Software Design Document

for

Optimized Course Schedule Generator

Version 1.0 approved

Prepared by

Brett M. Stricker
Doyle D. Bigelow
Diego R. Draguicevich

OCSG

April 16, 2021

Abstract

The Optimized Course Schedule Generator (OCSG) is a tool that will assist educational institutions with the creation of their school's master schedule. The system will be designed utilizing the client-server architecture and will have five subsystems for that architecture design, the UI Subsystem, the User Management Subsystem, the Schedule Generator Subsystem, the Authentication Subsystem, and the Database Management Subsystem. Each subsystem will be explained in detail as to their functionality and purpose to the system. These subsystems will be constructed of a number of classes that will be used to instantiate objects that will perform work that will perform use cases listed in this document. Each of these classes and their purpose to the functionality of the system will be explained in detail throughout this document. This paper will also contain multiple diagrams associated with the design of classes, objects, architecture, packages and the sequences to take place in the system. Upon reading this document the reader should gain an in-depth understanding as to how the system works and the purpose each element of the system serves.

Table of Contents

Abstract **1**

Table of Contents **2**

Revision History **2**

1. Introduction **3**
1.1 Purpose 3
1.2 Definitions, Acronyms, and Abbreviations 3
1.3 Overview of Document 3
1.4 References 3

2. Proposed Software Architecture **3**
2.1 Overview 3
2.2 Subsystem Decomposition 4
2.3 Persistent Data Management 5

3. Class & Object Design **5**
3.1 Overview 5
3.2 Detailed Class Design 6
3.3 Object Interaction 7

4. Appendix **8**
4.1 Appendix A - Use Case Diagrams 8
4.2 Appendix B - Use Case Tables 11
4.3 Appendix C - Class Diagrams 19
4.4 Appendix D - Sequence Diagram 24

Revision History

Name	Date	Reason For Changes	Version
First Draft	2021-04-10	Initial Draft	1.0
First Release	2021-04-17	Finalized	1.1

1 Introduction

1.1 Purpose

The Optimized Course Schedule Generator Version 1.0 is a software tool to assist universities in creating the master schedule for the semester. Schools have to manage thousands of students, and hundreds of teachers and classrooms. The software will generate a master schedule requiring minimal edits decreasing the labor required by a university.

1.2 Definitions, Acronyms, and Abbreviations

WIP,

OCSG: Optimized Course Schedule Generator

SRS: Software Requirements Specification

UML: Unified Modeling Language

SDD: Software Design Document

(s): Singleton (When used in a Class Diagram)

1.3 Overview of Document

This document contains information about the architecture that will be used for the development of the OCSG project and class and object designs of the systems and subsystems of the project. The reader will be able to find a multitude of diagrams that display how the system will operate and how elements of the system will interact with each other. These diagrams include: a Package Diagram, a Class Diagram, a Use Case Diagram, and a Sequence Diagram. Readers can expect to find detailed explanations as to how each subsystem will function, the data that will be stored and security requirements needed to store the data. Information about how the use cases defined in the SRS will be used in the system will be found in the sequence diagram.

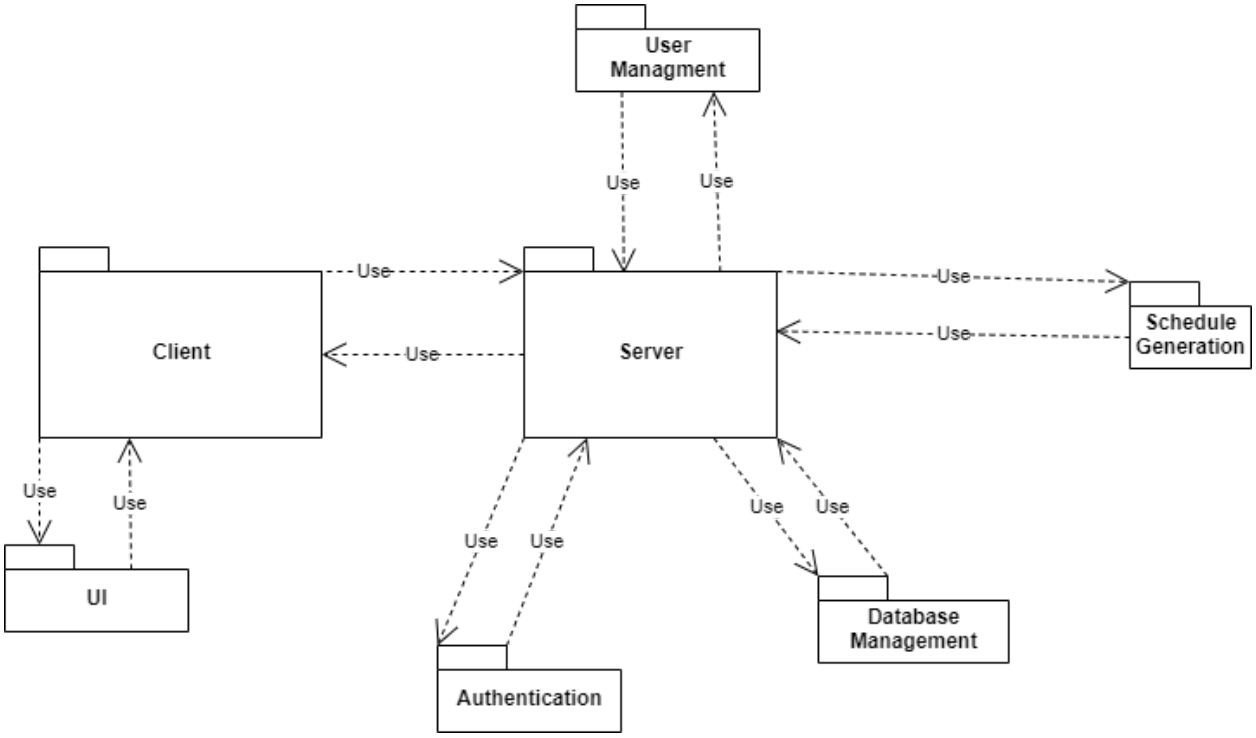
1.4 References

2 Proposed Software Architecture

2.1 Overview

The system will be constructed based on a client-server architecture. Both the Client and the Server subsystems will be made up of their own subsystems. The Client System will be composed of a UI Subsystem. The Server subsystem will be composed of a User Management Subsystem, a Schedule Generator subsystem, an Authentication subsystem, a Database Management Subsystem. The UI Subsystem will handle all of the user interface elements that will be found on the clients device when they are interacting with the system. The User Management Subsystem will handle all of the user permission management for the database that is being referenced. The Schedule Generator Subsystem will be utilized to generate a schedule from the available information on the database. The Database Management Subsystem will be used to edit data that is stored in the database. Finally, the Authentication Subsystem will handle all access to the system.

2.1.1 Package Diagram



2.2 Subsystem Decomposition

The UI Subsystem will be where the user will interact with the system. This will include the user viewing the data from the database, sending requests to edit data and requests to access the system. The UI Subsystem will be related to all the use cases as the UI is the method by which users interface with the system as a whole. The User Management Subsystem retrieves requests to edit user data for the client and update that data in the database. This subsystem is associated with the User Management Use Case. The Schedule Generation Subsystem will be used to create a master schedule that will contain all needed information about classes and the locations, timings, capacity and teachers. This subsystem will be related to the Generate Use Case. The Database Management Subsystem retrieves requests to edit data for the client and update that data in the database. This subsystem is associated with the Database Management Use Case. The Authentication Subsystem will control access to the system via logging in and out of the system and authenticating the user and determining what access they should have in the system. This subsystem is associated with the Authentication Use Case.

