Computer Science 4490Z Final Report:
TA Management Application

Alex Gyori
Student number: 250643023
Email: agyori@uwo.ca

Department of Computer Science
University of Western Ontario
London, Ontario, Canada

Supervisor: Laura Reid

Abstract

This paper fully details the results of developing the TA Manager, a web-based node.js application designed to manage teaching assistants, courses, supervisors, and their assignments between each other. The paper also gives rationale behind design decisions and provides a multitude of screen captures of the user interface in action. The TA Manager has other features like automatic assignment of teaching assistants to courses, and the generation of many different types of PDF reports.

The technology and libraries used in the TA Manager are also detailed, and the PostgreSQL database is shown and broken down by table. The relations between each table is also explored.

Acknowledgements

Many thanks to Laura Reid for supervising this thesis project and for her timely responses to my inquiries. She also provided the requirements and feature list for the TA Manager.

I would also like to thank Jeff Shantz for his guidance in technology selection and development strategies during this project.

Table of Contents

Introduction

Courses at Western have teaching assistants (TAs) that are assigned to courses. These TA assignments are done by the CS graduate admin assistant with input from CS professors. When assigning TAs to courses, the graduate admin assistant must adhere to specific rules. For example, courses may only have a finite amount of TAs and a graduate student may only be assigned as a TA a certain amount of times. Aside from the mandatory rules, graduate admin assistants may also take into consideration certain desirable conditions when determining TA assignments. For example, if a TA has been assigned to a course previously then a graduate admin assistant might be more inclined to assign that TA to the course a second time.

Currently, this process is done entirely by hand. This makes the TA assignment process time-consuming, complex, and error-prone. This project’s goal is to develop software that will simplify and expedite this process.

Chapter 1. User Manual

1.1 TA Manager

1.1.1 Site Structure

The TA Manager web application that consists of three pages:

- **Courses Page**: contains a list of courses and the TAs assigned to them. From this page, the user can add, edit, and remove courses, as well as assign TAs to courses. This page also allows the user to automatically assign TAs to courses using the Auto-Assignment feature. PDF reports relating to courses can also be generated here.
- **Teaching Assistants Page**: contains a list of TAs. This page allows a user to add, edit, and remove TAs. The user can also view what courses and supervisors the TA is assigned to, and can generate PDF reports pertaining to TAs.
- **Supervisors Page**: contains a list of supervisors and the TAs assigned to them. Note that supervisors are *not* professors. From this page the user can add, edit, and remove supervisor, and additionally can assign TAs to them.

1.1.2 Navigation

The user navigates through the different pages of the TA by using the navigation bar at the top of the page. Quite simply, there are links to the Courses, Teaching Assistants, and Supervisors page.

1.1.3 Filtering and Sorting

In many places throughout the TA Manager, the user is able to sort and filter through the table data. To sort the data, click the column header. To filter the data, the user can choose a column in the table to sort on. Entering text in the adjacent textbox will immediately start filtering the table. The clear button removes the filter.

1.1.4 Dates

Dates (used for start and end terms) appear throughout the application and all take the form of “Term, Year”. Possible terms are “Fall”, “Winter”, or “Summer”. A Fall term lasts roughly from September 1st to December 31st, a Winter term lasts from January 1st to April 30th, and a Summer term lasts from May 1st to August 31st. This means if a course has a start term of “Fall, 2017” and an end term of “Winter, 2018” then the course will run from September 1st 2017 to April 31st. If a course has a start and end term of “Winter, 2019” then the course will run from January 1st 2019 to April 31st 2019.

1.1.5 “Active” Courses and TAs

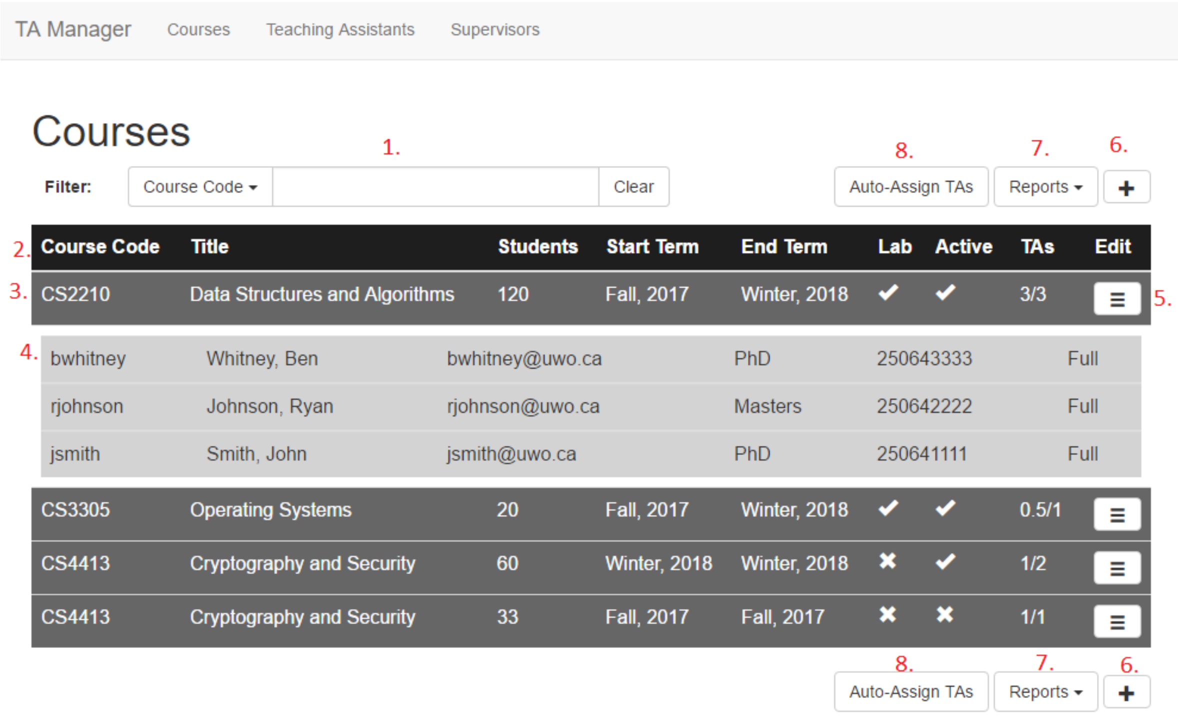
If a course or TA is “active” then that means the course/TA will be targeted when the TA Manager performs actions automatically (chiefly auto-assignment of TAs to courses as well as PDF reporting). The purpose of this field is to omit courses and/or TAs that are no longer relevant. For example, if Bob Smith was a TA that left Western in 2015, then we should set Bob to “inactive” so that he will not appear in PDF reports or be selected as a TA during automatic assignment. Likewise, if a course has finished in 2015, we should set the course to “inactive”. Simply deleting a TA or course instead of deactivating it is *not* recommended since the record of these TAs/courses is used for features such as assignment recommendation.

1.1.6 A Short Note on TAs and Student Type

The student type of a TA determines the (suggested) maximum amount of courses a TA can be assigned to. If a TA had two full course assignments as a Masters student and is switched to a PhD student, then the maximum will increase from 2/3 to 2/8. Additionally, the maximum amount of assignments can be exceeded (but the user will be shown a warning message). As such, it is possible for a TA to have 5/3 full assignments.

