User Guide

# TA Allocation Scheduler

## Team Members

**REDACTED**

**REDACTED**

Brandon Racich

**REDACTED**

**REDACTED**

**Table of Contents**

[**TA Allocation Scheduler**](#_30j0zll) **1**

[Team Members](#_frsf3e5vdxvq) 1

[**Overview**](#_1fob9te) **3**

[Document Overview](#_7ghe90u75fhn) 3

[**User Help**](#_v8psw680c23) **3**

[General](#_usixte8a0my0) 3

[Login](#_89wlepb18pf3) 3

[Recover Password](#_1ypqo5x797ja) 3

[Home](#_gl1xl1r23x0s) 3

[Logout](#_299mnfusgy7o) 3

[Students](#_kcdubhx218qn) 3

[Create Account](#_ny1kx226fsb) 3

[Create Application](#_bgs2i8jv4nln) 3

[View Account Info](#_b0gpx22lhq54) 4

[Edit Account Info](#_kw2xbip8cqum) 4

[Administrators](#_k3tkq0duxyme) 4

[List TAs](#_x7hl801p2ihs) 4

[Course List](#_2qcpmus5ek70) 4

[View Applications](#_o4nq9vl8w2ky) 4

[Upload Class Schedule](#_ochwc8dk6ppx) 4

[Other admin functions](#_rx446ewv8je1) 5

[Where to find the source code](#_qvxcwyd9b02) 5

[**Installation**](#_nn0rujw74xkj) **5**

[Server Configuration](#_l2z7h6ut1xy8) 5

[Web Development Stack](#_mc6xda2ql4zj) 5

[File Structure](#_i7s3ey6koo1w) 5

[Public\_html - MVC](#_bb8hc6unsx6p) 6

[View](#_cxuz8puz2nqo) 6

[Controller](#_5s8fmw99ihwc) 6

[Model](#_1t05id5shcy6) 6

[LIB](#_eq0ko1tyw8xk) 6

[Examples](#_lk9ph18vxc1q) 6

[fpdf](#_39z3nz1vtsxr) 6

[PHPMailer](#_v2tyexc3zm1) 6

[assignmentResults.csv](#_7po11y7r01k4) 6

[Database.class.php](#_8cp6qtpmadi) 6

[IP.py](#_n3ig36kiu6f9) 6

[markingResults.csv](#_tmb595hck31g) 6

[statusCode.ini](#_rl1d65dlzola) 7

[Required Permissions](#_f5oddg80ur7o) 7

[Database Configuration](#_vl4gyupedgcb) 8

[ta\_portal Tables](#_iasi2h1fm8l7) 8

[Administrator](#_v1cripn2c0b7) 8

[Application](#_rtwqdarwoo6r) 8

[Course](#_cqf3cl6sgy9w) 9

[CourseInstructor](#_ec4zbd5waarf) 9

[Professor](#_tnx3c7smnxyr) 9

[Schedule](#_omjm8rywdbkx) 9

[Section](#_apypm4j6z722) 9

[SectionTA](#_d8rigedsk1kx) 9

[TA](#_8g6xxmi3tof) 9

[UserAccount](#_m0yxi1i18oys) 9

[**Deployment**](#_w0ms5r5sbb73) **9**

[Continuous Integration](#_g5kqvmrg9m38) 9

[Continuous Deployment](#_947yx8w865yz) 9

[**Known Issues**](#_26ea4lwrlufk) **9**

# Overview

## Document Overview

This document contains a User Help guide that covers the functionality that Students and Administrators can carry out on the web application, as well as a installation and deployment guide covering the setup and operations of the software at a high level.

# User Help

## General

### Login

Admin user account is username = “Admin” and password = “pass”. Admins or students may log in from the login page.

### Recover Password

Admins or TAs can enter their username and email to reset their password.

### Home

Takes admins to the admin portal page, and TAs to the TA portal page.

### Logout

Returns the user to the TA Allocator Home page.

## Students

### Create Account

Students can create an account from the login page by clicking on “create one”. After entering their first name, last name, username, email, student number, and password, they click on create and the account is generated.

### Create Application

Application is where you fill out your student information such as session, program, preferred hours, maximum hours, Transcript, and Schedule.