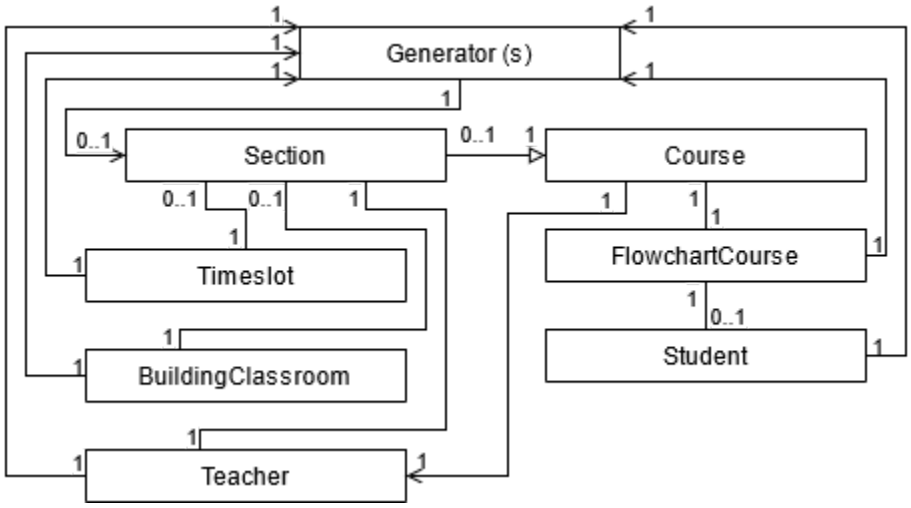
2.3 Persistent Data Management

The system will store information on time slots, buildings and other classroom locations, course subject and number, teacher, and section availability for the Generate Master Schedule function. The system will also store authentication information in the form of usernames and passwords for the Authentication Subsystem. All information must be encrypted with RSA using 2048 bit key length, no information may be stored in plain text.

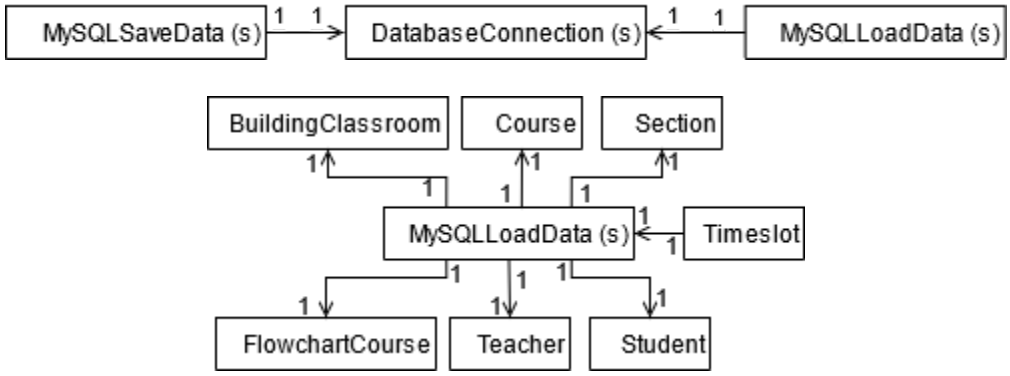
3 Class & Object Design

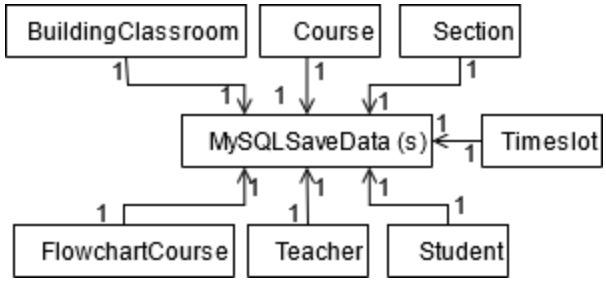
3.1 Overview

Note that any objects are assumed multitons unless otherwise specified.



The Generator Class is instantiated to generate a Master Schedule, and otherwise is not instantiated at all. A Master Schedule is made up of many Section Objects, the creation of which is the only purpose of the Generator Class. To maximise the efficiency of the Generator Class, all other classes except for Section are used as inputs for the Generate Master Schedule use case. Among the other classes, the Section Objects are used for actual, enrollable sections of a Course, which serves as an abstract template much like a course in a course catalog. Each Section Object inherits from a Course, and is associated with a Timeslot, BuildingClassroom, and Teacher. Timeslot, BuildingClassroom, and Teacher Objects may have any number of associated Section Objects. Instances of the FlowchartCourse Object are used to determine which courses certain majors should take in certain semesters, and is thus associated with any number of Student Objects which represent students who are expected to take certain courses as connected to their majors.





For data handling, the MySQLSaveData and MySQLLoadData Classes pass data to be stored or requests to be loaded to the DatabaseConnection Class which in turn communicates with the Server Subsystem. Both classes either receive information from the various data type classes or instantiate them.

3.2 Detailed Class Design

The first section of classes are the ones relating to generating the Master Schedule, and serve as objects to store database data inside of. The classes can then be used to generate a schedule based on the data inside or can be modified for storage back in the database. See: 4.3.5 “Generate” Class Diagram.

The Course Class is the catalog listing of a course tracking the id, name, and credit hours of a course.

The Sections Class inherits the Course Class, but serves as the class meeting. It stores the same data as Course Class, but also tracks the teacher, location, and the meeting time of the class.

The Timeslot class is the different time sections that classes can meet; including days of meeting, start time, end time, and how preferred a specified time slot is.

The BuildingClassroom class is the location(s) where classes can be held. This includes the id of the room and its max capacity.

The Teacher class is teachers which are available to teach courses. This includes their id, a list of courses they can teach, and their current number of courses being taught.

The FlowchartCourse class is the representation of a degree flowchart. This class stores a course id, the code for a major, and which semester the course will be scheduled for.

The Student class stores data on students relevant to the system: their id, major, and semester they are going to be in.

The Generate class is a singleton that takes all previous classes, except for Section, and generates a Master Schedule, based on the data in those classes.

The MySQLLoadData is a singleton class meant to load data from the database into their respective classes. See: 4.3.6 “Database Connector” and 4.3.7 “Load Data” Class Diagrams.

The MySQLSaveData is a singleton class meant to save data to the database from their respective classes. See: 4.3.6 “Database Connector” and 4.3.8 “Save Data” Class Diagrams.

The DatabaseConnection class is a singleton that stores the data relevant to connecting the code to the database: host, user, password. See: 4.3.6 “Database Connector” Class Diagram.

3.3 Object Interaction

The object interaction for this system can be seen in the Sequence Diagrams found in appendix D. These diagrams explain the sequence of events that will occur with each use case and are associated with specific use case diagrams.

Diagram 4.4.1 is associated with diagram 4.1.1 and shows the sequence of actions that occur when a user wishes to authenticate their login credentials. After the user boots up the system they will be met with a screen to enter their login credentials. After the credentials are entered the user will request that the server authenticate the credentials. If the server fails to authenticate the server will send a message informing the user of an error and return them to the login page. If the server authenticates the credentials the user will be granted access to the system and the user will be forwarded to the View data page.

Diagram 4.4.2 is associated with diagram 4.1.3 and displays the events of the User Management use case. After the user logs into the system they are sent to the view data page. The user then has three options if they have access to User Management, Add User, Remove User, and Modify User. When the user selects to add a user they will send a request to the server and then be returned to the view data page with a notification as to the status of the request. Remove and modify user requests will function similarly.

