System Design Document

Boardman Computer Science Lab Web Portal

Client: Mr. Christopher Dufour

Development Company: In-House Operations

Development Team:
Klei Bendo
Jack Brisson
Alex Landry
Samuel Morse
Aaron Schanck
Forrest Swift

Date: 10/28/2021
Version 1.0

Boardman Computer Science Lab Web Portal
System Design Document

Table of Contents

1. Introduction

The Boardman Computer Science Lab Web Portal is an all encompassing tool for computer science students at the University of Maine. It is designed to ensure better help for students seeking aid in both specific inquiries, and broad subject areas at the Boardman Computer Science Lab through use of an interactive calendar, individual and group meeting scheduling, forum posting, and news updates. The Web Portal will make the Boardman Computer Science Lab more accessible and easy to use for University of Maine Computer Science Students.

1.1 Purpose of This Document

The purpose of this document is to outline and describe the design of the Boardman Computer Science Lab Web Portal in terms of its system architecture and database. It details how each class is connected and the relationships between them. It is intended primarily for the client of the project and the development team to keep true to requirements and the project's intended design moving forward and for posterity of reference. This document is also intended for any other interested parties for official documentation of the application. It includes diagrams and descriptions of the system architecture with a decomposition description, diagrams and descriptions of the database, and a matrix of requirements as they relate to the different class diagrams.

1.2 References

Bendo, Klei, et al. "In-House Operations SRS." *Google Docs*, Google, 18 Oct. 2021, https://docs.google.com/document/d/1YIFScQdYOcsTWcKpfTEac3g4aTRo3XtSTO1Uay2CQv4/edit?usp=sharing.

Bendo, Klei, et al. "Ui Design Ideas." *Google Docs*, Google, 4 Nov. 2021, https://docs.google.com/document/d/1UTH4vEWQyTghzSD2DuvfVMSrA-7srWsTkONHvFu4Suo/edit?usp=sharing.

Dufour, Christopher, and Penny Rheingans. "dufour_help-Resource-Scheduling." 16 Sept. 2021.

Schanck, Aaron. "Team 17 Capstone Proposal." *Google Docs*, Google, 4 Oct. 2021, https://docs.google.com/document/d/19nm8LNdbCEEdSQNdVRj570LcsRJC7rjRumRwU4srtBE/edit?usp=sharing.

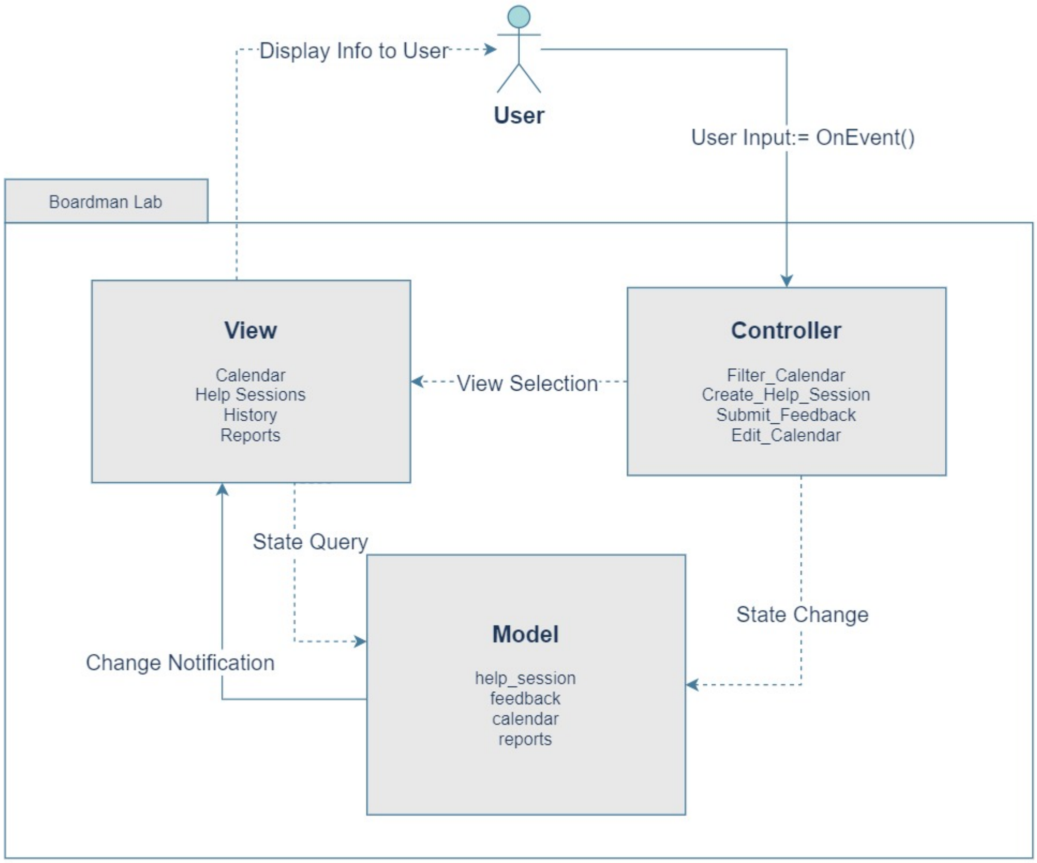
2. System Architecture

Included in this section are the class diagrams associated with the Boardman Computer Science Lab Web Portal including diagrams for Sign in/Sign Out, Help Scheduling, Feedback, and calendar. Additionally, this section includes diagrams depicting the decomposition of the components of the system. Overall it describes and overviews the front end design of the system architecture. For a complete description of which class diagrams associate with which requirements, see Section 4.

2.1 Architectural Design

For the system architecture, we are going to be using the model view controller design pattern. The backend will be using the Django framework with an SQL database (at this time we are between either mySQL or PostGreSQL) and the frontend will use a NodeJS framework. Django will handle the queries to the database (controller and models) and NodeJS will handle display and user interaction (view). We are going to be implementing a single sign-in using Google Oauth and Umaine login information to streamline user interactions and allow us to not store sensitive user information on our database. This means that we will not be storing identifiable user data - only an encrypted email address that will keep track of help sessions, attendance metrics, calendars and feedback.

Figure 1: Logical Architecture (Model, View, Controller Design Pattern)*
Showing a generalized architecture for the Boardman Lab Web Application.



*Please note that classes contained in 'views' and 'models' and methods within 'controller' do not represent the full scope of the application and are examples to assist in the understanding of the design pattern.

The Model View Controller design pattern is very useful for web applications because it allows for a highly dynamic user interface. Since we will have different user types, it is important that we manage permissions across user types. Typically, user permissions will be dictated by the authority prescribed via the administrator control panel, but depending on the information we can get from Google Oauth, permissions also might be handled directly by the umaine account system (for example, instructors and student aids have specific designations within their account).