1.2 Courses Page

1.2.1 Courses Menu



Course Code	Title	Students	Start Term	End Term	Lab	Active	TAs	Edit
CS2210	Data Structures and Algorithms	120	Fall, 2017	Winter, 2018	✓	✓	3/3	≡
CS3305	Operating Systems	20	Fall, 2017	Winter, 2018	✓	✓	0.5/1	≡
CS4413	Cryptography and Security	60	Winter, 2018	Winter, 2018	✗	✓	1/2	≡
CS4413	Cryptography and Security	33	Fall, 2017	Fall, 2017	✗	✗	1/1	≡

bwhitney	Whitney, Ben	bwhitney@uwo.ca	PhD	250643333	Full
rjohnson	Johnson, Ryan	rjohnson@uwo.ca	Masters	250642222	Full
jsmith	Smith, John	jsmith@uwo.ca	PhD	250641111	Full

Figure 1.2.1 – Courses Menu

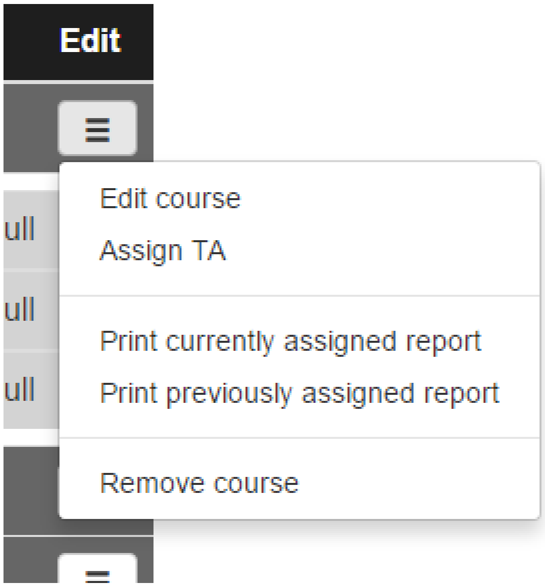
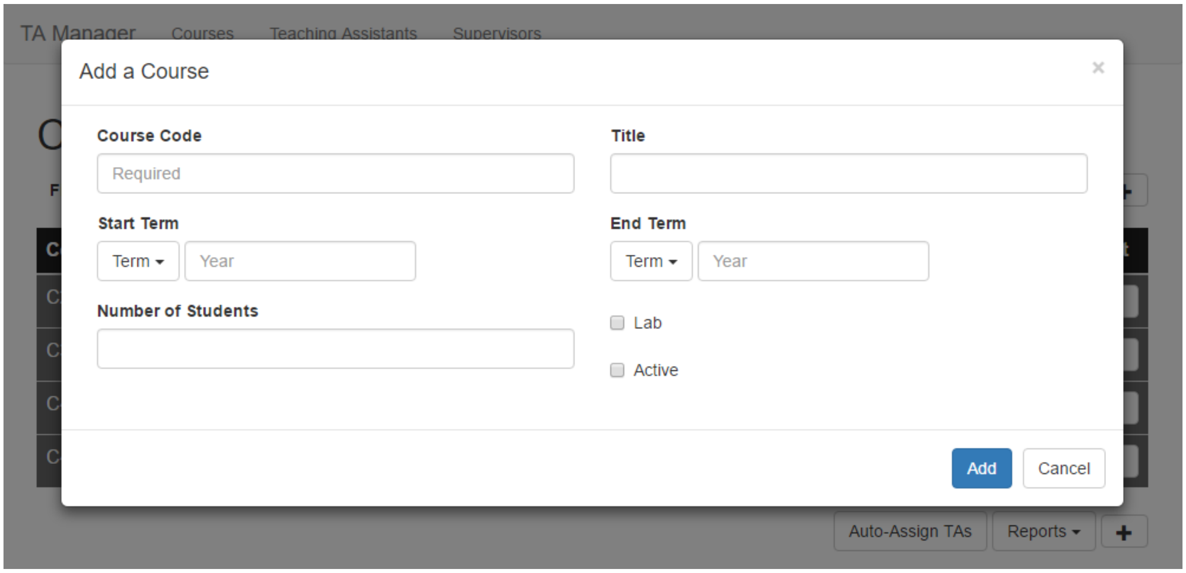

Edit
≡
ull
ull
ull
Edit course
Assign TA
Print currently assigned report
Print previously assigned report
Remove course

Figure 1.2.2 – Course edit menu expanded

1. Filter. Type to search on the selected table column.
2. Table header. Click to sort.
3. A course. Click anywhere on the row to reveal the list of TAs assigned to the course.
4. List of TAs assigned to the course. These TAs are assigned to CS2210 – Data Structures and Algorithms. Contains the user IDs, names, emails, student types, student numbers, and assignment types of the students.
5. Edit button. Opens a list of ways to interact with the selected course as shown in Figure 1.2.2
6. Add course button. Displays a modal to add a course.
7. Reports dropdown. Contains a list of reports for courses in general. Currently contains “print term report”.
8. Auto-Assign TAs button. Clicking this button will generate a potential list of assignments *but will not perform the assignments* unless given confirmation. This feature is described in section 1.2.6.

1.2.2 Add Course Modal



Add a Course

Course Code: Required

Title:

Start Term: Term ▾ Year

End Term: Term ▾ Year

Number of Students:

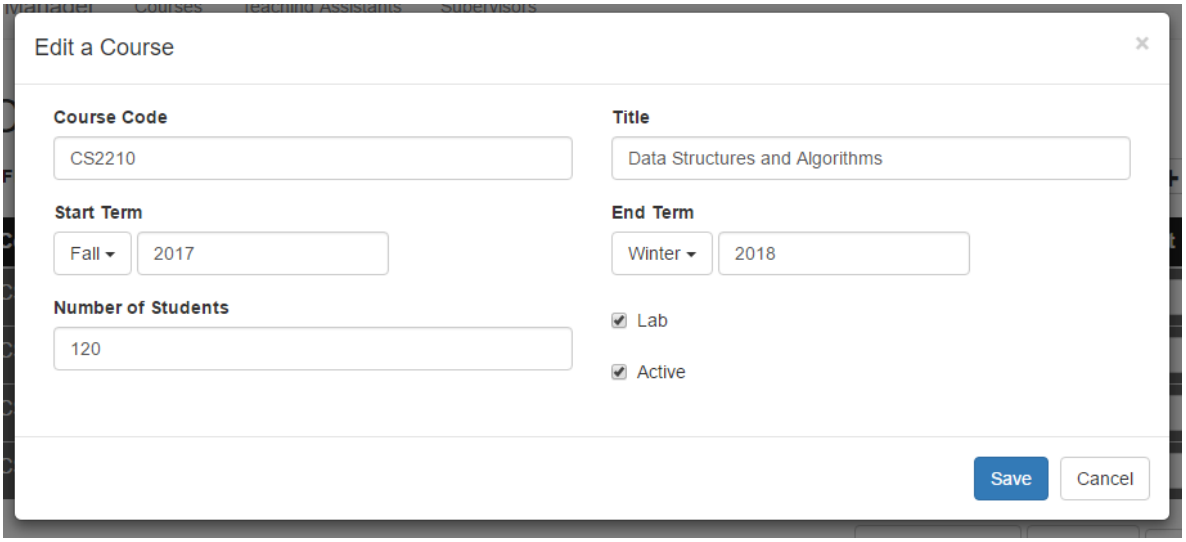
□ Lab

□ Active

Figure 1.2.3 – Add Course Modal

- A course code is required.
- Start and end terms take the form of “Term, Year” such as “Fall, 2017” or “Winter, 2018”. A course that has a start term and end term of “Fall, 2017” means that the course will run approximately from September 1st 2017 to December 31st 2017.