Diagram 4.4.3 is associated with diagram 4.1.4 and displays the events of the View Data use case. Once the user gains access to the system they will be brought to the View Data page. The user can then send a request to the server to view data. The server will then return the data the user wishes to view and display it to them.

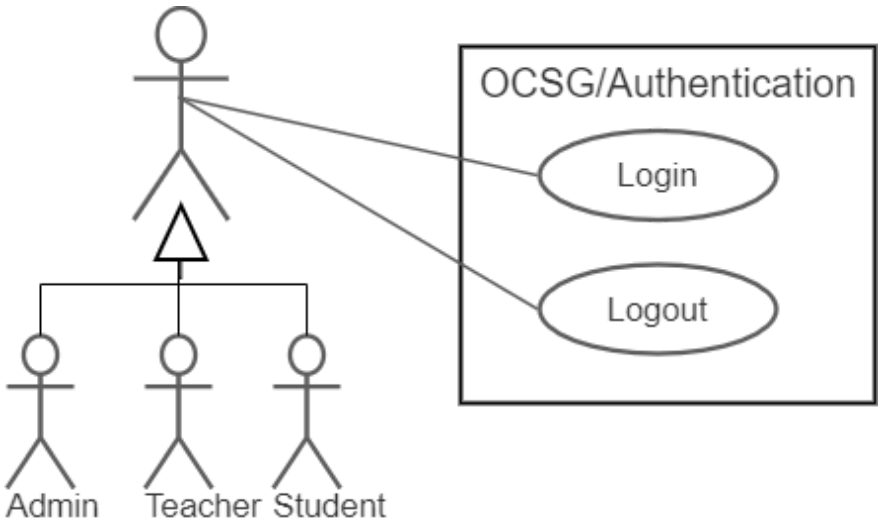
Diagram 4.4.4 is associated with diagram 4.1.2 and displays the events of the Generate use case. The user will request the server to generate a schedule. The server will then display the schedule that was generated to the user in the view data tab.

Diagram 4.4.5 is associated with diagram 4.1.5 and displays the events of the Data Management use case. After the user logs into the system they are sent to the view data page. The user then has three options if they have access to Data Management, Add Data, Remove Data, and Modify Data. When the user selects to add Data they will send a request to the server and then be returned to the view data page with a notification as to the status of the request. Remove and modify data requests will function similarly.