Models are essentially the classes that are described below and shown in their respective class diagrams. The relationships between these classes will be represented in the database and their methods will be written in Django. The various views of the application are going to be coded in a combination of HTML, CSS and javascript while utilizing the Node.JS framework that will allow us to

2.2 Decomposition Description

2.2.1 Sign In/Out

Description (Sign In): When a user attempts to login to their profile, the system shall submit the imputed credentials to OAUTH for validation. If the attempt is successful, a token is generated containing the user's Maine.edu email name and the user is permitted to access the Boardman Computer Science Lab Web Portal.

Description (Sign Out): When a user clicks a UI element to sign out of their profile or if they have been inactive for a period of time, they will be brought to the login screen, and the token containing the user's Maine.edu email address will be removed.

Figure 2: Sign In Class Diagram

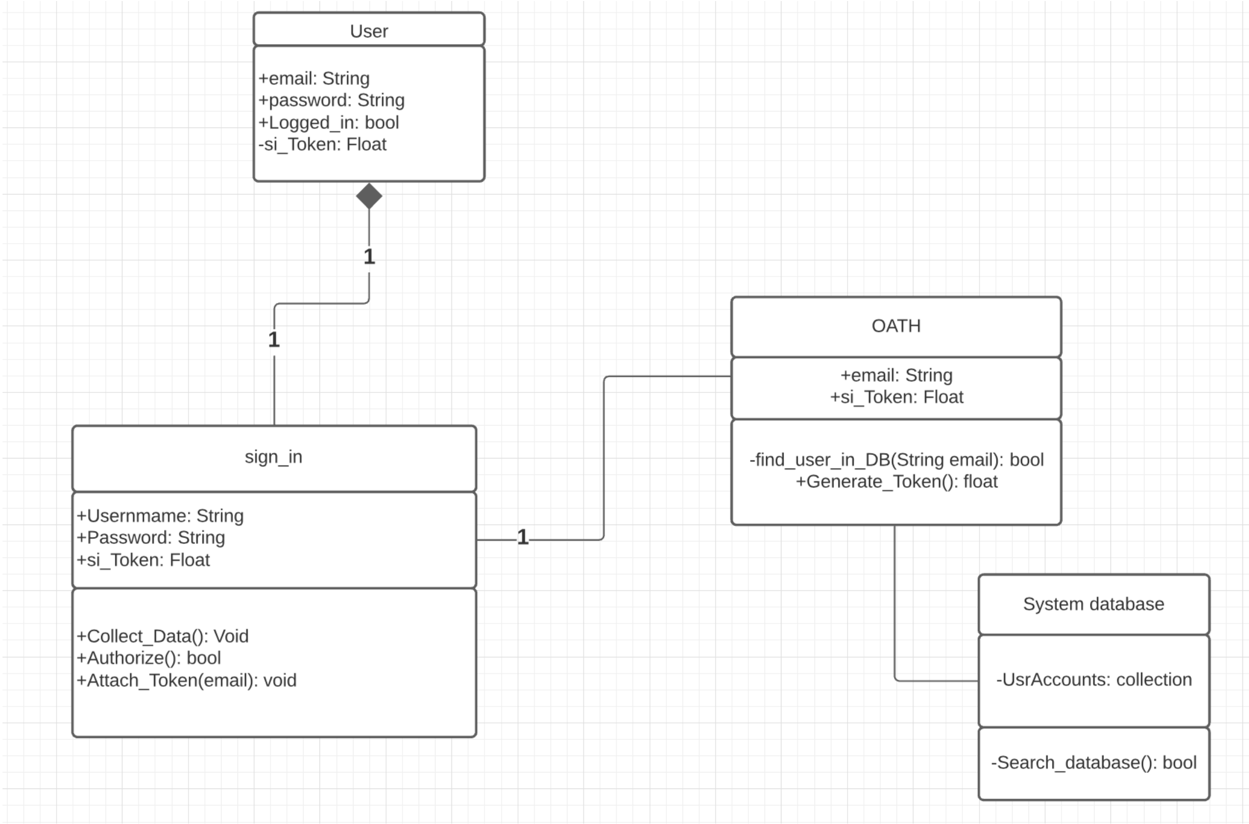
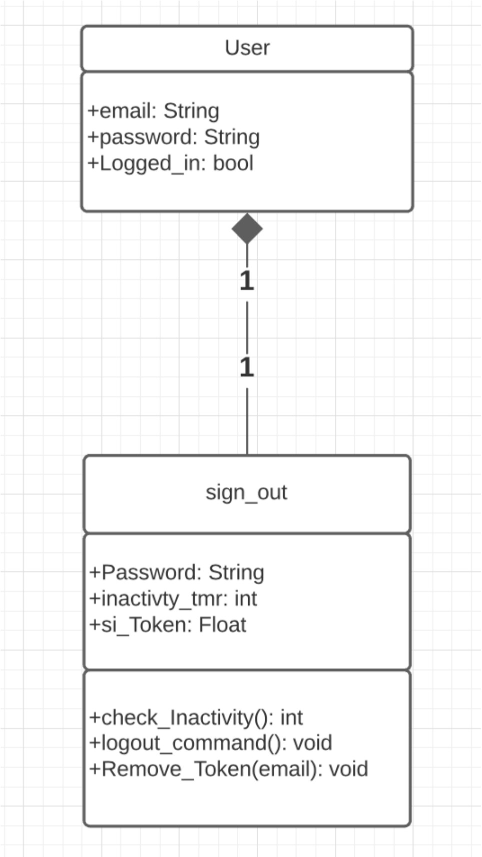