1.2.3 Edit Course Modal



Edit a Course

Course Code: CS2210

Title: Data Structures and Algorithms

Start Term: Fall 2017

End Term: Winter 2018

Number of Students: 120

☑ Lab

☑ Active

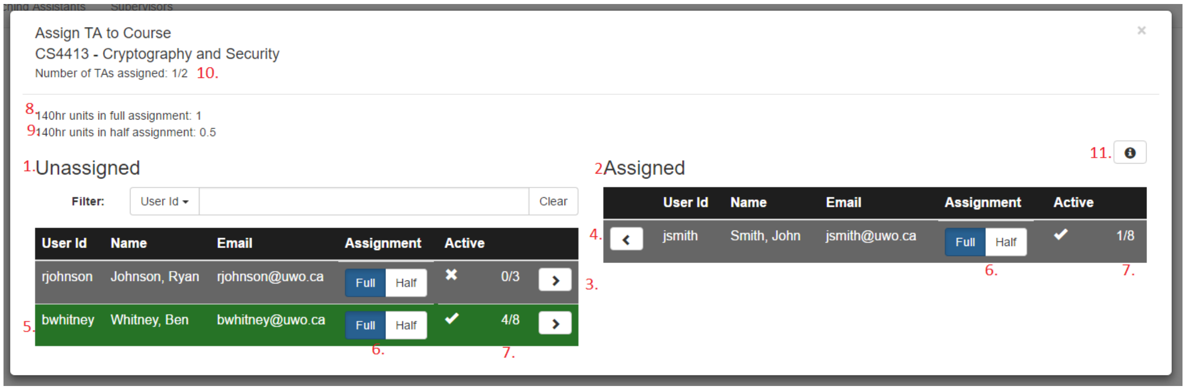
Figure 1.2.4 – Edit Course Modal

- Edits a course.
- Dates follow the same rules as the add course modal.

1.2.4 Remove Course Modal

- Accessed by using the edit button on a course and selecting “remove course”.
- Asks the user to confirm or cancel the deletion of the course.

1.2.5 Assign TA Modal



Assign TA to Course
CS4413 - Cryptography and Security
Number of TAs assigned: 1/2

140hr units in full assignment: 1
140hr units in half assignment: 0.5

Unassigned

Filter: User Id

User Id	Name	Email	Assignment	Active		
rjohnson	Johnson, Ryan	rjohnson@uwo.ca	Full Half	✗	0/3	>
bwhitney	Whitney, Ben	bwhitney@uwo.ca	Full Half	✓	4/8	>

Assigned

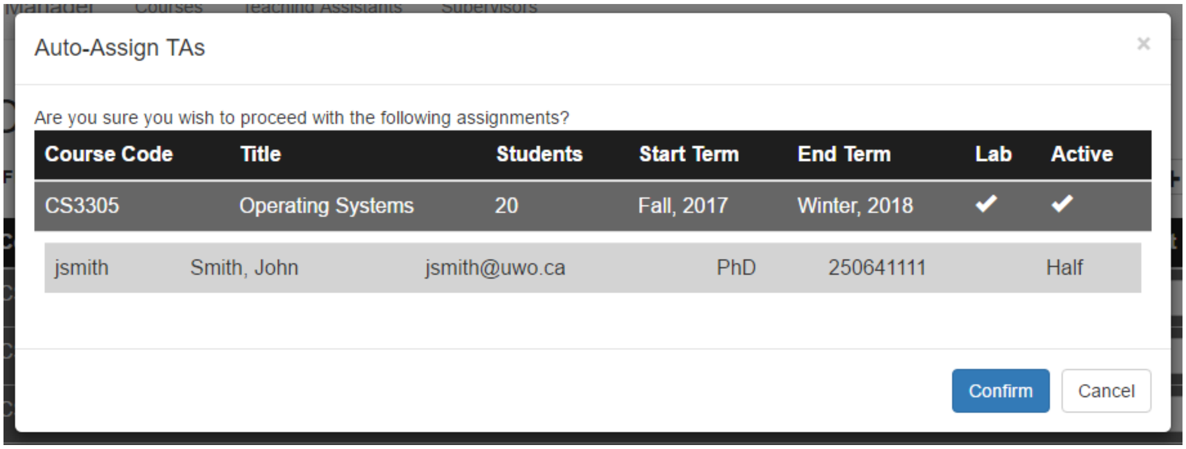
	User Id	Name	Email	Assignment	Active	
<	jsmith	Smith, John	jsmith@uwo.ca	Full Half	✓	1/8

Figure 1.2.5 – Course Assign TA Modal

1. Unassigned TA list.
2. Assigned TA list.

3. Assign TA to course. There are a number of “soft” restrictions on assigning TAs to courses. They soft in the sense that if a restriction is met, a warning will appear to the user asking them if they want to proceed with the assignment or not. A full list of restrictions is found in section 5.1.
4. Unassign TA. There are no restrictions on unassigning a TA from a course.
5. Recommended assignment. This TA is highlighted in green because they have taught a course with the course code “CS4413” previously.
6. Assignment type toggle. Toggles whether the TA will be assigned full-time (140hrs) or half-time (70hrs) to a course.
7. The fraction of how many 140hr units a TA has taught and their maximum depending on whether they are Masters or PhD students. A Masters student can have three 140hr units while a PhD student can have eight.
8. Amount of 140hr terms in a full assignment.
9. Amount of 140hr units in a half assignment.
10. Number of TAs assigned. A half assignment counts as half a TA.
11. Information button. Contains basic assignment rules and a shorter description of the information held in this section of the user manual.

1.2.6 Auto-Assign TA Modal



Are you sure you wish to proceed with the following assignments?

Course Code	Title	Students	Start Term	End Term	Lab	Active
CS3305	Operating Systems	20	Fall, 2017	Winter, 2018	✓	✓

jsmith	Smith, John	jsmith@uwo.ca	PhD	250641111	Half

Figure 1.2.6 – Auto-Assign TA Modal

- Displays a list of course-TA assignments generated by the auto-assignment algorithm.
- Goes through all active courses/TAs and makes assignment pairs while considering assignment rules and prioritizing TAs who have previously taught a course.
- Clicking the confirm button will carry out the assignments displayed.
- Clicking the cancel button will not apply the assignments.

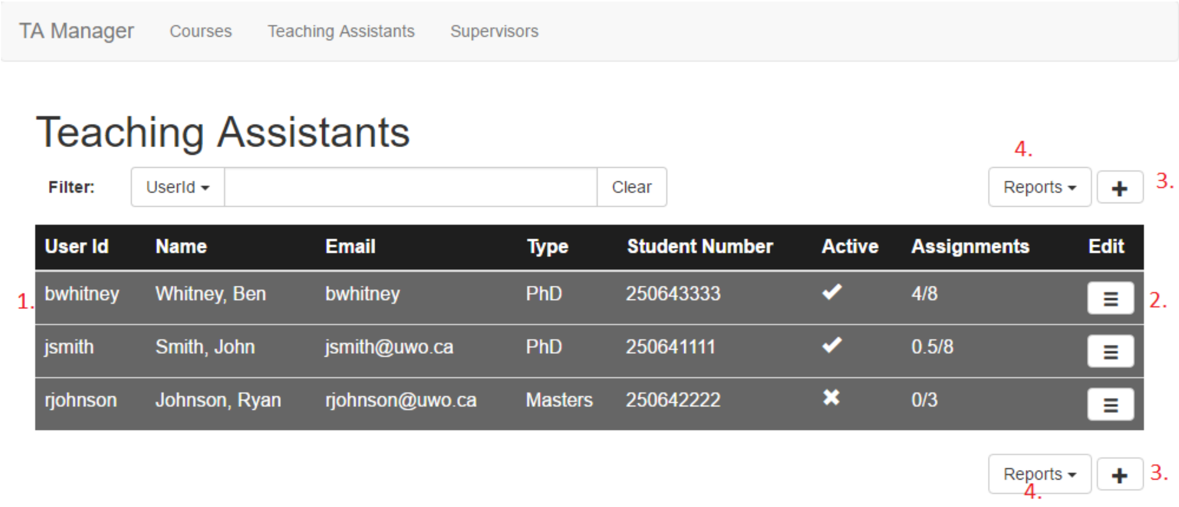
- After confirmation, the user can assign/unassign as normal.