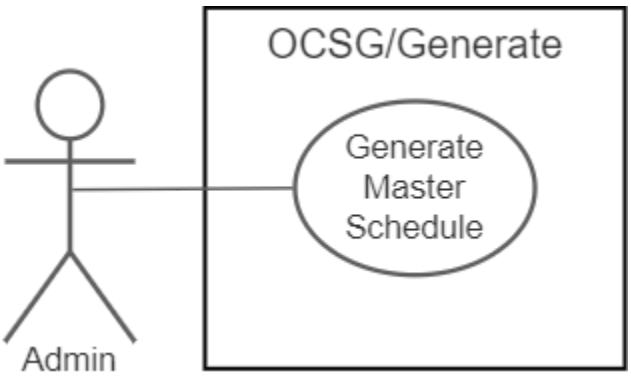
4 Appendix

4.1 Appendix A - Use Case Diagrams

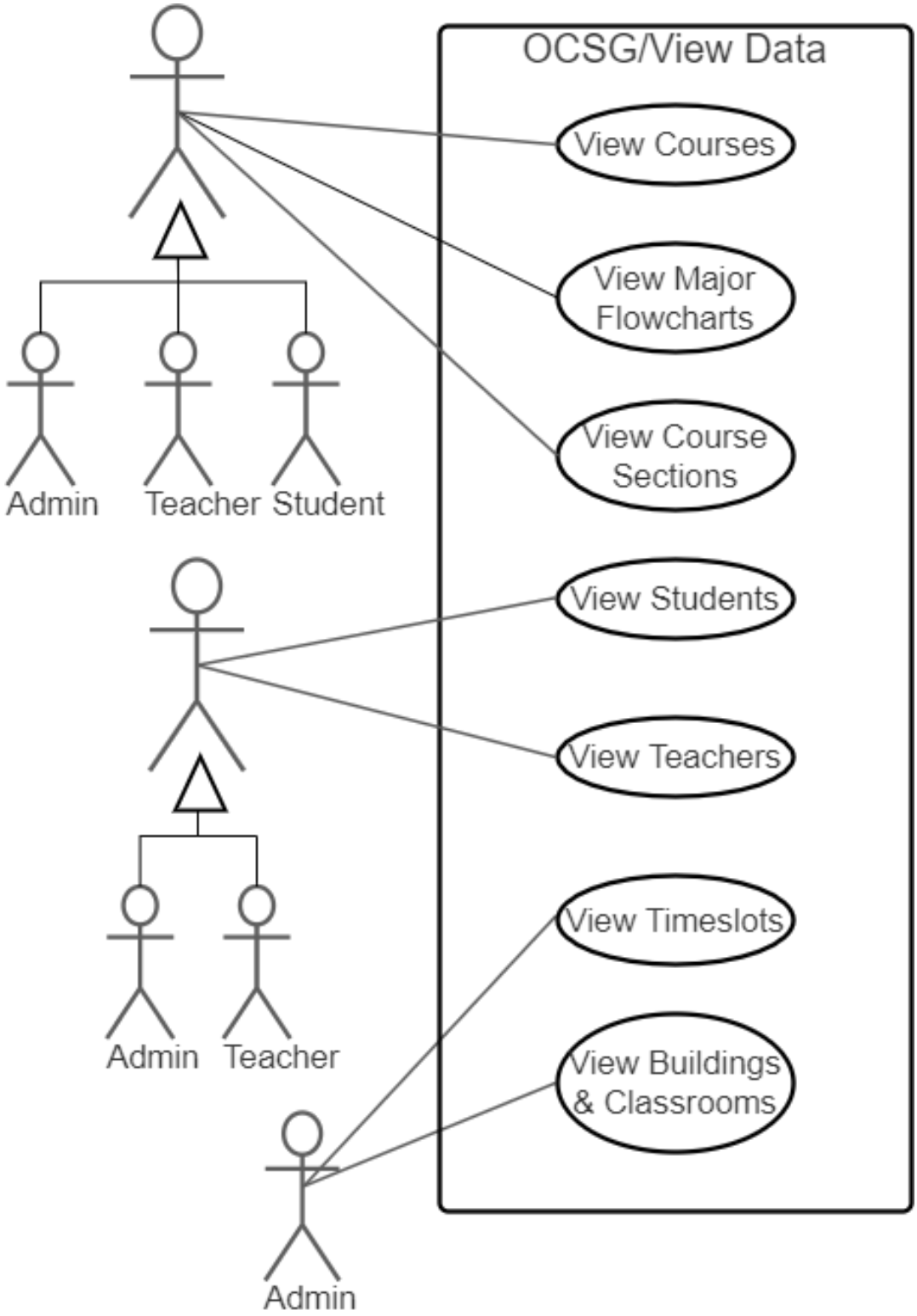
4.1.1 Authentication



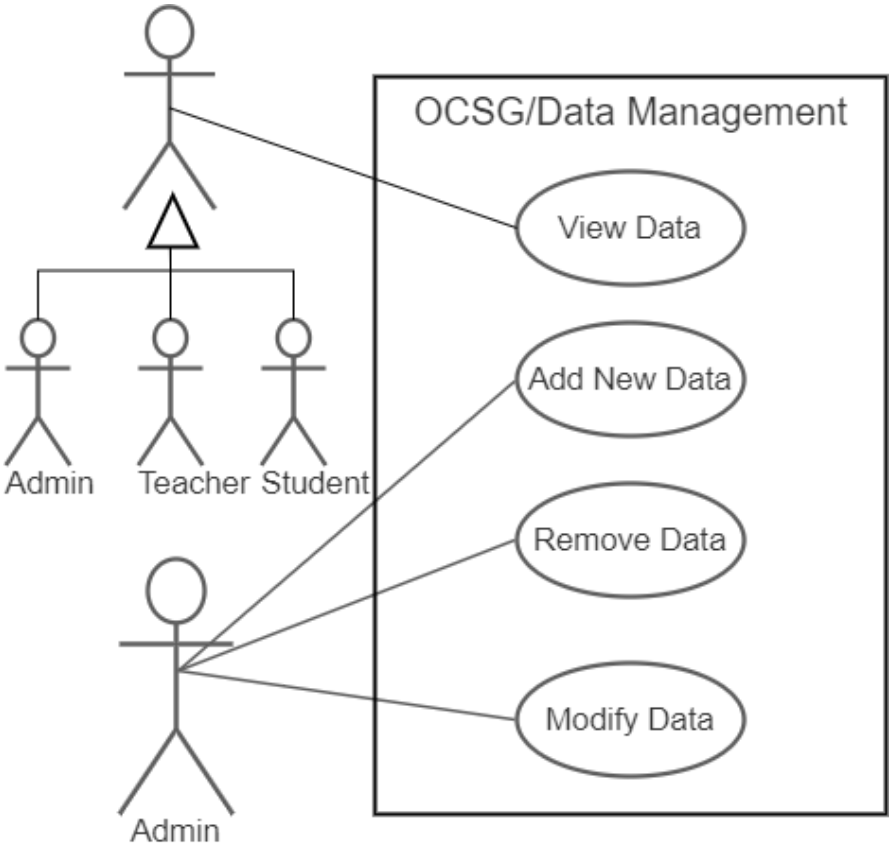
4.1.2 Generate



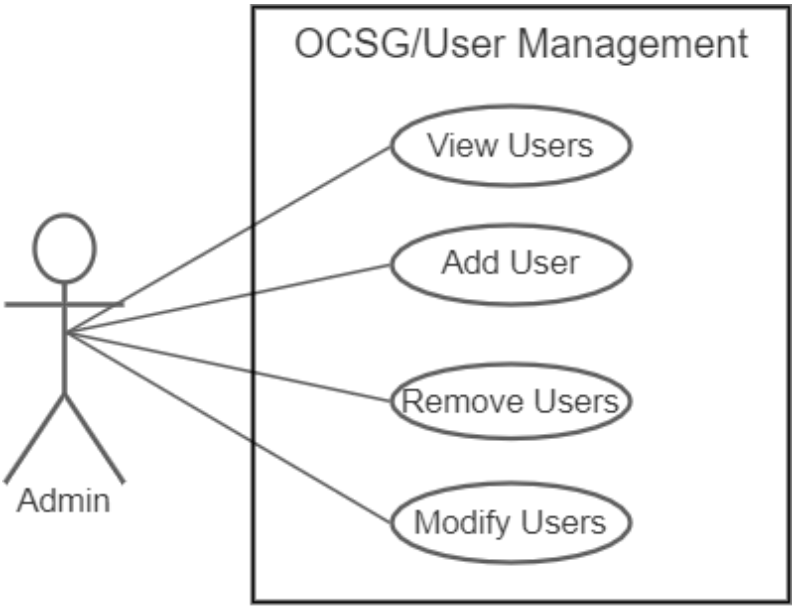
4.1.3 View Data



4.1.4 Data Management



4.1.5 User Management



4.2 Appendix B - Use Case Tables

4.2.1 View

Use Case ID:	VC-01		
Use Case Name:	View		
Created By:	Diego Draguicevich	Last Updated By:	Diego Draguicevich
Date Created:	2021-03-14	Date Last Updated:	2021-04-11

Actor:	User
Description:	A user views data from a table
Preconditions:	1. The user has a valid account on the system 2. The user has a computer with the system user interface installed 3. The user has an internet connection 4. The server portion of the system is active and has an internet connection
Postconditions:	1. The user has logged off the system
Priority:	Second
Frequency of Use:	At least once per semester
Normal Course of Events:	1. The user starts the system user interface on their computer 2. Once the system has connected to the server, the user logs in with their account credentials 3. The user selects search criteria for the information they wish to view via the appropriate user interface elements 4. When the user requests it, the system displays the search results from the server's database 5. Once the user is done viewing information, they log off and close the system user interface
Alternative Courses:	None
Exceptions:	VC.1.EX.1: The user requests a search which has no results The system informs the user their query has no results, and inquires as to whether they wish for another search VC.1.EX.2: Loss of contact with server The system user interface indicates a lack of connection to the user, and if a search query was interrupted the system stores the query to submit again once connection is reestablished with the server VC.1.EX.3: Incorrect login credentials

	The system informs the user their login credentials are incorrect, and prompts for reentry
Includes:	**UA-01**
Special Requirements:	**User interface should be satisfactory for users**
Assumptions:	**The server hosting the software and the client's computer have a network connection to each other**
Notes and Issues:	**None**

4.2.2 Generate Master Schedule

Use Case ID:	**GMS-01**		
Use Case Name:	**Generate Master Schedule**		
Created By:	**Doyle D Bigelow**	**Last Updated By:**	**Diego Draguicevich**
Date Created:	**2021-03-16**	**Date Last Updated:**	**2021-04-11**

Actor:	**Administrator**
Description:	**An administrator requests the generation of a Master Schedule**
Preconditions:	**1. Initial data is entered into the system.** **1.1. Buildings/classrooms** **1.2. Courses** **1.3. Timeslots** **1.4. Students** **1.5. Teachers** **1.6. Flowcharts** **2. User is logged in with account that has administrator permissions**
Postconditions:	**1. Course sections table is filled (Master Schedule is Created)** **2. The user has logged off the system**
Priority:	**Fourth**
Frequency of Use:	**At least once per semester**
Normal Course of Events:	**1. The user starts the system user interface on their computer** **2. Once the system has connected to the server, the user logs in with their account credentials** **3. The user clicks the button labeled "Generate Master Schedule"** **4. Within three minutes the schedule shall be generated, and a message box will pop up with the message "Master Schedule generated"** **5. Once the user is done viewing information, they log off and close the system user interface**

Alternative Courses:	**GMS.1.AC.1: Delete current Master Schedule before generating new Master Schedule** **1. Steps 1-2 of the Normal Course of Events occur** **2. The user clicks on the edit toolbar section** **3. The user clicks "Delete Current Master"** **4. A confirmation box appears with the message "Are you sure you want to delete the master schedule? This cannot be undone."** **5. The user clicks "Yes" to confirm their choice** **6. Steps 4-5 of the Normal Course of Events occur** **GMS.1.AC.2: Current Already Exists** **1. Steps 1-3 of the Normal Course of Events occur** **2. A confirmation box appears with the message "Are you sure you want to generate a new Master Schedule? Doing this will delete the already existing one."** **3. The user clicks "Yes" to confirm their choice** **4. Steps 4-5 of the Normal Course of Events occur**
Exceptions:	**GMS.1.EX.1: More students are enrolled than there are available classroom slots** **The system informs the user that too many students are enrolled and so a Master Schedule cannot be generated without more classroom space or fewer students** **GMS.1.EX.2: One or more of the required tables for generating the Master Schedule is missing** **The system does not generate a Master Schedule and instead informs the user which table is missing and that said table must be populated before a Master Schedule can be generated** **GMS.1.EX.3: The system takes too long to generate a Master Schedule** **The system informs the user that an error has occurred generating the Master Schedule and they should contact technical support** **GMS.1.EX.4: Connection to the server cannot be established** **The system informs the user that a connection to the server is not present and they should try again when a connection has been established** **GMS.1.EX.5: Incorrect login credentials**

