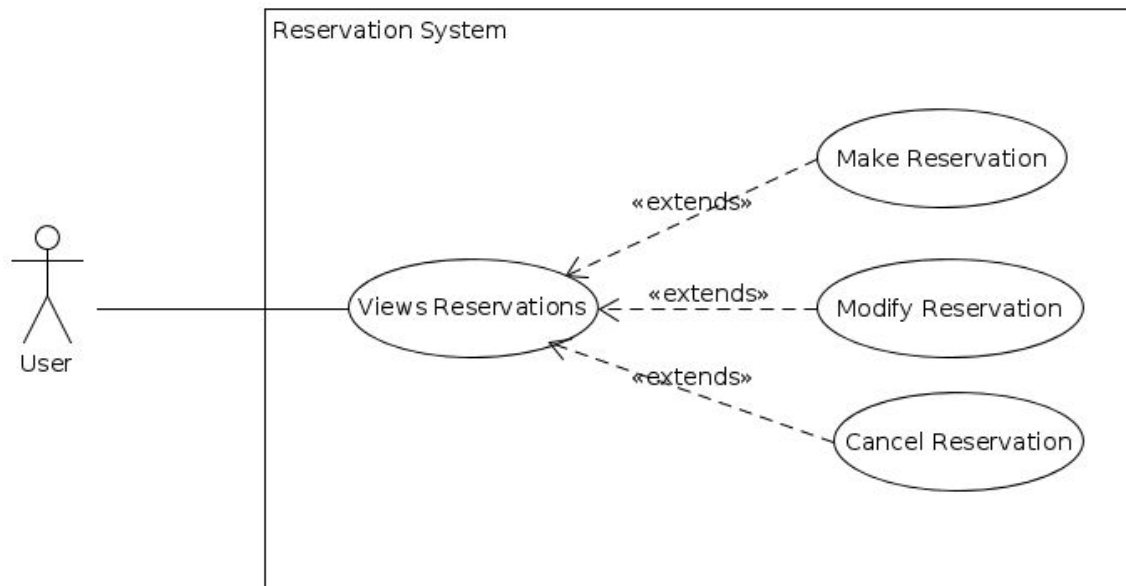
Figure 1. Class Diagram



## 1. Use case view :

### Use Case Diagram

Figure 2. Use Case Diagram



## 2. Architectural requirements: goals and constraints

### Functional requirements (Use case view)

The overview below refers to architecturally relevant Use Cases from the Use Case Model (see references).



Table 1. Use Case View

Source	Name	Architectural relevance	Addressed in:
UC1	Make Reservation	Requires us to interactive with database and verify other reservations; this was the bulk of business logic complexity, designed around this use case.	Sequence Diagram 2
UC2	View Reservation	Needs to be updated often in the front end; lead us to implement some form of an observer	Sequence Diagram 1
UC3	Modify Reservation	Given that we may be modifying often we implemented a unit of work to reduce load on the database.	Sequence Diagram 3
UC4	Cancel Reservation	Given that we may be modifying and cancelling often we implemented a unit of work to reduce load on the database.	Sequence Diagram 4

## **Non-functional requirements**

### **Accessibility**

Only ENCS registered students in the database should have access to the web application.

### **Usability**

90% of users will be able to perform all major activities provided by the system in less than 3 minutes.

### **Portability**

The web application should function on all major operating systems.