1.2.7 Reports

- Term report: Generates a report for all courses between a user-given term range.
- Currently assigned report: Generates a report for a given course containing a list of TAs that are currently assigned.
- Previously assigned report: Generates a report for a given course containing a list of TAs that were previously assigned.

1.3 Teaching Assistants Page

1.3.1 Teaching Assistants Menu



User Id	Name	Email	Type	Student Number	Active	Assignments	Edit
bwhitney	Whitney, Ben	bwhitney	PhD	250643333	✓	4/8	≡
jsmith	Smith, John	jsmith@uwo.ca	PhD	250641111	✓	0.5/8	≡
rjohnson	Johnson, Ryan	rjohnson@uwo.ca	Masters	250642222	✖	0/3	≡

Figure 1.3.1 – Teaching Assistants Menu

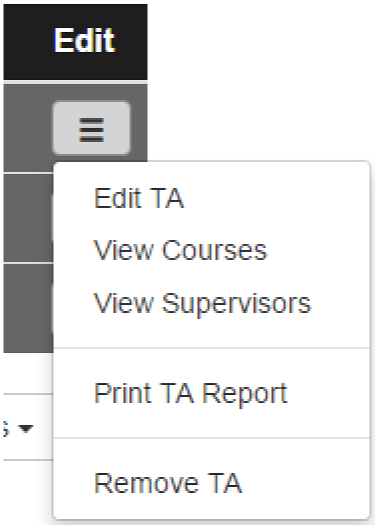


Figure 1.3.2 – Teaching Assistants edit menu expanded

1. TAs.
2. Edit button. Shows the menu in Figure 1.3.4
3. Add TA button. Displays the modal to add TAs.
4. Reports dropdown. Contains a list of reports for TAs in general. Currently contains "print TAs report".

1.3.2 Add TA Modal

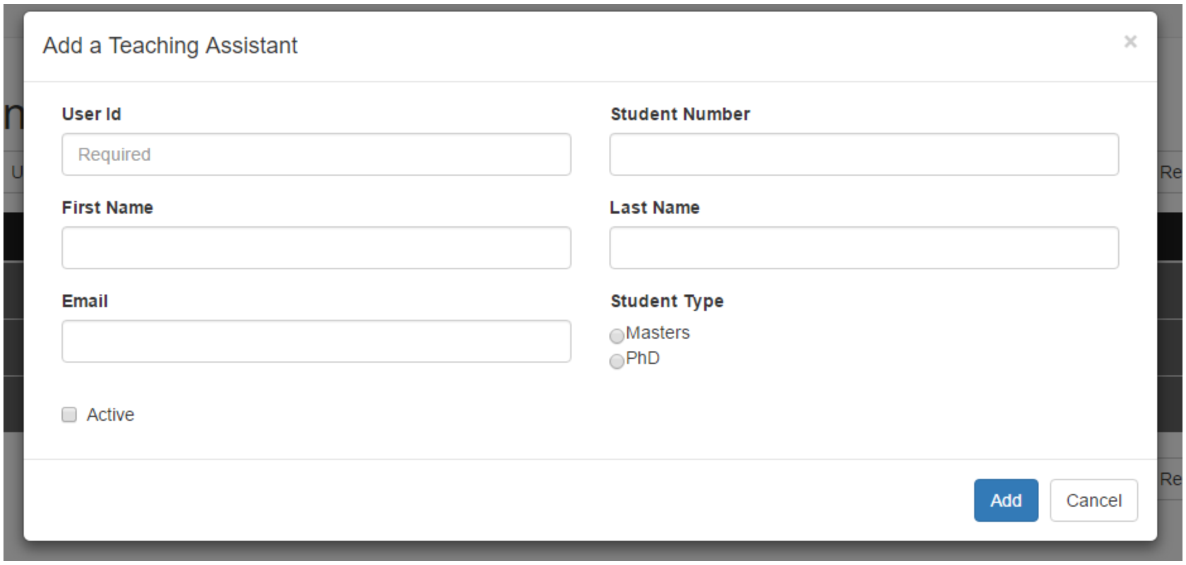
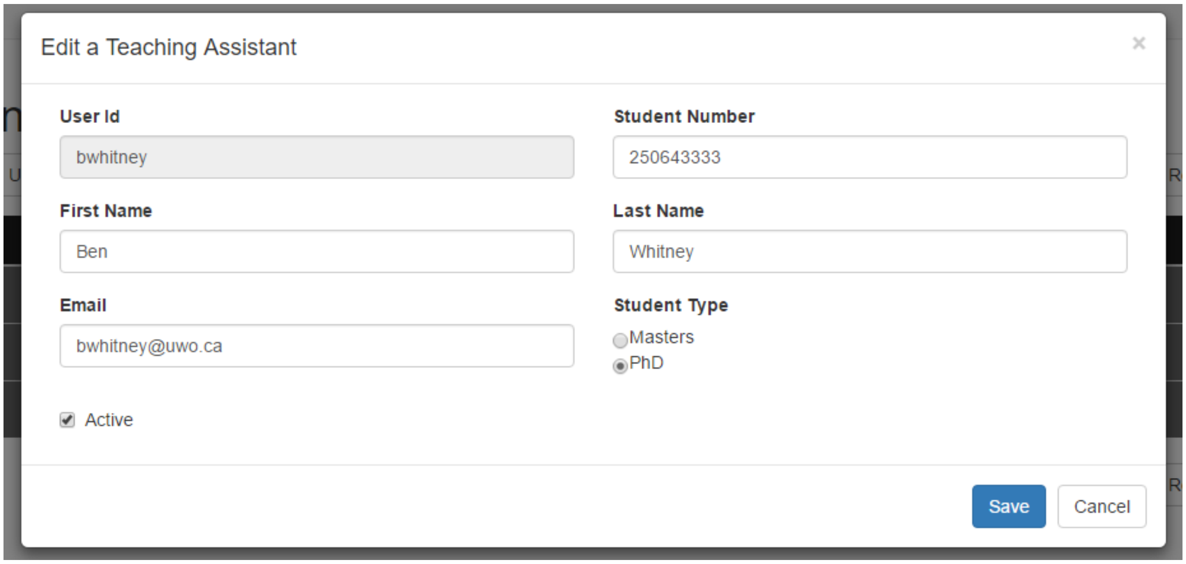


Figure 1.3.3 – Add TA Modal

- A user ID is required and may not be changed after creation. The rest of the fields are optional.
- The student type determines how many 140hr assignments the TA may have. A Masters student may have three 140hr assignments while a PhD may have eight.