	The system informs the user their login credentials are incorrect, and prompts for reentry
Includes:	**UA-01**
Special Requirements:	**1. The master schedule generation shall take no more than 3 minutes to generate a schedule from the time the user requests a schedule to time of delivery to the computer** **2. The process shall hold no more than 4GB of RAM.** **3. During any given second of the process shall hold no more than 90% of all processing time across all logical cores** **4. During any given second of the process on the user's personal computer the system shall hold no more than 50% of all processing time across all logical cores**
Assumptions:	**The server hosting the software and the client's computer have a network connection to each other**
Notes and Issues:	**None**

4.2.3 Database Management

Use Case ID:	**DBM-01**		
Use Case Name:	**Database Management**		
Created By:	**Brett Stricker**	**Last Updated By:**	**Diego Draguicevich**
Date Created:	**2021-03-17**	**Date Last Updated:**	**2021-04-11**

Actor:	**Administrator**
Description:	**An administrator modifies the data in the database**
Preconditions:	**1. The user has administrator access** **2. The database being modified contains data**
Postconditions:	**1. The user has requested the database update to reflect their changes** **2. The user has logged off the system**
Priority:	**Fifth**

Frequency of Use:	**Varies from once per semester to as often as within five minutes of a previous use**
Normal Course of Events:	**1. The user starts the system user interface on their computer** **2. Once the system has connected to the server, the user logs in with their account credentials** **3. The user selects the data that will be modified** **4. The user indicates the modifications to be made to the selected data** **5. The database is updated with the new information** **6. Once the user is done viewing information, they log off and close the system user interface**
Alternative Courses:	**DBM.1.AC.1** **1. Steps 1-5 of the Normal Course of Events occur** **2. The user indicated they wish to continue modifying data** **3. Steps 3-5 of the Normal Course of Events occur repeatedly at user discretion** **4. Step 5 of the Normal Course of Events occurs**
Exceptions:	**DBM.1.EX.1: Connection to the server cannot be established** **The system informs the user that a connection to the server is not present and they should try again when a connection has been established** **DBM.1.EX.2: Incorrect login credentials** **The system informs the user their login credentials are incorrect, and prompts for reentry** **DBM.1.EX.3: Database fails to process update request in time** **The system informs the user their request was nor received and the request has been terminated. Should the server-side of the system not receive a response from the client-side of the system after processing the modification request the server-side shall perform a data integrity check and rollback if data is out of order**
Includes:	**Login use case**
Special Requirements:	**Database is updated within 1 minute of receiving a modification request**

Assumptions:	**The server hosting the software and the client's computer have a network connection to each other**
Notes and Issues:	**None**

4.2.4 User Management

Use Case ID:	**UM-01**		
Use Case Name:	**User Management**		
Created By:	**Diego Draguicevich**	**Last Updated By:**	**Diego Draguicevich**
Date Created:	**2021-04-11**	**Date Last Updated:**	**2021-04-11**

Actor:	**Administrator**
Description:	**An administrator modifies the users in the database**
Preconditions:	**The user has administrator access**
Postconditions:	**1. The user has requested the database update to reflect their changes** **2. The user has logged off the system**
Priority:	**Sixth**
Frequency of Use:	**Varies from once per semester to as often as within five minutes of a previous use**
Normal Course of Events:	**1. The user starts the system user interface on their computer** **2. Once the system has connected to the server, the user logs in with their account credentials** **3. The user indicates whether they will add, remove, or modify user accounts** **4. The user indicates the modifications to be made to the selected user account(s)** **5. The database is updated with the new information** **6. Once the user is done viewing information, they log off and close the system user interface**

Alternative Courses:	**None**
Exceptions:	**DBM.1.EX.1: Connection to the server cannot be established** **The system informs the user that a connection to the server is not present and they should try again when a connection has been established** **DBM.1.EX.2: Incorrect login credentials** **The system informs the user their login credentials are incorrect, and prompts for reentry** **DBM.1.EX.3: Database fails to process update request in time** **The system informs the user their request was nor received and the request has been terminated. Should the server-side of the system not receive a response from the client-side of the system after processing the modification request the server-side shall perform a data integrity check and rollback if data is out of order**
Includes:	**UA-01**
Special Requirements:	**Database is updated within 1 minute of receiving a modification request**
Assumptions:	**The server hosting the software and the client's computer have a network connection to each other**
Notes and Issues:	**None**

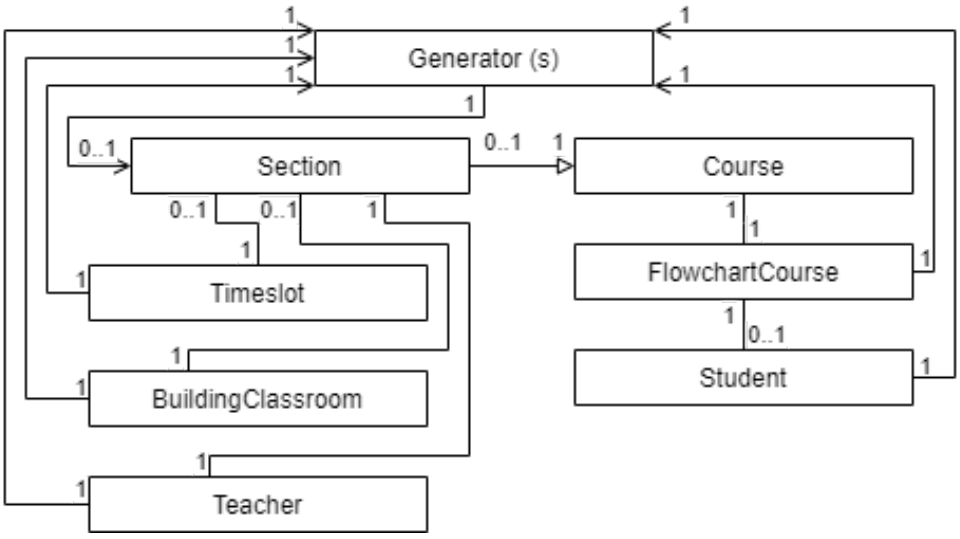
4.2.5 Authentication

Use Case ID:	**UA-01**		
Use Case Name:	**Authentication**		
Created By:	**Diego Draguicevich**	**Last Updated By:**	**Diego Draguicevich**
Date Created:	**2021-04-11**	**Date Last Updated:**	**2021-04-11**

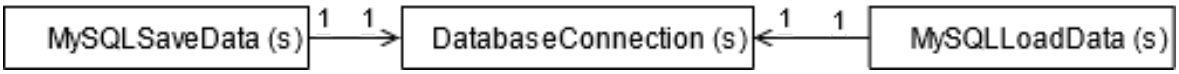
Actor:	**User**
Description:	**A user logs in to the User Interface**
Preconditions:	**The user has a valid user account**
Postconditions:	**The system has confirmed the user's credentials are valid and logs the user in**
Priority:	**Sixth**
Frequency of Use:	**Varies from once per semester to as often as within five minutes of a previous use**
Normal Course of Events:	**1. The user launched the User Interface Program** **2. The user provides the system their username and password** **3. The system indicates login credentials were accepted and transitions to the default home screen**
Alternative Courses:	**UM.1.AC.1: Invalid credentials** **1. Steps 1-2 of the Normal Course of Events occur** **2. The system detects invalid credentials** **3. The system indicates the user's credentials are invalid** **4. Repeat steps 1-2 of the Normal Course of Events until valid credentials are provided**
Exceptions:	**DBM.1.EX.1: Connection to the server cannot be established** **The system informs the user that a connection to the server is not present and they should try again when a connection has been established**
Includes:	**None**
Special Requirements:	**In the event of invalid credentials being presented, the system shall not indicated which part of the credentials are invalid, only that the credentials are invalid as a whole**
Assumptions:	**The server hosting the software and the client's computer have a network connection to each other**
Notes and Issues:	**None**