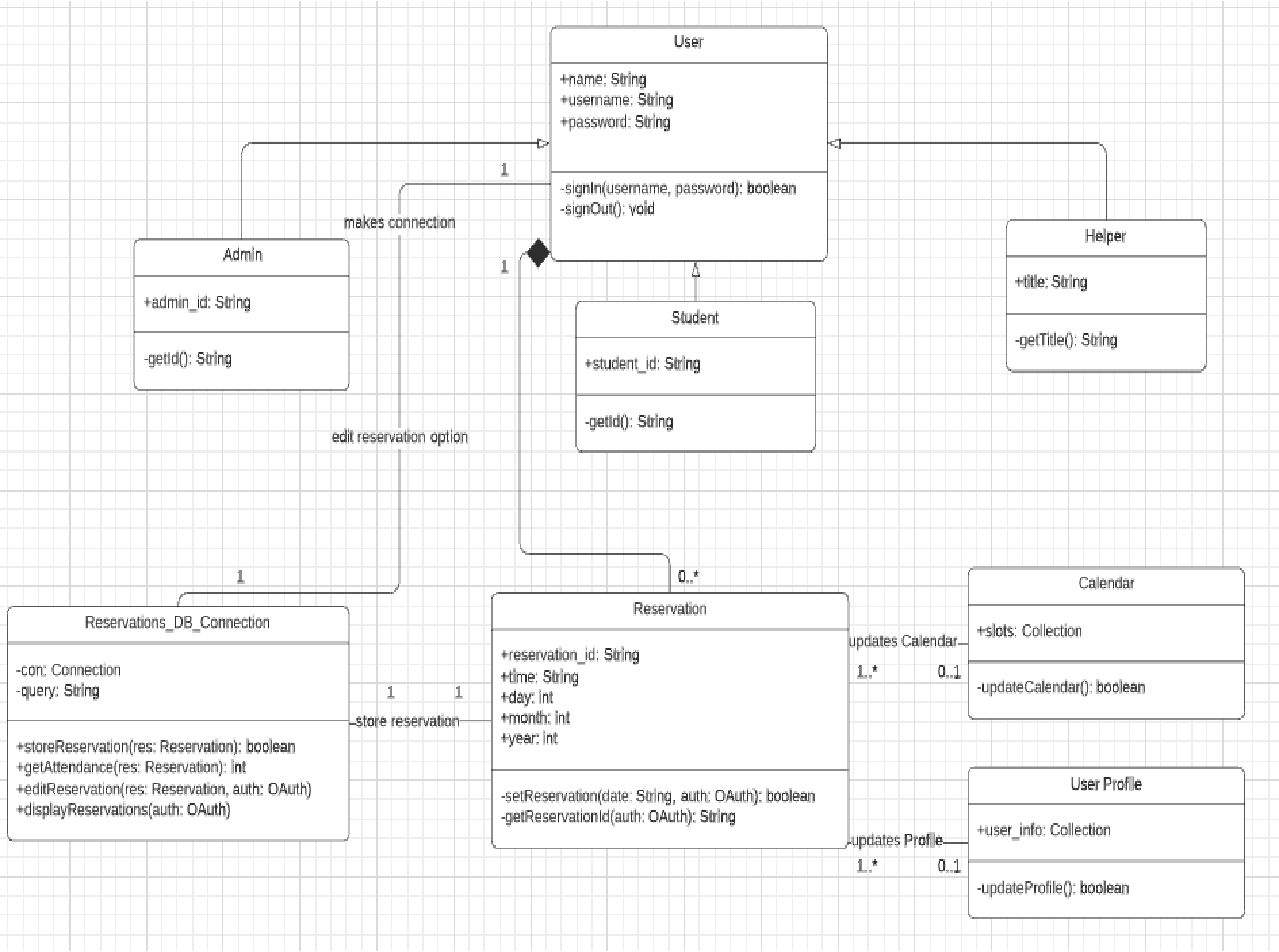
Figure 3: Sign Out Class Diagram



2.2.2 Store Reservation Request/ Edit Reservation

Description: When making a new reservation, or changing an existing one, the system shall be able to update the calendar and the user profile. It shall also create a new entry in the Reservations table of the Database. The system shall be able to display their own reservations to the student user, display the reservations they will help in to the helper user, and display all reservations to the administrator user. An administrator or a student shall be able to edit a reservation.

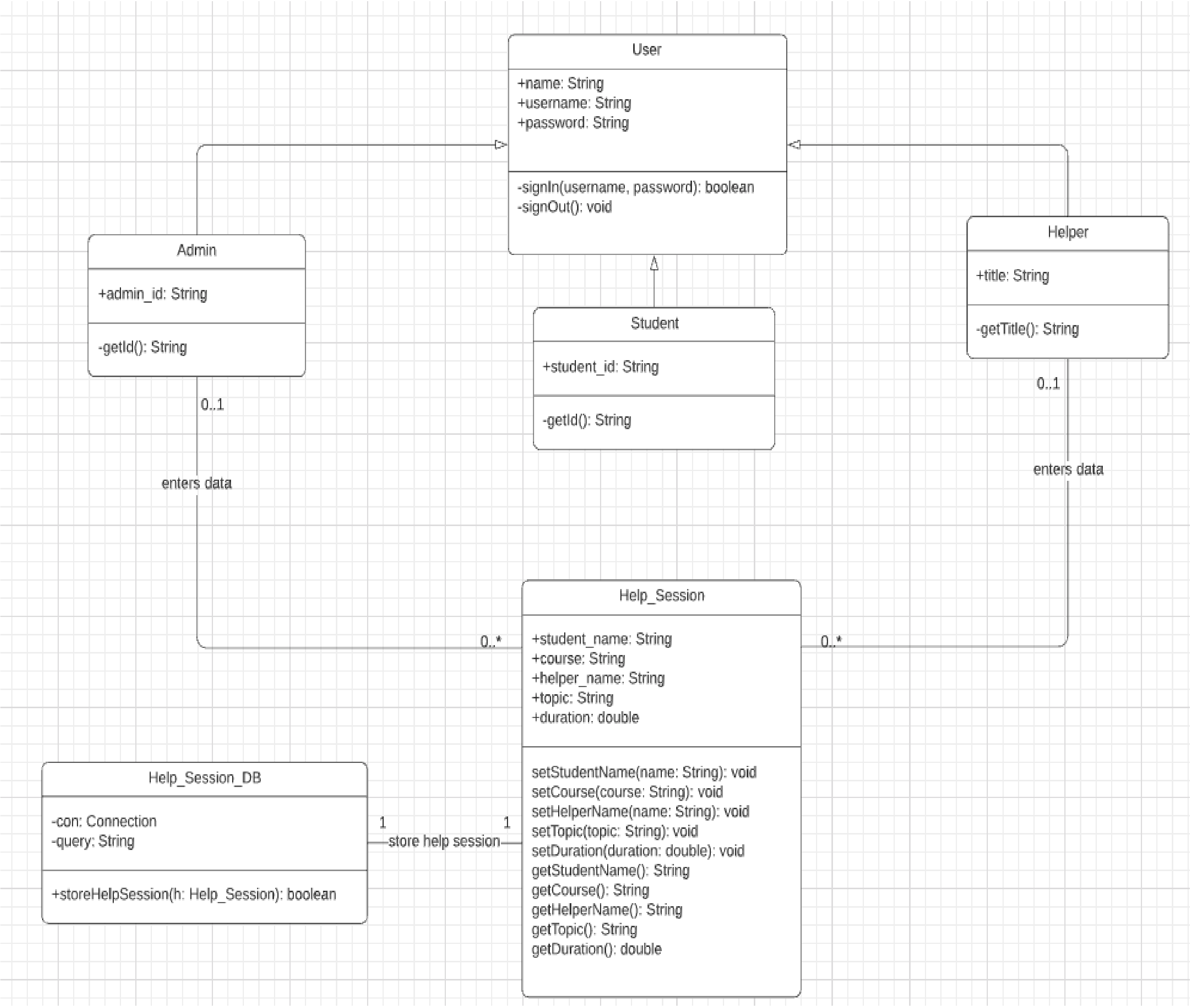
Figure 4: Store Reservation Request/ Edit Reservation Class Diagram



2.2.3 Store Help Session Data

Description: After a student provides feedback, an administrator or a helper shall be able to enter information about a particular help session. The information about the session will then be stored into the Database.