1.3.3 Edit TA Modal



User Id: bwhitney

Student Number: 250643333

First Name: Ben

Last Name: Whitney

Email: bwhitney@uwo.ca

Student Type: ○ Masters ● PhD

☑ Active

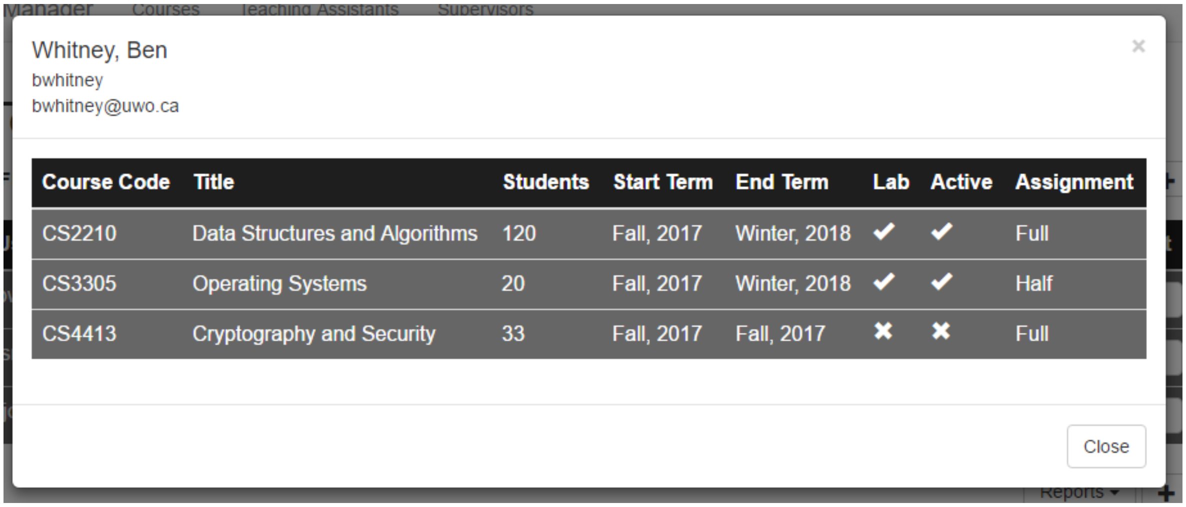
Figure 1.3.4 – Edit TA Modal

- Edits a TA.
- User ID cannot be changed.

1.3.4 Remove TA Modal

- Asks the user to confirm or cancel the deletion of the course.

1.3.5 View Courses Modal



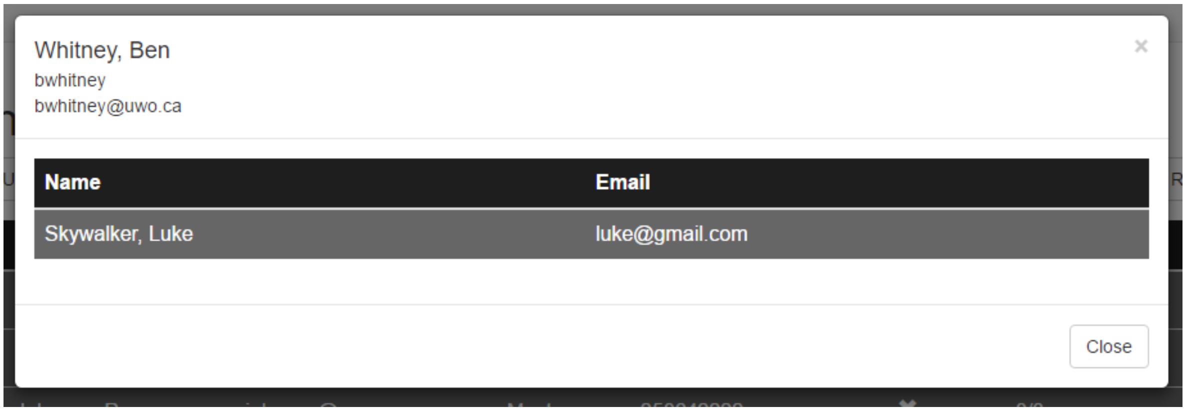
Whitney, Ben
bwhitney
bwhitney@uwo.ca

Course Code	Title	Students	Start Term	End Term	Lab	Active	Assignment
CS2210	Data Structures and Algorithms	120	Fall, 2017	Winter, 2018	✔	✔	Full
CS3305	Operating Systems	20	Fall, 2017	Winter, 2018	✔	✔	Half
CS4413	Cryptography and Security	33	Fall, 2017	Fall, 2017	✖	✖	Full

Figure 1.3.5 – TA View Courses Modal

- Shows courses the TA is assigned to.

1.3.6 View Supervisors Modal



Whitney, Ben
bwhitney
bwhitney@uwo.ca

Name	Email
Skywalker, Luke	luke@gmail.com

Figure 1.3.6 – TA View Supervisors Modal

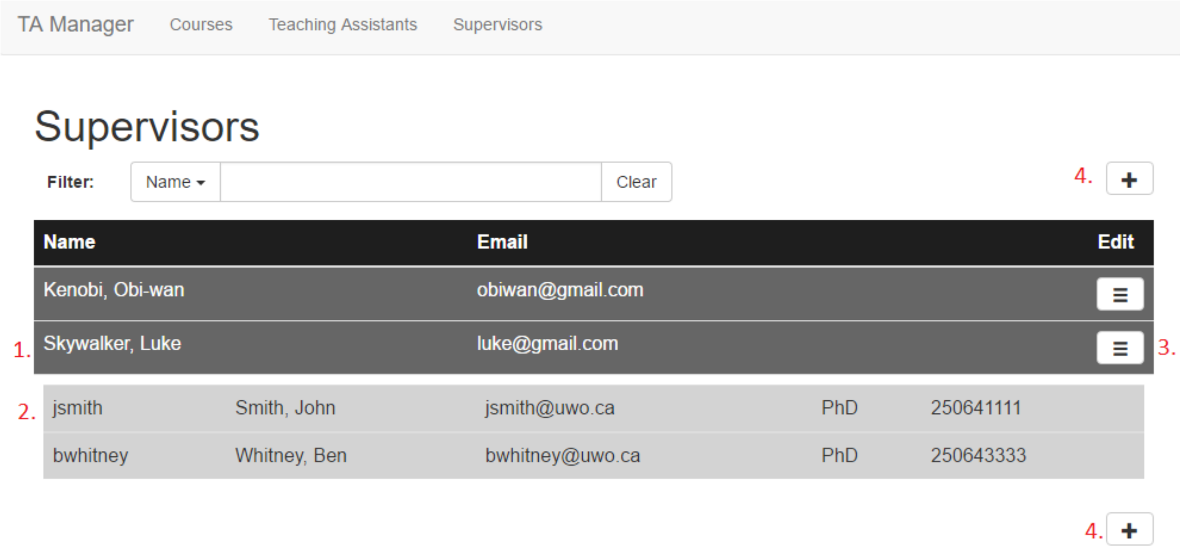
- Shows supervisors the TA is assigned to.

1.3.7 Reports

- Print TAs report: generates a report containing data with all active TAs.
- Print TA report: generates a report containing data for the selected TA.

1.4 Supervisors Page

1.4.1 Supervisors Menu



Name	Email	Edit
Kenobi, Obi-wan	obiwan@gmail.com	☰
Skywalker, Luke	luke@gmail.com	☰

jsmith	Smith, John	jsmith@uwo.ca	PhD	250641111
bwhitney	Whitney, Ben	bwhitney@uwo.ca	PhD	250643333

Figure 1.4.1 – Supervisors Menu

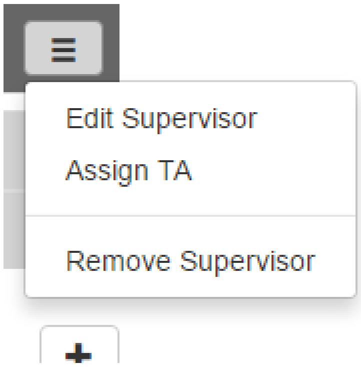


Figure 1.4.2 – Supervisor edit button expanded

1. A supervisor. Click anywhere on the row to show a list of TAs assigned to the supervisor.
2. List of TAs assigned to the supervisor. These TAs are assigned to Luke Skywalker.