4.3 Appendix C - Class Diagrams

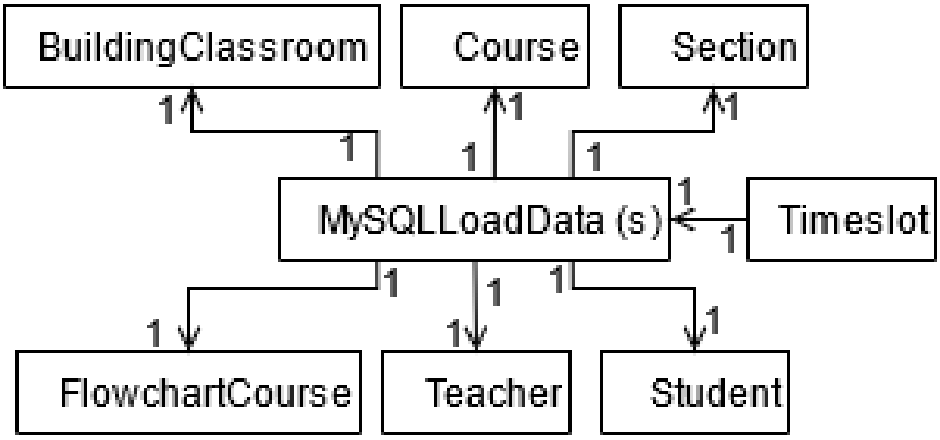
4.3.1 Simple Generate



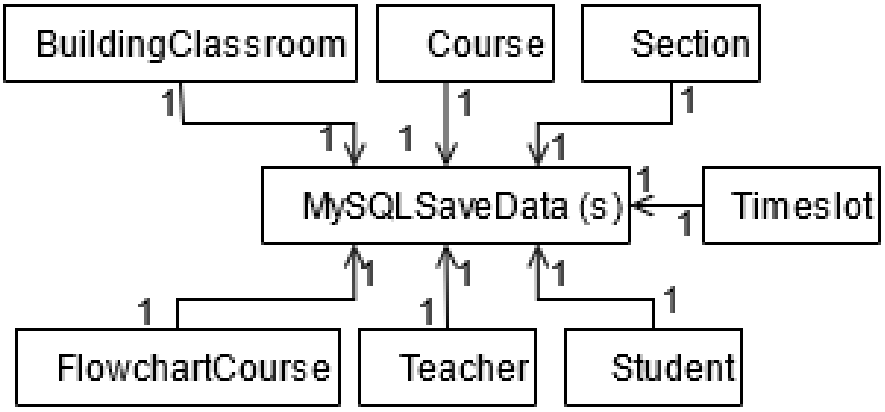
4.3.2 Simple Database Connector



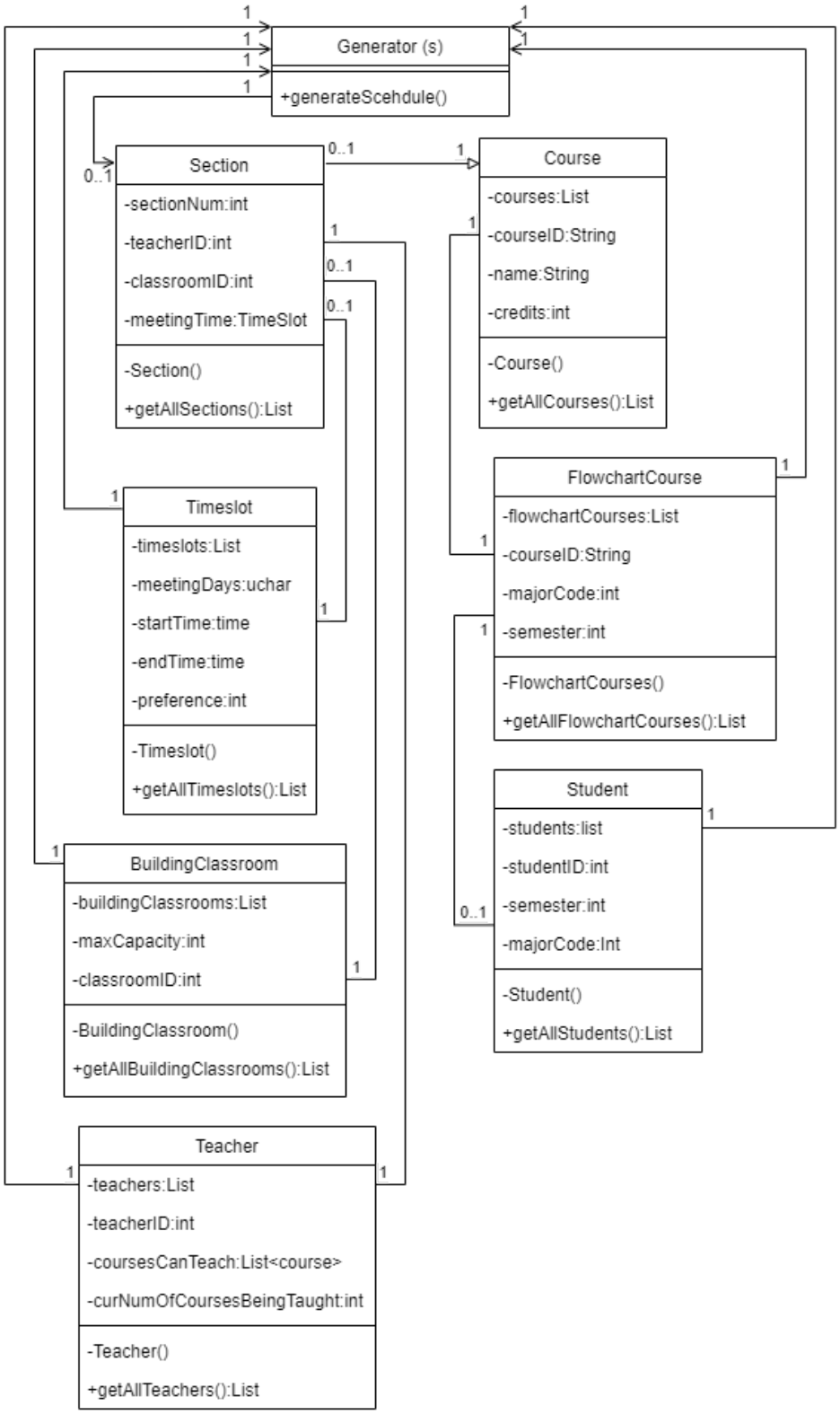
4.3.3 Simple Load Data