### View Account Info

Clicking on view Account brings you to the ta-single page where you can see the your previous course history and grades, currently assigned courses, previous TA Assignments, and 4 types of hours. The print report button will print out the page and the edit info button will take you to Edit Account Info function.

### Edit Account Info

Edit account info allows the student to edit their account info by changing their email, first name and last name, and finally, their password.

## Administrators

### List TAs

Clicking on List TAs will display all currently accepted UTAs and GTAs. A drop down box will update the page with a list of either summer session TAs or winter session TAs depending on the admins choice. The ability to update the minimum and maximum hours for a TA exists with the Save Changes button and checking any boxes in the Email Offer field will send an offer to that specific TA when the Email Offer(s) button is clicked.

### Course List

The page Course List contains some of the most important features of the TA allocator website and thus has the most functions. This page displays all courses TAs can be allocated to. The filter options can list courses by program and year, session filter by summer or winter session, and term filter by term 1 or 2. Next from left to right, we’ll go over the a course row and explain the functions. A course has many options so starting from the left, a button labelled “More Detailed Info” will take the admin to the course single page where they can see which TAs have been added to the course. Inside this course single page, the admin can update each TAs prep, mark, and other hours. Back on the Course List page, we next under the   
“More Detailed Info” button, we see a Set Restrictions option. Inside a simple form can update a courses restrictions such as labHours, markingHours, minCredits etc… Next we have expand button labelled with the plus sign above the block column which will expand the list of all sections for a given course. Under the expand button inside the Block column, we have a checkbox which will prevent the given section from being included in the optimization schedule. The TA Info column allows the admin to manually select accepted TAs into that given section. Once selected from the dropdown box, clicking the Add button will add that TA to the section. Furthest on the right we have the Scheduled TA column which will list the TA currently assigned to that section (given that one has been already assigned). Finally, we have our optimization function in the top right of the page labelled “Automated Scheduler”. It provides automated scheduling with customizable time limits for length of time the scheduler runs. A time limit of ~5 minutes (300 seconds) will result in a respectable feasible schedule while ~4 hours will result in an excellent schedule where undesirable situations such as hiring individuals with no experience is minimized. UTA and GTA hour budgets may also be set prior to running the Automated Scheduler.

### View Applications

View applications let the admin see all applications submitted. The Status column will allow the admin to Accept or Decline offers. The Schedule column and Transcript column allow the admin to view the submitted TAs Schedule and Transcript respectively. TA usernames are clickable and link to the TAs profile page.

### Upload Class Schedule

This function allows the administrator to upload classes to the system from a CSV format downloaded from UBC SISC. Uploaded CSV files are validated based on presence of required headers fields. Upon successful upload, the administrator is redirected to a page where the hour settings for each course can be defined and sections can be marked as blocked (by default, a course with no lab sections will be blocked). The course section uploader operates on a create if not exists otherwise update basis and therefore can be used to update section data should their attributes change in UBC’s system. Removal of any sections must be done manually through the add/remove course section tool.

### Other admin functions

Create Admin Account: A simple form to create new Admin Accounts.

Edit TA Account: Simple form to edit a TAs username, first name, last name, email, and graduation status.

Add/Remove Course Section: This function allows the admin to add and delete courses, as well as adding or deleting sections.

### 

### Where to find the source code

**REDACTED**

# Installation

## Server Configuration

Below is an overview of the configuration used during development of the project. The project folder was served under /var/www/ta-scheduler/ in our server environment. /var/www/ta-scheduler/public\_html/ is the intended folders to be served to the public with anything outside of public\_html being referenced libraries or configuration files.

### Web Development Stack

The server used to host this web application was configured with a LAMP stack. The server runs:

* Ubuntu 18.04
* Apache Web Server 2.4.29
* MySQL 5.7.25
* PHP 7.2.15

External PHP Libraries used:

* FPDF - Used to generate PDFs
* PHPMailer - Used to send emails from the server configured with a gmail account for testing

Other technologies used within the server:

* Python3 - Used to formulate the constraint problem for Google-OR Tools
* Google-OR Tools - API for solving constraint scheduling problems

### 

### File Structure

#### Public\_html - MVC

##### View

The views contains the static HTML markup of the pages making up the web application. These files are .php files as they include the header files within PHP. The content of these pages should be static and contain almost no inline JS or PHP code, some inline PHP is acceptable if it is using stored account information to populate some fields. Every page has its own view.