3. Edit button. Displays the menu shown in Figure 1.4.4.
4. Add supervisor button. Displays the modal to add a supervisor.

1.4.1 Add Supervisor Modal

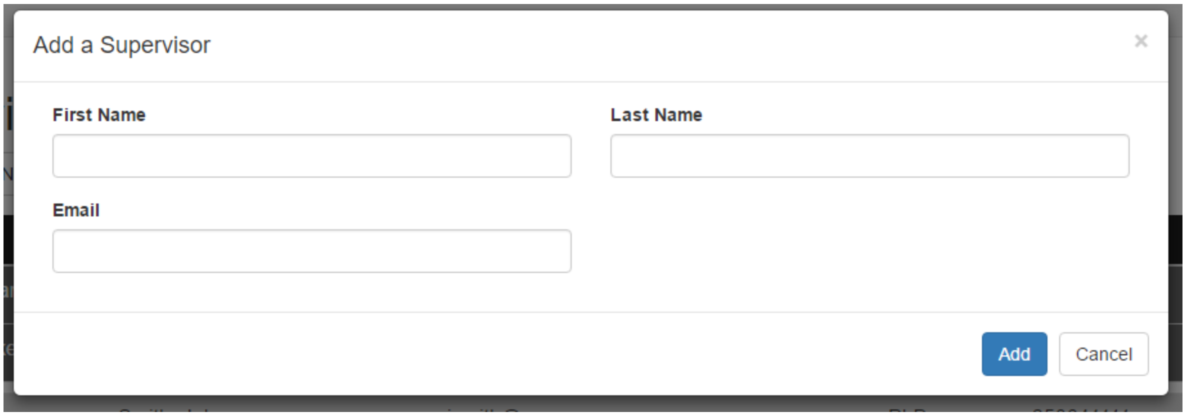
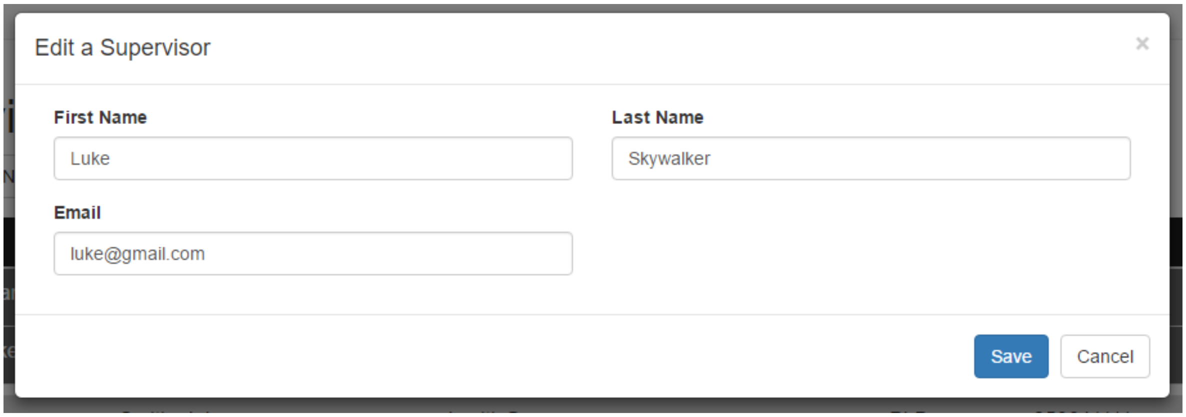


Figure 1.4.3 – Add Supervisor Modal

- Adds a supervisor.

1.4.2 Edit Supervisor Modal


Edit a Supervisor

First Name
Luke

Last Name
Skywalker

Email
luke@gmail.com

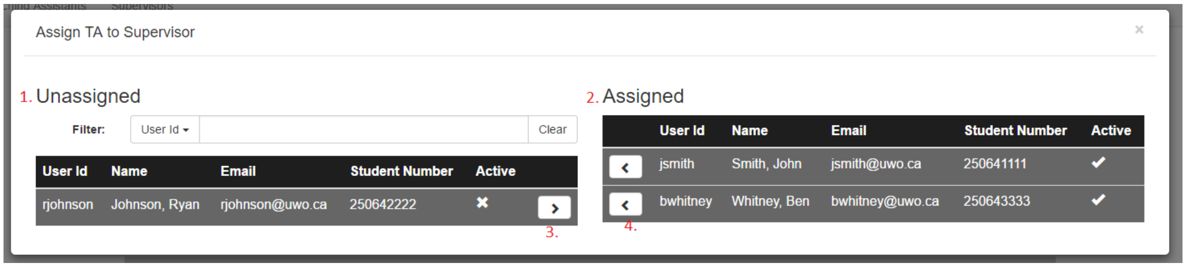
Figure 1.4.4 – Edit Supervisor Modal

- Edits a supervisor.

1.4.3 Remove Supervisor Modal

- Asks the user to confirm or cancel the deletion of the course.

1.4.4 Assign TA Modal



Assign TA to Supervisor

1. Unassigned

Filter: User Id ▾ Clear

User Id	Name	Email	Student Number	Active	
rjohnson	Johnson, Ryan	rjohnson@uwo.ca	250642222	✖	>

2. Assigned

	User Id	Name	Email	Student Number	Active
<	jsmith	Smith, John	jsmith@uwo.ca	250641111	✔
<	bwhitney	Whitney, Ben	bwhitney@uwo.ca	250643333	✔

Figure 1.4.5 – Supervisor Assign TA Modal

1. Unassigned TA list.
2. Assigned TA list.
3. Click to assign the TA to the supervisor.
4. Click to Unassign the TA from the supervisor.

Chapter 2. Technology Used

For the TA Manager, one of my main goals was to use the least amount of libraries as possible. The overzealous use of libraries can greatly reduce the quality of code and the ease of maintenance and extension. With more libraries, the knowledge required to maintain and add features to the application increases. As well, sometimes libraries have their own bugs associated with them or they conflict with other libraries. It is important to have a substantial reason to introduce a new library.

2.1 Server:

- **Node.js**: the JavaScript execution platform that handles all server requests.
- **Express**: a framework for Node.js. By itself, Node does not come with easy methods of handling GET and POST requests. Instead of re-inventing the wheel, the TA Manager uses Express to handle these requests.
- **PostgreSQL**: an open-source database that uses a SQL-like syntax.
  - **pgAdmin 4**: the primary tool for managing the PostgreSQL database. This is comparable to software like Microsoft’s SQL Server Management Studio.
  - **pg-promise**: this is used to connect the server created with Node.js to the PostgreSQL database. With pg-promise our server can easily run queries and receive data from the database.
- **body-parser**: this is a simple library to more easily extract the body data of a POST request. This functionality is not present in Node.js or Express without writing your own parser.
- **Passport.js**: handles all server-side authentication.