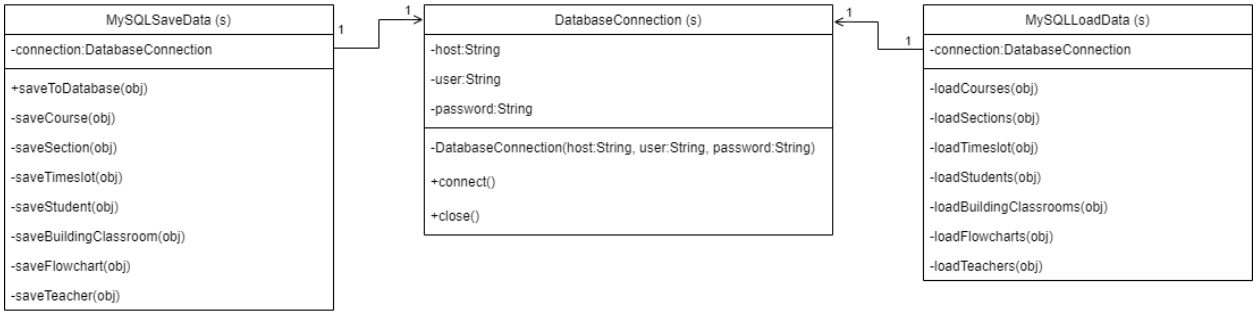
4.3.4 Simple Save Data



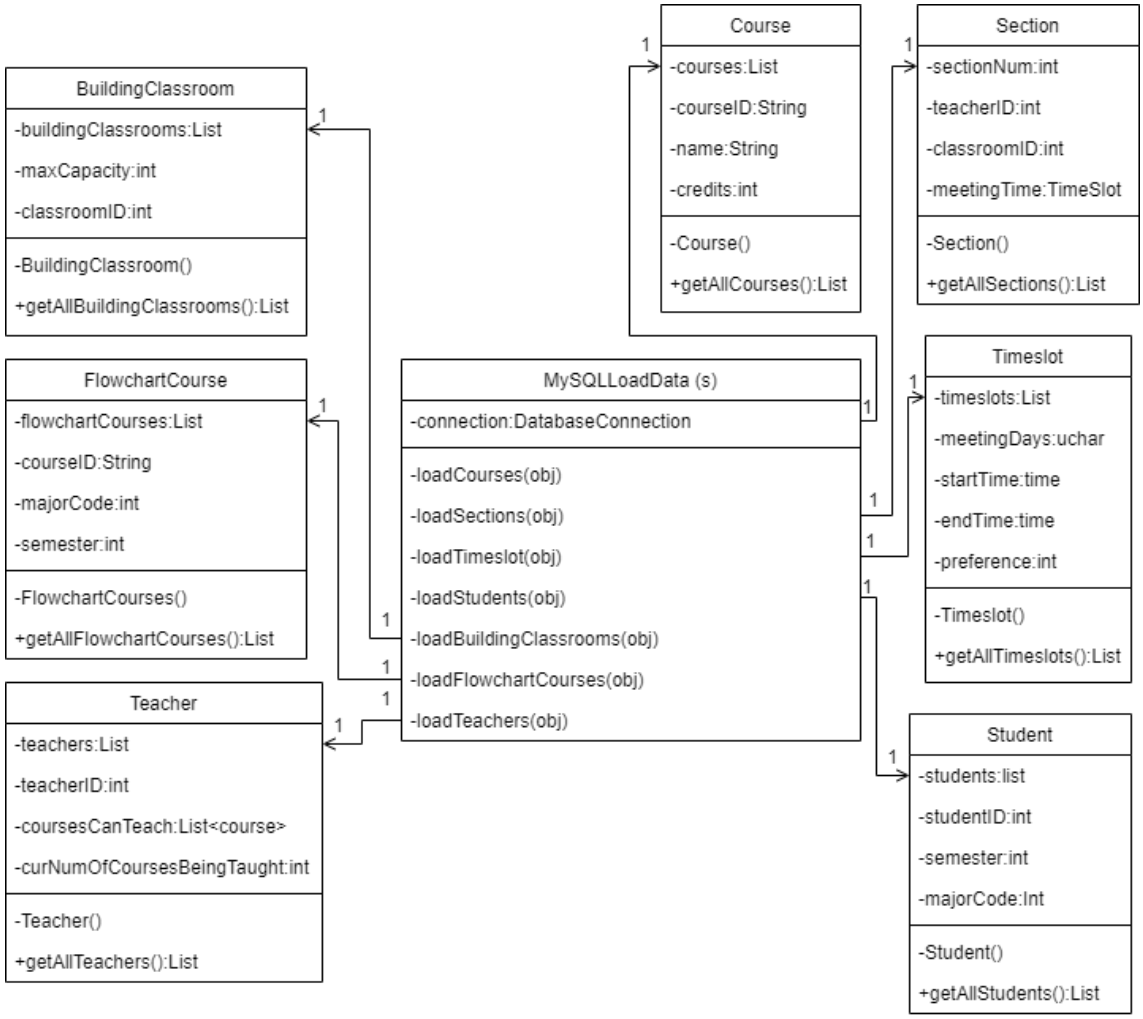
4.3.5 Generate



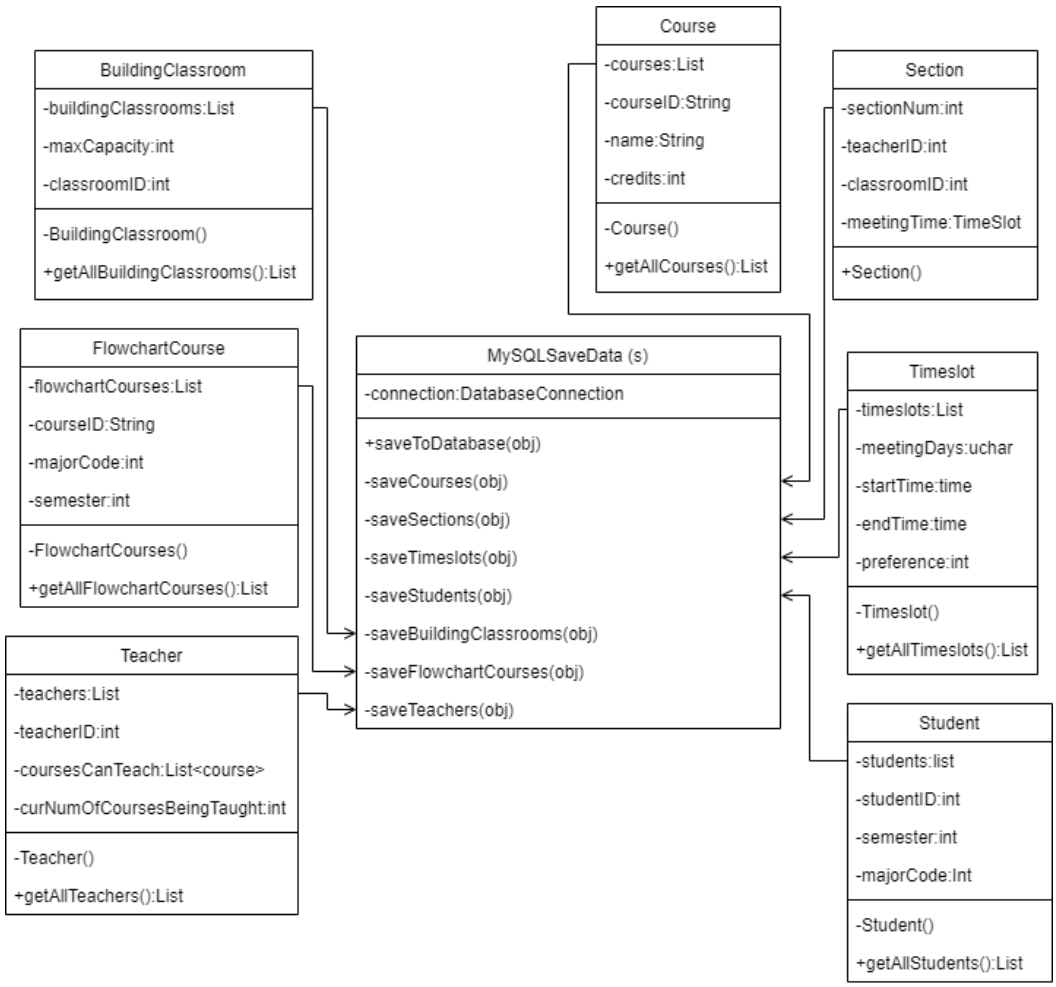
4.3.6 Database Connector



4.3.7 Load Data

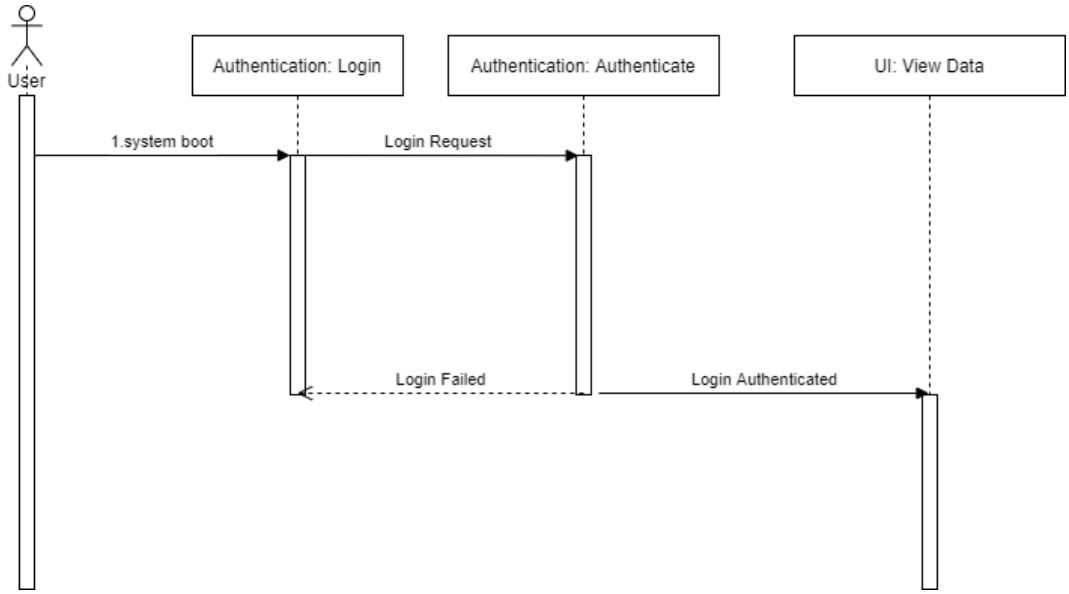


4.3.8 Save Data