Table 2. NFR Table

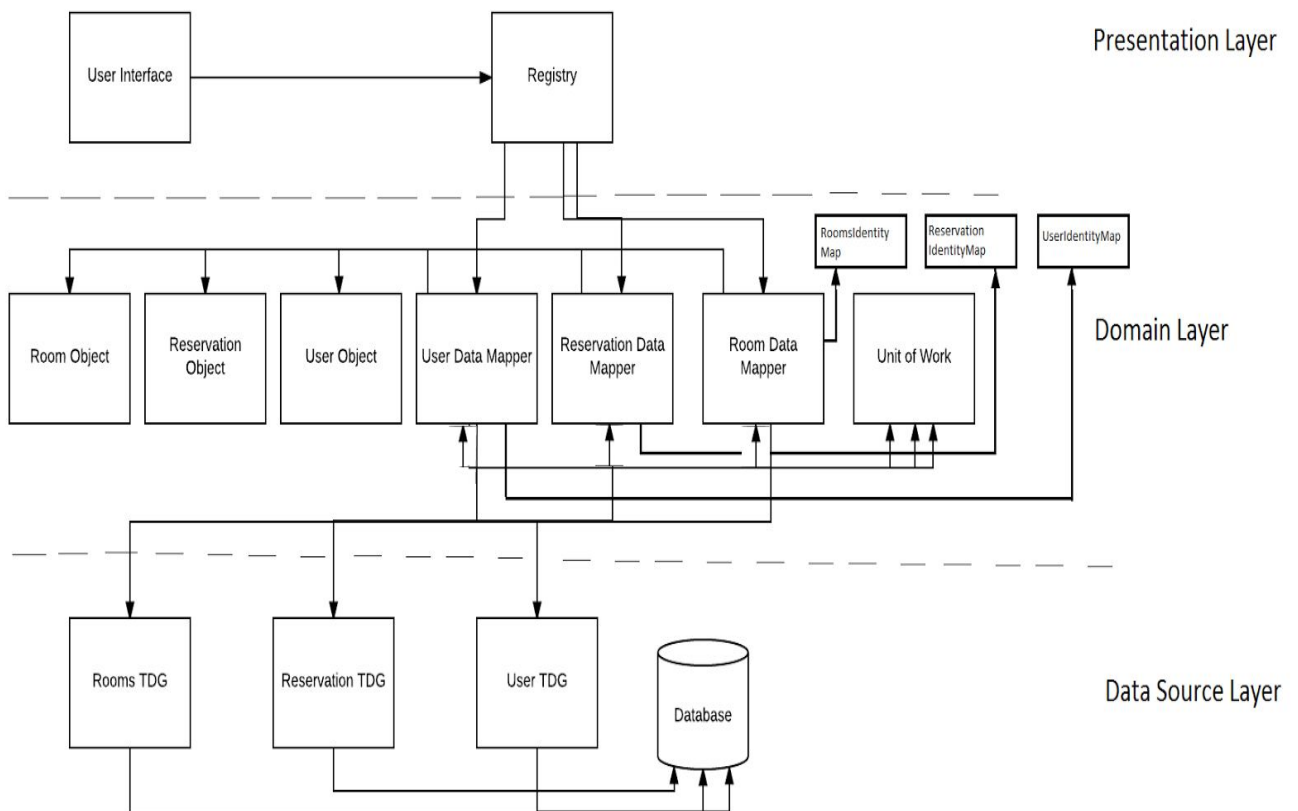
Source	Name	Architectural relevance	Addressed in:
SRS	Accessibility	Login and logout functionality has to access database of registered students	Logical View
SRS	Usability	A Calendar UI that marks which timeslots were taken, this visual value makes the system easier to use	Logical View
SRS	Portability	Used standard libraries, web standards and popular frameworks to ensure compatibility on all major operating systems.	Physical View