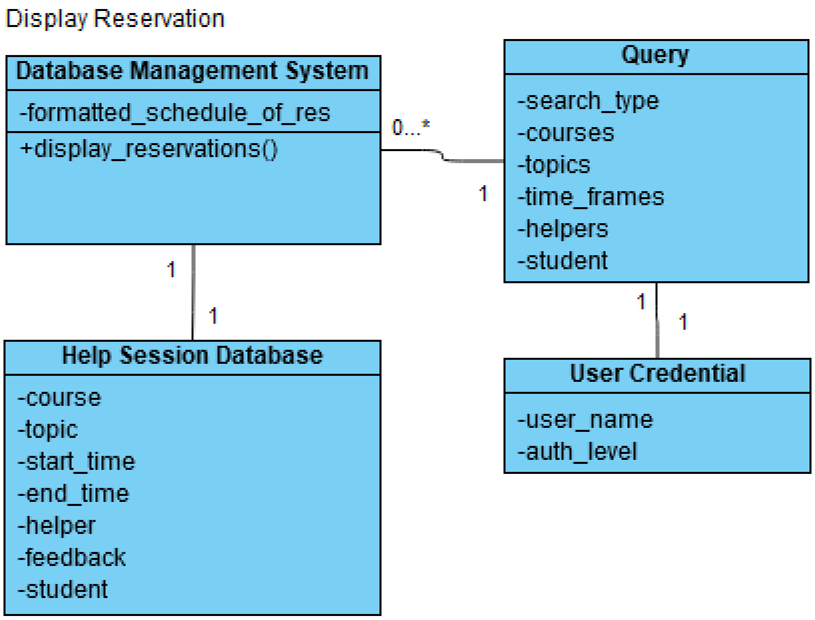
Figure 5: Store Help Session Data Class Diagram



2.2.4 Display Reservation

Description: Provides the data required to display reservation information according to a user's authorization level and query criteria.

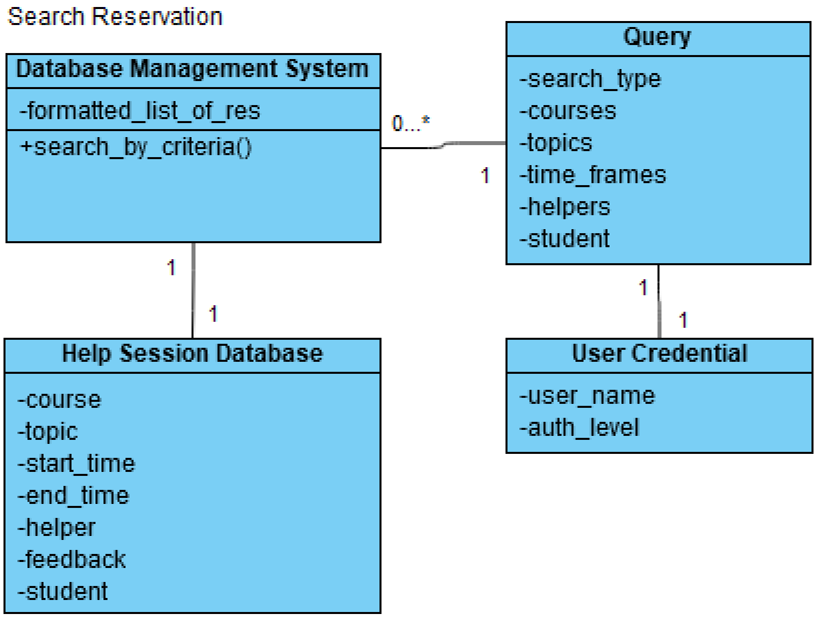
Figure 6: Display Reservation Class Diagram



2.2.5 Search Reservation

Description: Provides the data required to display all reservations that meet a specified criteria. This functionality is accessible only by administrators.

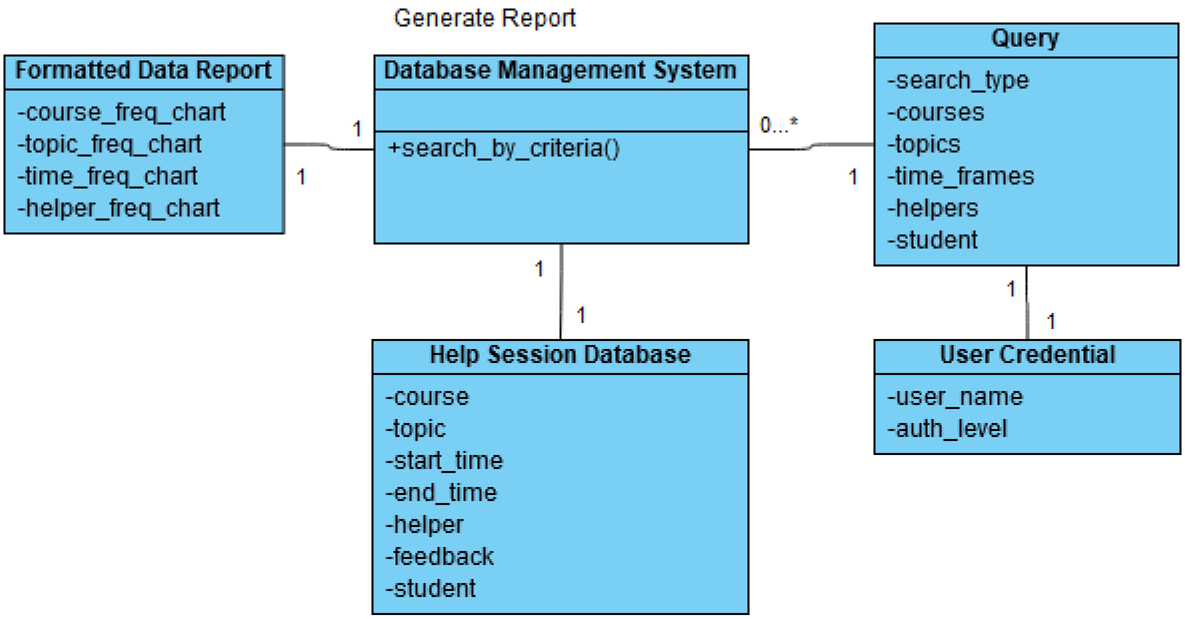
Figure 7: Search Reservation Class Diagram



2.2.6 Generate Report

Description: Generates several frequency charts to show distribution of student help sessions by course, topic, time, and helper. Generated reports sample from reservations that meet a specified criteria. This functionality is accessible only by administrators.