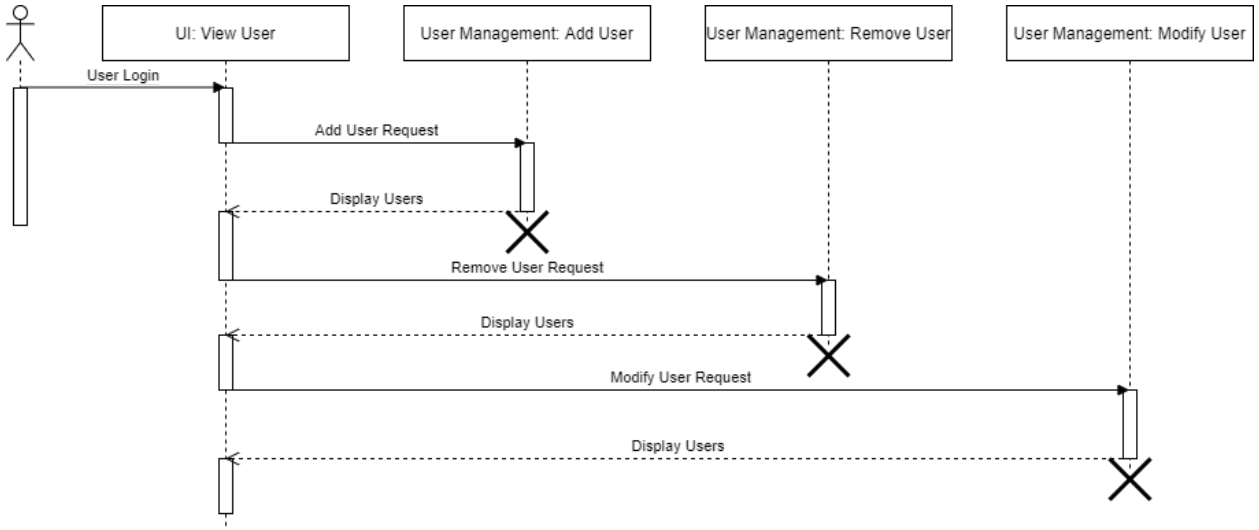


4.4 Appendix D - Sequence Diagram

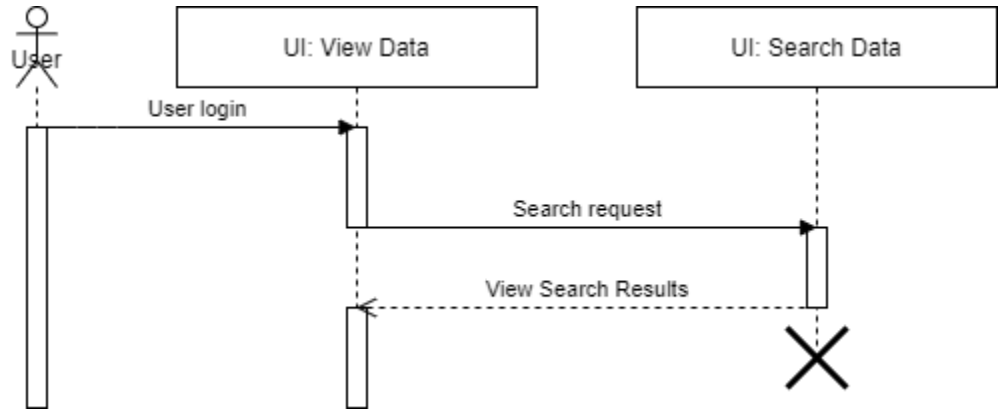
4.4.1 Authentication



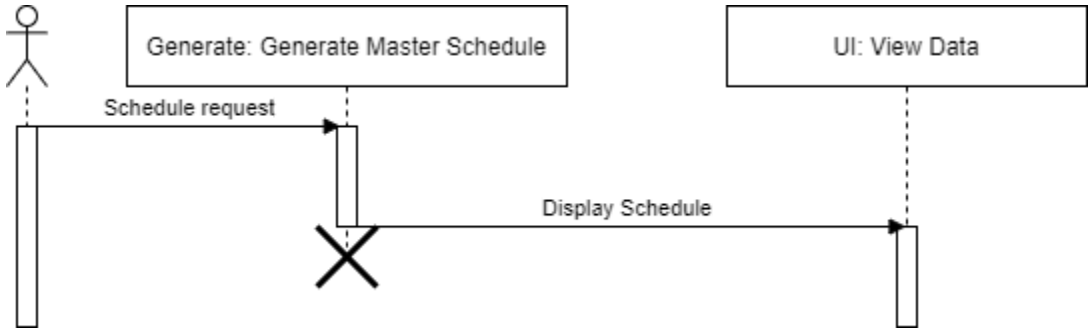
4.4.2 User Management



4.4.3 View Data



4.4.4 Generate



4.4.5 Data Management