### 3. Logical view

#### Layers, tiers etc.

#### Layered Architecture Class Diagram

Figure 3. Layered Class Diagram

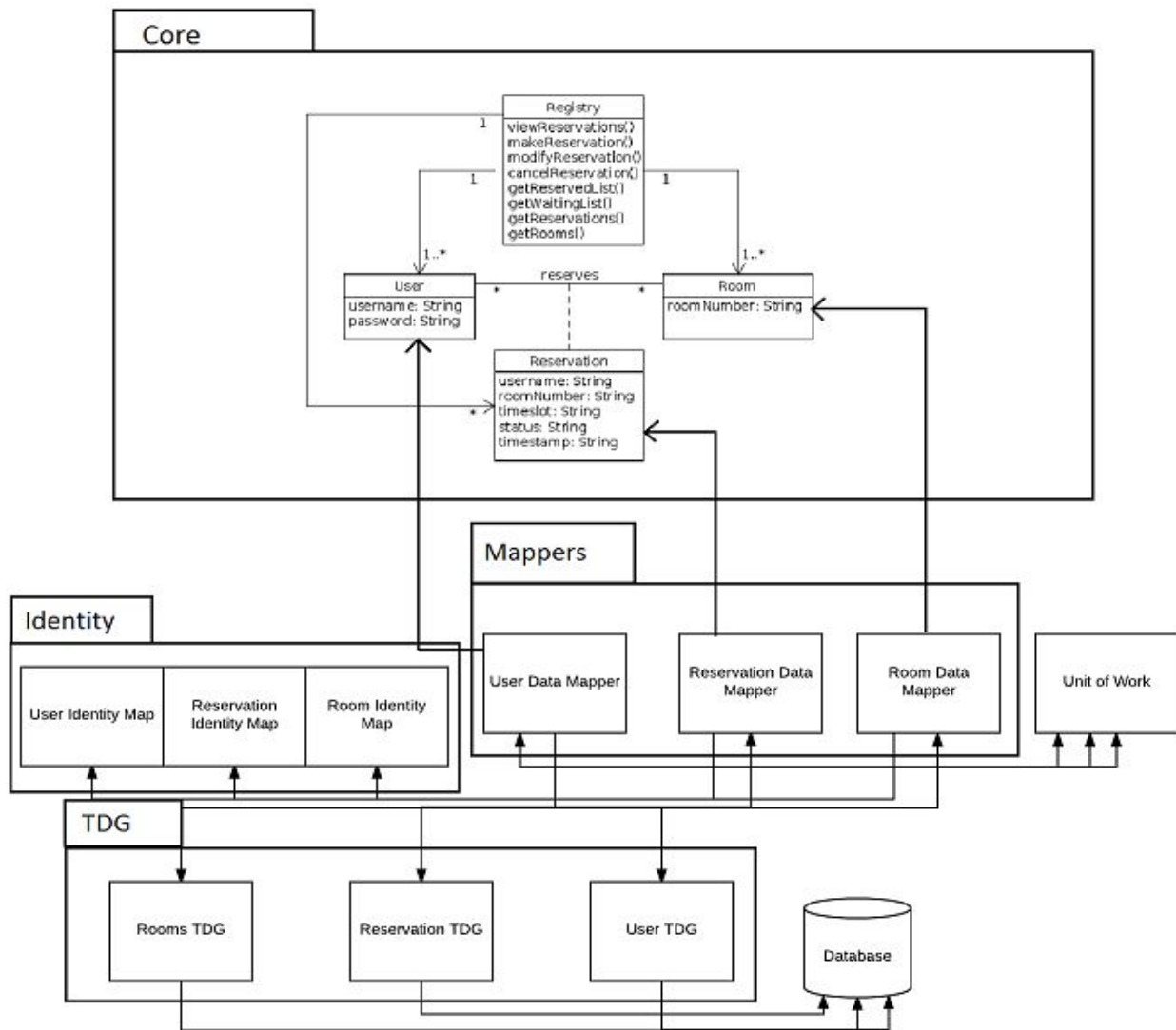


## **Subsystems**

The web application can be divided into two discrete subsystems: the client and the server. The client handles the presentation layer of the system, which is the only portion of the overall system that is visible to the user. This includes the Graphical User Interface, and also handles all user actions and requests. If the client receives a request from a user, it is then processed and sent to the next subsystem: the server, which contains the domain and data source layer of the system. The server handles the core business logic of the application and also contains the relational database, which contains tables to store domain object information in order to develop persistence in the overall system. These two subsystems interact with each other (the client sends a request to the server upon user input, the server then processes the action and sends a response), to form the overall system.

## Architecturally significant design packages

Figure 4. Design Package Class Diagram



## Use case realizations

### Interaction Diagrams (Sequence Diagrams)

#### View Reservations Sequence Diagram

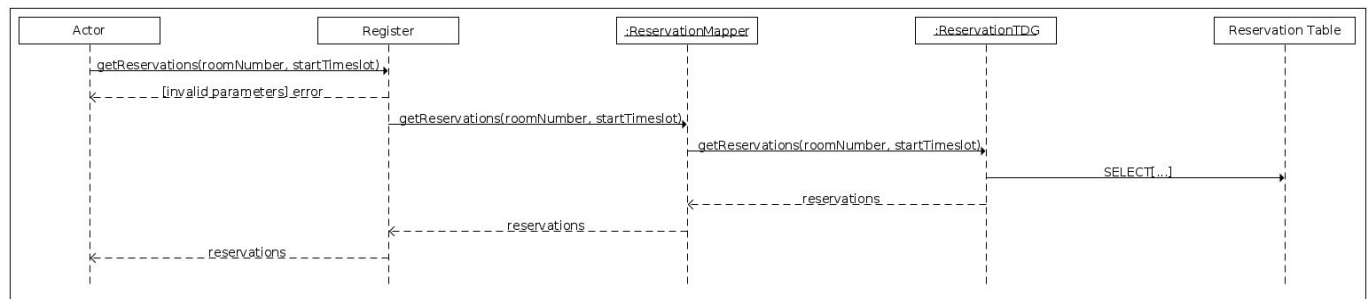


Figure 5. View Reservation Sequence Diagram

\*\*\* For a better view, see *getReservations\_SD.pdf* in the *SequenceDiagrams\_PDF* folder or <http://i.imgur.com/Hr2QqO0.png>