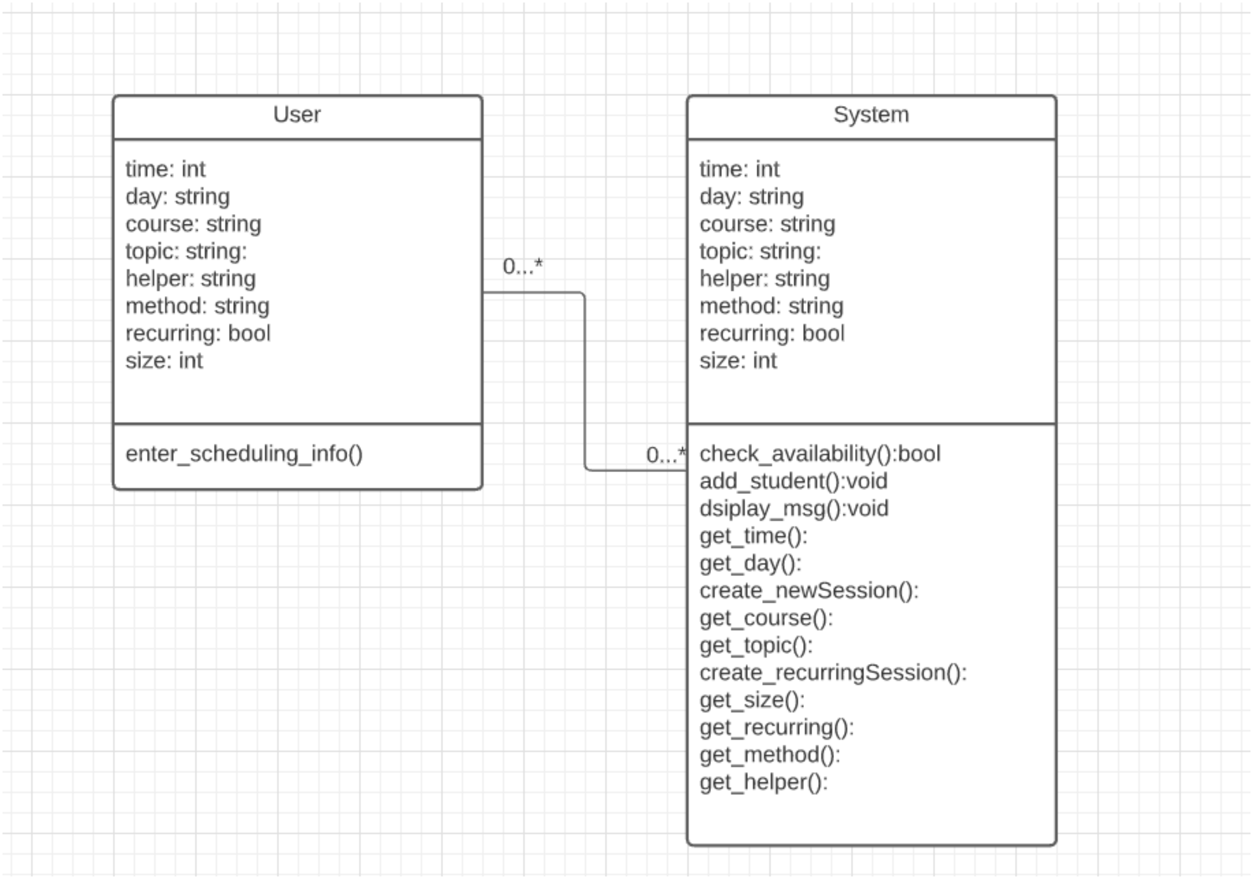
Figure 8: Generate Report Class Diagrams



2.2.7 Schedule Help Request

Description: This class diagram allows the user to create a help session request. The user enters time, day, method, size, course, topic, help, and whether it is supposed to be a recurring meeting. The system then checks availability and either places the student into the time slot or notifies the user that their time slot is not available.

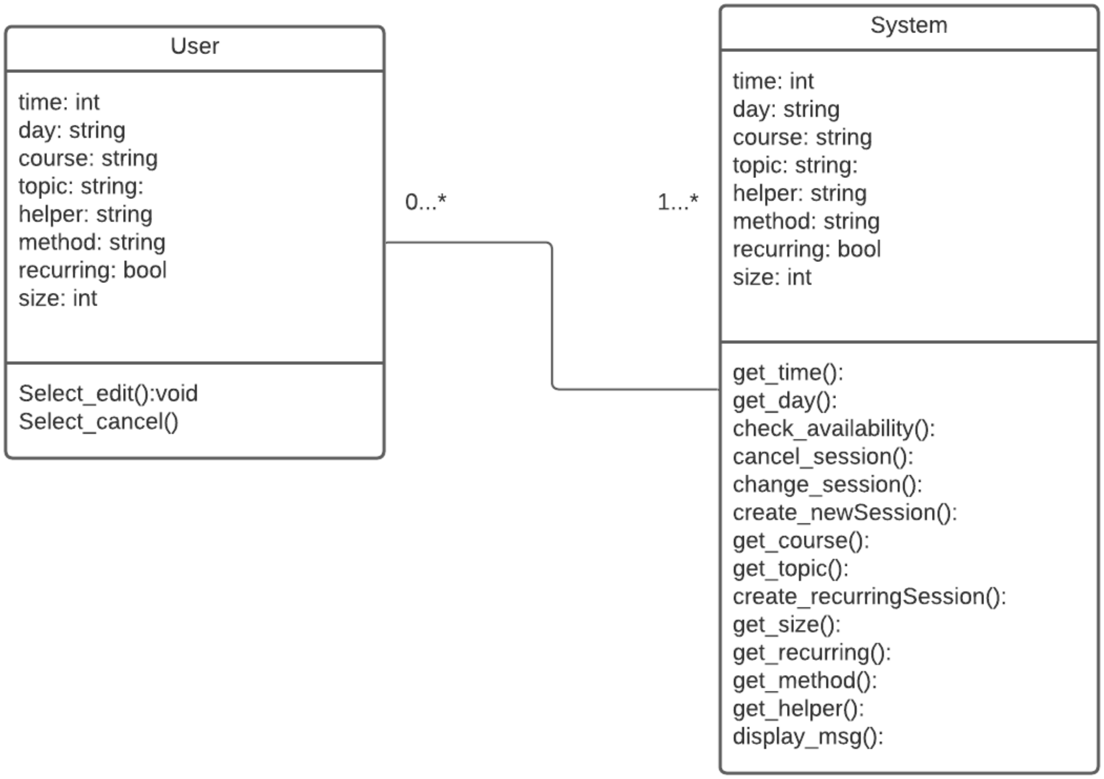
Figure 9: Schedule Help Request Class Diagrams



2.2.8 Edit Help Request

Description: This class diagram illustrates how a user would go about editing a help request. The User will be able to request changes to their help request by changing the time, day, method, size, course, topic, help, and whether it is supposed to be a recurring meeting. This Diagram shows the basic front end of this system.