2.2 Client:

- **Bootstrap**: this library provides CSS that guarantees a clean and consistent look throughout the website. Some of the core features are the grid system used for creating screen-scalable UI, and an easy-to-use modal system.
- **Knockout**: this enforces the MVVM design pattern. Variables can be bound to the UI such that changes view-model will immediately update the view and vice-versa. Knockout also makes it very easy to make tables from arrays of objects (which are used extensively throughout the application) by providing tools that automatically generate HTML for every item in the array.

- **jQuery**: this library was reluctantly added to the TA Manager app because the modal system provided by Bootstrap requires jQuery to run effectively. The main problem with including jQuery is that it has a ton of features that are unnecessary for the TA Manager and can easily break Knockout's MVVM implementation. In order to keep use of jQuery to an absolute minimum, jQuery will only be used to operate the Bootstrap modal functions.
- **AJAX**: this technology is used by the TA Manager to send asynchronous GET and POST requests to the Node.js server. Obviously, this provides a slew of benefits such as not having to reload a page when a request to the server is made.
- **pdfmake**: library used to easily create PDF documents. It is a declarative-based PDF generator, meaning that values are simply assigned in a JavaScript object and pdfmake is executed to automatically generate the PDF.

Chapter 3. Site Structure

3.1 File Layout

- config
  - database.js: contains database configuration settings
- node_modules: contains node packages
- taManager.js: the node.js server-side app code
- public: folder holding all files accessible to the client
  - css: contains css files
    - modal.css
    - table.css
  - js: holds JavaScript files
    - courses.js: viewmodel for Course page
    - date-tools.js: collection of date tools used across the application
    - pdf-tools.js: collection of pdf tools used across the application
    - supervisors.js: viewmodel for Supervisor page
    - table-tools.js: collection of table tools used across the application
    - teaching-assistants.js: viewmodel for Teaching Assistant page
  - libraries
    - bootstrap-3.3.7-dist
    - jquery-3.1.1
    - knockout-3.4.1
    - pdfmake
  - courses.html: view for Course page
  - navbar.html
  - supervisors.html: view for Supervisor page
  - teaching-assistants.html: view for Teaching Assistant page

3.2 Structure Explanation

The files in the public folder are accessible to the client. From here, the client sends AJAX requests to the node server (taManager.js) and receives the responses.

The files date-tools.js, pdf-tools.js, and table-tools.js are used across the system. The goal of these files is reduce code duplication for common functionality across the TA Manager.

One thing that I would like to have changed is to use require.js to manage loading JS files and have proper dependency tracking. The TA Manager instead includes files by including them in the header of the html file.

Chapter 4. Database

4.1 ER Diagram

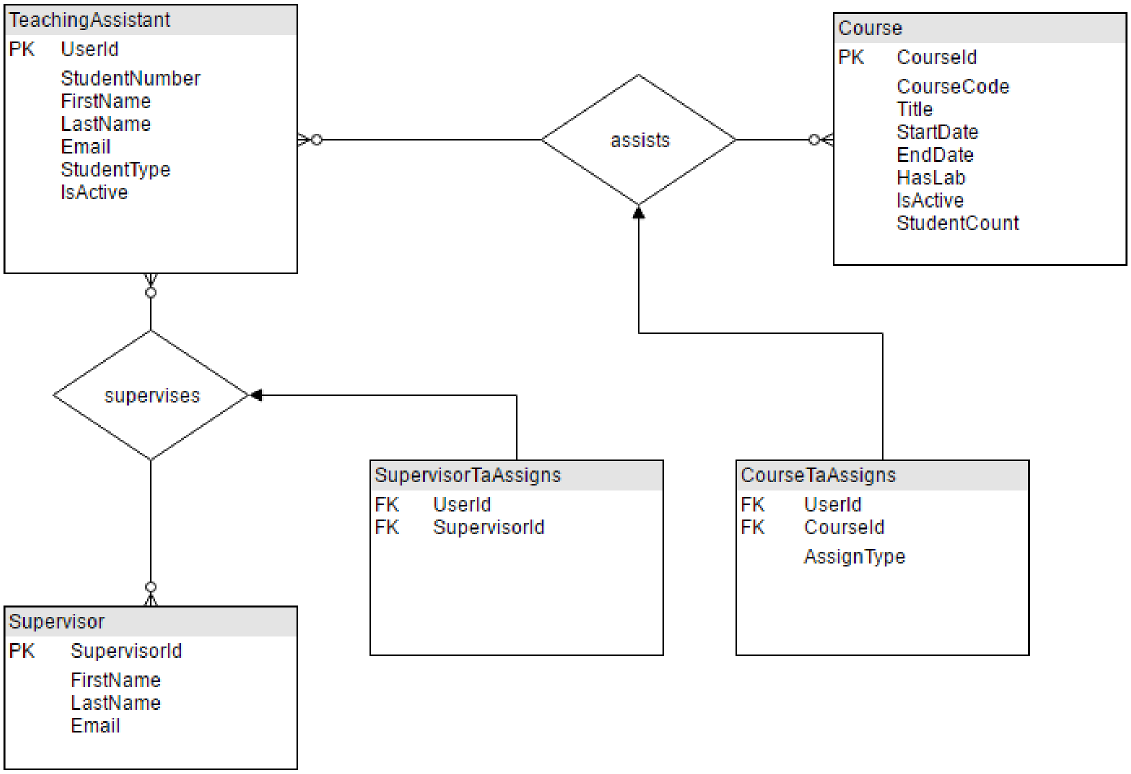
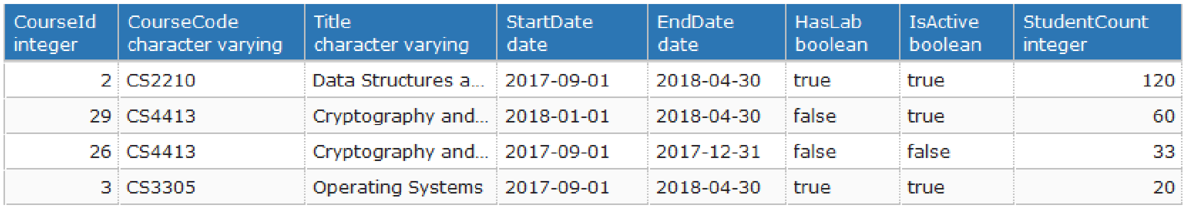


Figure 4.1 – ER Diagram for TA Manager

4.2 Tables

The database for the TA Manager was implemented with PostgreSQL. The following section contains sample date for each table in the PostgreSQL database complete with the column names and column data types.

4.2.1 Course

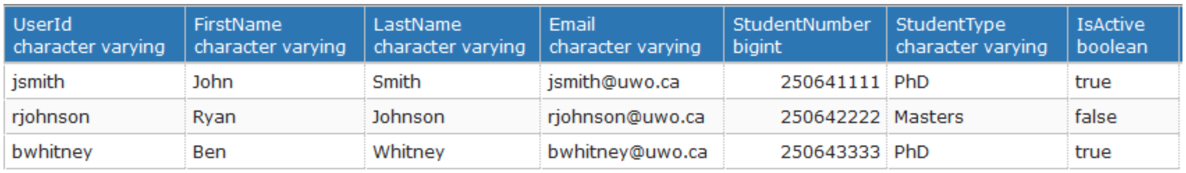


CourseId integer	CourseCode character varying	Title character varying	StartDate date	EndDate date	HasLab boolean	IsActive boolean	StudentCount integer
2	CS2210	Data Structures a...	2017-09-01	2018-04-30	true	true	120
29	CS4413	Cryptography and...	2018-01-01	2018-04-30	false	true	60
26	CS4413	Cryptography and...	2017-09-01	2017-12-31	false	false	33
3	CS3305	Operating Systems	2017-09-01	2018-04-30	true	true	20

Figure 4.2 – Sample data for Course table

- CourseId is used to tell different courses apart even if they have the same CourseCode. The CourseCode is used to determine TAs who have taught the course previously. For example, if a TA teaches CS4413 they will be suggested as a recommended TA for all courses with the CourseCode CS4413.
- Dates are stored as PostgreSQL date objects. When these objects are retrieved from the database, they become date strings. These date strings are then converted into various other date formats such as JavaScript date objects and term/year pairs. The advantage of storing date values as dates instead of strings is that they can use certain operators that do not make sense for strings in this scenario. For example, if we need to check if dateA is earlier than dateB, we can use dateA < dateB, whereas stringA < stringB would give a different result.
- It is important to note that the number of assignments and maximum number of assignments for the course is *not* stored in the database. It is computed from several other columns and as such is computed when the Course is loaded.