#### Make Reservation Sequence Diagram

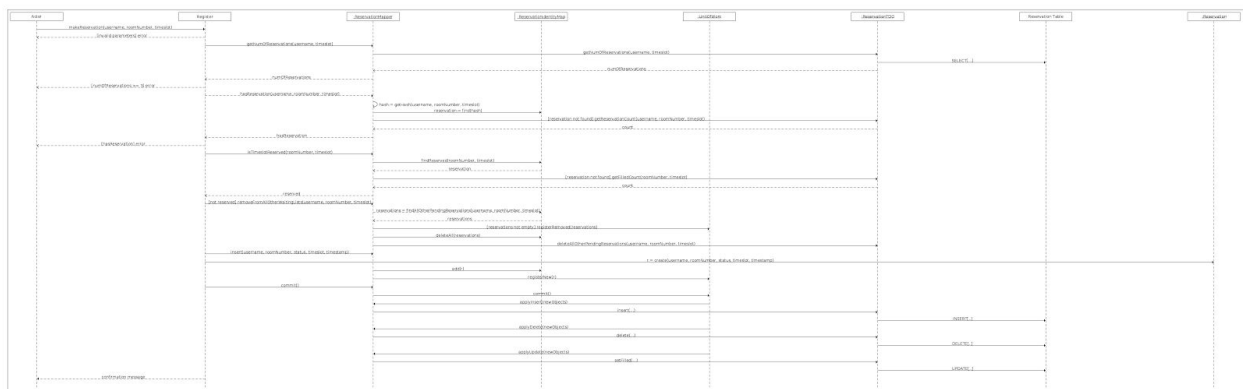


Figure 6. Make Reservation Diagram

\*\*\* For a better view, download *makeReservation\_SD.pdf* in the *SequenceDiagrams\_PDF* folder or <http://i.imgur.com/nLpqaN1.png>

## Modify Reservation Sequence Diagram

### Figure 7. Modify Reservation Diagram

\*\*\* For a better view, download *modifyReservation\_SD.pdf* in the *SequenceDiagrams\_PDF* folder or <http://i.imgur.com/U3pIJbi.png>

### Cancel Reservation Sequence Diagram

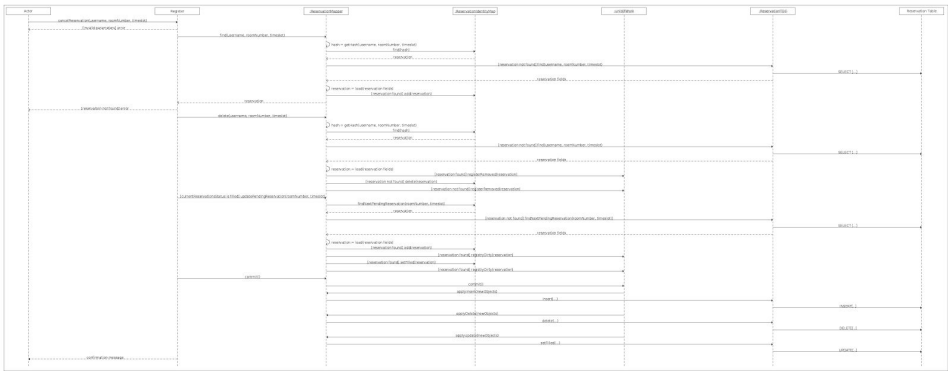


Figure 8. Cancel Reservation Sequence Diagram

\*\*\* For a better view, download *cancelReservation\_SD.pdf* in the *SequenceDiagrams\_PDF* folder or <http://i.imgur.com/8EsTfF6.png>



### **Reuse of components and frameworks**

The web framework which was used in the implementation of this application was Django. Django is a python based web framework. Due to requirements in the project outline, we were restricted from incorporating certain functionalities of the framework, as certain limitations were imposed. In addition to this, Node.js, a run-time environment for JavaScript, was used to assist with front-end development.

### **4. Size and performance**

Volumes:

- Estimated ENCS Capstone students : 400 students
- Estimated Number of Capstone rooms: 15
- => Total number of Capstone room timeslots: 60
- Capacity for Number of Capstone rooms: Limited only by disk space and database table lengths

Performance:

- Time for a user to login: less than 5 seconds
- Time for the system to perform operations on the schedule (add, cancel, modify): less than 8 seconds