Figure 10: Edit Help Request Class Diagrams



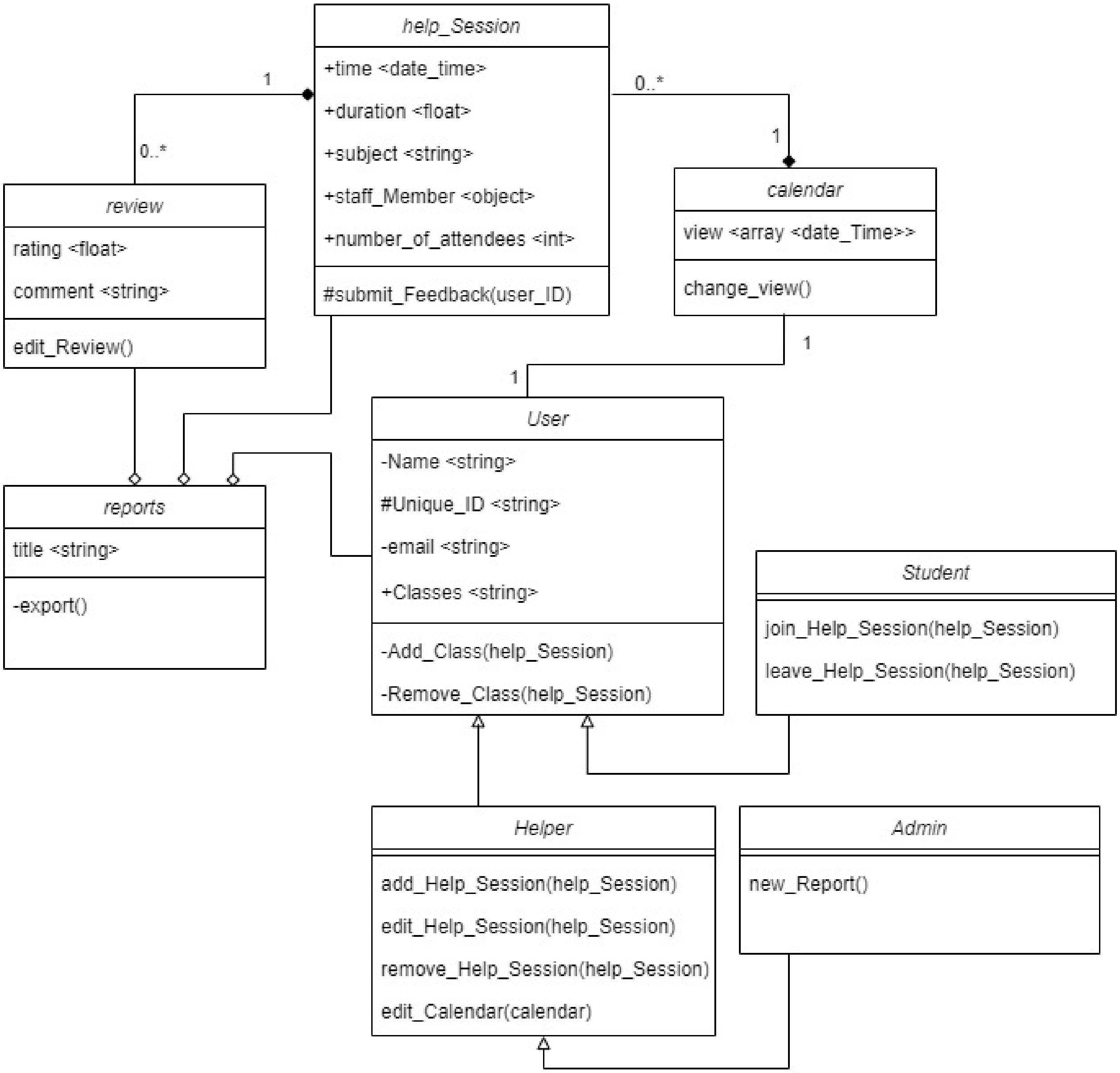
2.2.9 Submit Feedback/ User Hierarchy / Calendar View

Description: The purpose of this diagram is to expand on the front end features of the application by showing the interactions and relationships between the class models. The diagram shows the calendar, help sessions feedback (reviews) and reports as well as the inheritance hierarchy of ‘user’. It also shows how reporting is an aggregation of many different classes including feedback, users and help sessions.

A help session is a composition of reviews - and reviews/feedback cannot exist without a corresponding help session. This is also the case for the relationship between the calendar and help sessions. The calendar holds all of the help sessions that are active in the system and organizes them within the calendar view (see Architectural Design Section: 2.1).

Admin inherits from Helper and both Helper and Student inherits from User. This separation is important for the permission hierarchy of the application. Admins will have the ability to generate reports and gather data from the help sessions while helpers will be able to alter their own schedules in the lab and manage their individual help sessions that they own. Students will be able to sign up for help sessions, cancel a reservation for help and submit feedback on help sessions that they have attended.

Figure 11 Submit Feedback / User Hierarchy / Calendar View Class Diagrams



3. Persistent Data Design

Included in this section are both descriptions of the proposed database for the Boardman Computer Science Lab Web Portal and class diagrams depicting the relations between each larger area of the database as a whole. If there were files that were important to include for this project they would also be included in this section. Moreover this section provides a description of the back end processes of the application.

3.1 Database Descriptions

- Our database will consist of a single table of reservations.
- Users with student authorization will only see reservations they have made.

- Users with helper authorization will be able to see any reservation they are a helper in or any reservations they have made.
- Users with administrator authorization will be able to see all reservations.
- Start and End times are in UTC with the format YYYY-MM-DD HH:MM:SS
- Student and Helper identifiers are in the format firstname.lastname

Figure 12: Database Table Fields and Data Types

ID	**Course**	**Topic**	**Start Time**	**End Time**	**Student**	**Helper**	**Feedback**
integer	text	text	text	text	text	text	text