4.2.2 TeachingAssistant

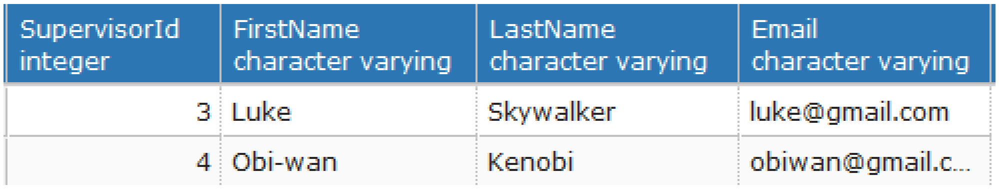


UserId character varying	FirstName character varying	LastName character varying	Email character varying	StudentNumber bigint	StudentType character varying	IsActive boolean
jsmith	John	Smith	jsmith@uwo.ca	250641111	PhD	true
rjohnson	Ryan	Johnson	rjohnson@uwo.ca	250642222	Masters	false
bwhitney	Ben	Whitney	bwhitney@uwo.ca	250643333	PhD	true

Figure 4.3 – Sample data for TeachingAssistant table

- The StudentType must be either the string ‘PhD’ or ‘Masters’. This is ensured in the server-side calls to the database.
- It is important to note that the number of assignments and maximum number of assignments for the TA is *not* stored in the database. It is computed from several other columns and as such is computed when the TA data is loaded.

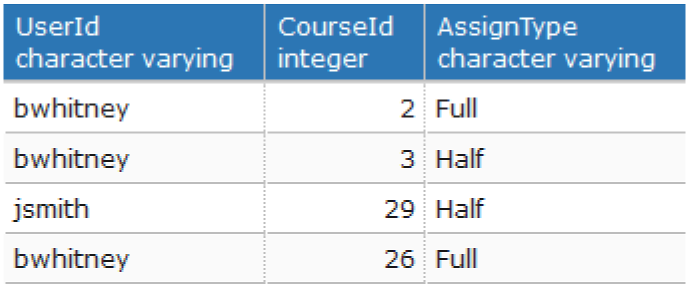
4.2.3 Supervisor



SupervisorId integer	FirstName character varying	LastName character varying	Email character varying
3	Luke	Skywalker	luke@gmail.com
4	Obi-wan	Kenobi	obiwan@gmail.c...

Figure 4.4 – Sample data for Supervisor table

4.2.4 CourseTaAssigns

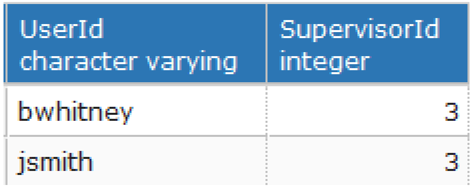


UserId character varying	CourseId integer	AssignType character varying
bwhitney	2	Full
bwhitney	3	Half
jsmith	29	Half
bwhitney	26	Full

Figure 4.5 – Sample data for CourseTaAssigns table

- A constraint exists to ensure that the UserId/CourseId pairs are unique.

4.2.5 SupervisorTaAssigns



UserId character varying	SupervisorId integer
bwhitney	3
jsmith	3

Figure 4.6 – Sample data for SupervisorTaAssigns table

- A constraint exists to ensure that each row is unique.

Chapter 5. Course-TA Assignment & the Auto-Assignment Algorithm

The Auto-Assignment algorithm is designed to provide a list of course and TA pairs from a pool of active courses and TAs. To be specific, this algorithm generates the list of TA-Course pairs but does *not* carry out the assignments. The list that results from the auto-assignment algorithm is processed after the user confirms the assignments. After confirming, the user is freely available to make assignments manually as usual.

The algorithm makes pairs prioritizing TAs that have taught a course before. The algorithm must take into account the many restrictions for assigning courses and TAs. Assignments may be “full” assignments (140hrs per term) or “half” assignments (70hrs per term). As is expected, one full assignment is worth two half assignments.

5.1 Course-TA Assignment Restrictions

- TAs have a maximum amount of full assignments they can take. If the TA is a Masters student, the TA may have three 140hr/term assignments. If the TA is a PhD student, the TA may have eight 140hr/term assignments. For example, if a course lasts two terms and a TA is assigned full-time, then that counts as two 140hr/term assignments. If the TA was assigned half-time, then that counts as one 140hr/term assignment.
- TAs may only serve 140hrs in a term.
- Courses may have up to $\lceil num\ of\ students\ in\ course/50 \rceil$ TAs that are assigned full-time. As usual, a half-time assignment counts as half a full-time assignment.
- A TA can only be assigned to a course with CourseId once (i.e. all UserId/CourseId pairs in the CourseTaAssigns table must be unique). It is important to make a distinction between CourseId (the primary key, integer such as “59”) and CourseCode (just a string field such as “CS1000”).

5.2 Algorithm Description

Data must be retrieved from the database. First, we retrieve all active TAs and courses from the database. Then we get a list of all active TAs with courses they have taught before by the CourseCode. Finally, we get a list of all CourseTaAssigns joined on the Course and TeachingAssistant tables that we call an assignment schedule.

Now we process these lists in JavaScript and compute the output assignment list. We need to compute the current number of assignments and maximum number of assignments for both the active courses and the active TAs. This is done by using the assignment schedule and looking at how long each assignment was, as well as the type of assignment (full or half). At this point, we sort the list of active courses based on the amount of terms descending. This makes it so that the largest assignments (the ones that are harder to fill) are attempted to be made first in the following steps. We then filter the active TAs and courses lists if they have no space left for assignments.

Then, we loop through the list of active courses in order to try to make assignments where a student has previously taught the course. For each course, we get the sub-list of TAs that have taught the course with the same CourseCode. We the find the largest possible assignment (either full, half, or none) for this course and every TA in the list until we have either gone through all the courses or the TA list is empty. This value is found by using the course, TA, and assignment schedule with the restrictions described above in section 5.1. We take this largest possible assignment and put it in the output object.

Then we do the remainder of the TAs regardless of them being previously assigned. The largest assignment value is put in the output object and sent to the client. When the user on the client confirms to proceed with the assignments, the list of assignments is processed.

Conclusions

The TA Manager contains a plethora of features and makes the job of assigning teaching assistants significantly easier than earlier processes. It contains assignment restriction checking functionality, as well as automatic assignment. It also can print PDF reports and give recommended assignments. The UI makes it easy to monitor how much a TA or course can be assigned.

At the time of this writing, the TA Manager is not currently uploaded, but will be uploaded soon. It has completed all development (aside from some minor tasks that must be done at deployment) and is tested thoroughly in an offline environment. I look forward to seeing it online.