3.2 File Descriptions

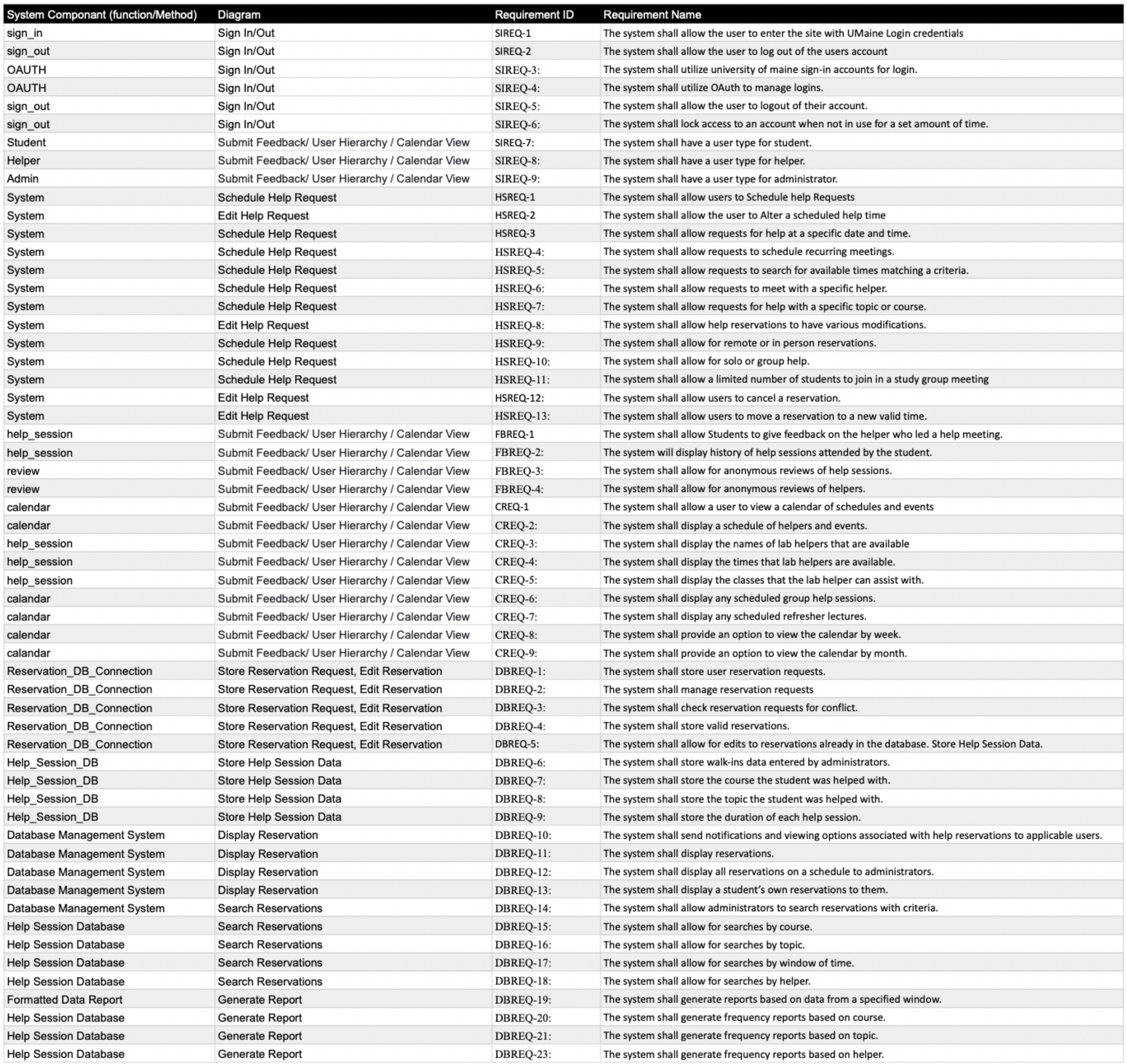
No files are used in the production or use of this application.

4. Requirements Matrix

This section outlines which system components satisfy each of the functional requirements from the SRS. Each relationship is outlined using a tabular format including the associated system component, the diagram it’s located in, the requirement ID, and Requirement name.

4.1 Requirements Matrix Expanded

Figure 13: Requirements Matrix



System Componant (function/Method)	Diagram	Requirement ID	Requirement Name
sign_in	Sign In/Out	SIREQ-1	The system shall allow the user to enter the site with UMaine Login credentials
sign_out	Sign In/Out	SIREQ-2	The system shall allow the user to log out of the users account
OAUTH	Sign In/Out	SIREQ-3:	The system shall utilize university of maine sign-in accounts for login.
OAUTH	Sign In/Out	SIREQ-4:	The system shall utilize OAuth to manage logins.
sign_out	Sign In/Out	SIREQ-5:	The system shall allow the user to logout of their account.
sign_out	Sign In/Out	SIREQ-6:	The system shall lock access to an account when not in use for a set amount of time.
Student	Submit Feedback/ User Hierarchy / Calendar View	SIREQ-7:	The system shall have a user type for student.
Helper	Submit Feedback/ User Hierarchy / Calendar View	SIREQ-8:	The system shall have a user type for helper.
Admin	Submit Feedback/ User Hierarchy / Calendar View	SIREQ-9:	The system shall have a user type for administrator.
System	Schedule Help Request	HSREQ-1	The system shall allow users to Schedule help Requests
System	Edit Help Request	HSREQ-2	The system shall allow the user to Alter a scheduled help time
System	Schedule Help Request	HSREQ-3	The system shall allow requests for help at a specific date and time.
System	Schedule Help Request	HSREQ-4:	The system shall allow requests to schedule recurring meetings.
System	Schedule Help Request	HSREQ-5:	The system shall allow requests to search for available times matching a criteria.
System	Schedule Help Request	HSREQ-6:	The system shall allow requests to meet with a specific helper.
System	Schedule Help Request	HSREQ-7:	The system shall allow requests for help with a specific topic or course.
System	Edit Help Request	HSREQ-8:	The system shall allow help reservations to have various modifications.
System	Schedule Help Request	HSREQ-9:	The system shall allow for remote or in person reservations.
System	Schedule Help Request	HSREQ-10:	The system shall allow for solo or group help.
System	Schedule Help Request	HSREQ-11:	The system shall allow a limited number of students to join in a study group meeting
System	Edit Help Request	HSREQ-12:	The system shall allow users to cancel a reservation.
System	Edit Help Request	HSREQ-13:	The system shall allow users to move a reservation to a new valid time.
help_session	Submit Feedback/ User Hierarchy / Calendar View	FBREQ-1	The system shall allow Students to give feedback on the helper who led a help meeting.
help_session	Submit Feedback/ User Hierarchy / Calendar View	FBREQ-2:	The system will display history of help sessions attended by the student.
review	Submit Feedback/ User Hierarchy / Calendar View	FBREQ-3:	The system shall allow for anonymous reviews of help sessions.
review	Submit Feedback/ User Hierarchy / Calendar View	FBREQ-4:	The system shall allow for anonymous reviews of helpers.
calendar	Submit Feedback/ User Hierarchy / Calendar View	CREQ-1	The system shall allow a user to view a calendar of schedules and events
calendar	Submit Feedback/ User Hierarchy / Calendar View	CREQ-2:	The system shall display a schedule of helpers and events.
help_session	Submit Feedback/ User Hierarchy / Calendar View	CREQ-3:	The system shall display the names of lab helpers that are available
help_session	Submit Feedback/ User Hierarchy / Calendar View	CREQ-4:	The system shall display the times that lab helpers are available.
help_session	Submit Feedback/ User Hierarchy / Calendar View	CREQ-5:	The system shall display the classes that the lab helper can assist with.
calandar	Submit Feedback/ User Hierarchy / Calendar View	CREQ-6:	The system shall display any scheduled group help sessions.
calandar	Submit Feedback/ User Hierarchy / Calendar View	CREQ-7:	The system shall display any scheduled refresher lectures.
calendar	Submit Feedback/ User Hierarchy / Calendar View	CREQ-8:	The system shall provide an option to view the calendar by week.
calandar	Submit Feedback/ User Hierarchy / Calendar View	CREQ-9:	The system shall provide an option to view the calendar by month.
Reservation_DB_Connection	Store Reservation Request, Edit Reservation	DBREQ-1:	The system shall store user reservation requests.
Reservation_DB_Connection	Store Reservation Request, Edit Reservation	DBREQ-2:	The system shall manage reservation requests
Reservation_DB_Connection	Store Reservation Request, Edit Reservation	DBREQ-3:	The system shall check reservation requests for conflict.
Reservation_DB_Connection	Store Reservation Request, Edit Reservation	DBREQ-4:	The system shall store valid reservations.
Reservation_DB_Connection	Store Reservation Request, Edit Reservation	DBREQ-5:	The system shall allow for edits to reservations already in the database. Store Help Session Data.
Help_Session_DB	Store Help Session Data	DBREQ-6:	The system shall store walk-ins data entered by administrators.
Help_Session_DB	Store Help Session Data	DBREQ-7:	The system shall store the course the student was helped with.
Help_Session_DB	Store Help Session Data	DBREQ-8:	The system shall store the topic the student was helped with.
Help_Session_DB	Store Help Session Data	DBREQ-9:	The system shall store the duration of each help session.
Database Management System	Display Reservation	DBREQ-10:	The system shall send notifications and viewing options associated with help reservations to applicable users.
Database Management System	Display Reservation	DBREQ-11:	The system shall display reservations.
Database Management System	Display Reservation	DBREQ-12:	The system shall display all reservations on a schedule to administrators.
Database Management System	Display Reservation	DBREQ-13:	The system shall display a student's own reservations to them.
Database Management System	Search Reservations	DBREQ-14:	The system shall allow administrators to search reservations with criteria.
Help Session Database	Search Reservations	DBREQ-15:	The system shall allow for searches by course.
Help Session Database	Search Reservations	DBREQ-16:	The system shall allow for searches by topic.
Help Session Database	Search Reservations	DBREQ-17:	The system shall allow for searches by window of time.
Help Session Database	Search Reservations	DBREQ-18:	The system shall allow for searches by helper.
Formatted Data Report	Generate Report	DBREQ-19:	The system shall generate reports based on data from a specified window.
Help Session Database	Generate Report	DBREQ-20:	The system shall generate frequency reports based on course.
Help Session Database	Generate Report	DBREQ-21:	The system shall generate frequency reports based on topic.
Help Session Database	Generate Report	DBREQ-23:	The system shall generate frequency reports based on helper.

Appendix A – Agreement Between Customer and Contractor

This section denotes that both the client and the development team have agreed upon the information contained within this document. It will be used as both a guideline and as an end goal in terms of the requirements needed for the application to function to the clients vision.

In the case that an addition or edit be needed after the completion and signing of this document, the change or addition must be agreed upon by both client and development team and included in **Appendix D - Document Additions** with the title of the addition, date, brief description, and signature from both parties.

-Client-

Name: Mr. Christopher Dufour

Date: 11/9/2021

Signature: Christopher Dufour

Christopher Dufour (Nov 9, 2021 13:26 EST)

-Development Team-

Name: Klei Bendo

Date: 11/9/2021

Signature:

Kleition Bendo (Nov 9, 2021 15:02 EST)

Name: Jack Brisson

Date: 11/10/2021

Signature: John H Brisson

John H Brisson (Nov 10, 2021 15:34 EST)

Name: Alex Landry

Date: 11/10/21

Signature: Alex Landry

Alex Landry (Nov 10, 2021 15:55 EST)

Name: Samuel Morse

Date: 11/9/2021

Signature:

Samuel Morse (Nov 9, 2021 19:50 EST)

Name: Aaron Schanck

Date: 11/10/2021

Signature: Schanck

aaron schanck (Nov 10, 2021 15:56 EST)

Name: Forrest Swift

Date: 11/9/2021

Signature: Forrest Swift

Forrest Swift (Nov 9, 2021 14:10 EST)

Client Comments (Continues on next page if needed):

Client Comments Cont.

Appendix B – Team Review Sign-off

This section denotes that all members of the In-House Operations development team have reviewed this document and agree on its content and format. If any minor disagreements in content and format are present, they are listed below the development team signatures.

Name: Klei Bendo

Date: 11/9/2021

Signature:

Kleition Bendo (Nov 9, 2021 15:02 EST)

Name: Jack Brisson

Date: 11/10/2021

Signature:

John H Brisson

John H Brisson (Nov 10, 2021 15:34 EST)

Name: Alex Landry

Date: 11/10/21

Signature:

Alex Landry

Alex Landry (Nov 10, 2021 15:55 EST)

Name: Samuel Morse

Date: 11/9/2021

Signature:

Samuel Morse (Nov 9, 2021 19:50 EST)

Name: Aaron Schanck

Date: 11/10/2021

Signature:

Schanck

aaron schanck (Nov 10, 2021 15:56 EST)

Name: Forrest Swift

Date: 11/9/2021

Signature:

Forrest Swift

Forrest Swift (Nov 9, 2021 14:10 EST)

Minor Disagreements in Content and Format (if any):

Appendix C – Document Contributions

This section denotes the contributions of each team member to this document. It includes the sections each member worked on and their percentage contributed in parentheses.

Name: Klei Bendo
Sections worked on (percentage contributed):
Section 2: 15% (Sections 2.2.2, 2.2.3)
Section 3: 10% (Edits and final lookover of the content of section 3)

Name: Jack Brisson
Sections worked on (percentage contributed):
Section 2: 15% (Section 2.2.1)

Name: Alex Landry
Sections worked on (percentage contributed):
Section 2: 15% (sections 2.2.7, 2.2.8)

Name: Samuel Morse
Sections worked on (percentage contributed):
Section 2: 35% (secitons 2.1, 2.2.9)
Section 3: 10% (additional formatting and edits)

Name: Aaron Schanck
Sections worked on (percentage contributed):
Section 1: 100%
Section 2: 5% (introduction)
Section 3: 5% (introduction)
Section 4: 100%
Appendices: 100%

Name: Forrest Swift
Sections worked on (percentage contributed):
Section 2: 15% (sections 2.2.4, 2.2.5, 2.2.6)
Section 3: 75% (description and table)

Appendix D – Document Additions

No Document additions to date.