##### Controller

These files are written in JavaScript + JQuery to listen to any events that take place on the views, any AJAX calls are made to the PHP Models here if needed. Client side validation is also carried out through the controllers. Each view has its own controller.

##### Model

The Models are written in PHP using OOP design. The models themselves contain different functions that the corresponding view will carry out, these are specified by a functionCall parameter set in the AJAX call of the controller. Objects representing different entities (Students, TAs, Applications, Sections, Courses,…) are used within the models to carry out much of the business logic.

#### LIB

Within the project folder is the lib folder containing a number of files used within the web application on a more global scale.

##### Examples

Examples contains reference material for creating additional views/controllers/models as well as an example for using Database.class.php for opening DB connections.

##### fpdf

This library is used to generate PDFs.

##### PHPMailer

This library is used to send emails from the server.

##### assignmentResults.csv

This file contains the assignment results from running the automatic scheduler, the sectionID, and taID are stored indicating which ta are **teaching** which sections.

##### Database.class.php

This class stores the DB login information and is used to create a DB conn object that can be used within the web application for doing SQL.

##### IP.py

This is the Python script launched from running the automatic scheduler within the web application. It is responsible for taking in the constraint model data for scheduling, formulating it into a constraint problem and using Google-OR Tools to solve it. Once solved the script writes the data back to another file for PHP to read in.

##### markingResults.csv

This file contains the assignment results from running the automatic scheduler, the sectionID, and taID are stored indicating which ta are **marking** which sections.

##### statusCode.ini

This file is used to communicate the status of the automatic scheduler within the web application. Status codes are used to indicate the current running status and are documented within the file itself. The status codes cover: idle, running, done, and errors.

#### Required Permissions

The file *statusCode.ini* must be read/writable by PHP. In addition, */public\_html/views/uploads* must be writable by PHP.

### Database Configuration

The database used is a MySQL DB setup in a relational schema, the DB UML and DDL can be found within the Design Document. PhpMyAdmin was used to configure the database, the login information was configured as:

**REDACTED**

To restore the DB use ‘DB\_Schema\_Empty\_Default\_Admin.sql’ which provides the empty table schema. These SQL DDL backups are found within the backups folder in the project directory. The default Admin account credentials for the web application are:

**REDACTED**

#### ta\_portal Tables

Below is a brief overview of each table found in the schema.

##### Administrator

This table stores which accounts are marked as Administrators.

##### Application

This table stores the Students Applications for TA positions. The database and software are configured to allow 1 Application per Semester/year, overwriting the previous entry if resubmitted. This results in 2 application per year (W/S Sessions). The Application table will store the JSON data of the transcript, schedule, and application form data.

##### Course

This table stores the course information and restriction information used for scheduling.

##### CourseInstructor

This table serves a many to many relationship if we have more than one professor assigned to a course.

##### Professor

Professors are Instructors for the courses.

##### Schedule

This table is meant to store the automatic schedule data for reuse if desired. Currently unused and unimplemented.

##### Section

Sections are the Lectures/Labs/Seminars for a given Course, these are what are used for scheduling.

##### SectionTA

This table stores the assignment information of which TAs are assigned to what Sections and the hours they’ve been assigned to that particular section.

##### TA

Once a Student’s Application has been accepted they will be marked as an inactive TA, once marked as a TA they can then be scheduled and have an offer emailed, once they accept the offer they will be marked as active.

##### UserAccount

This stores all user account information Student/TA/Administrator.

# Deployment

## Continuous Integration

Travis CI was used for the continuous integration solution, the test suite found within the Tests folder was ran on every development build verifying the tests were passing before pushing the changes up.

## Continuous Deployment

Travis CI is used to deploy our code that passes the test suite to the live server, on a successful build Travis CI will launch a script to pull the development branch of our repository to our live server. This is authenticated via ssh keys found within the server and the project folder. These ssh keys are encrypted with Travis CI, within the project it is named ‘deploy\_rsa.enc’. To configure the Travis CI build options the ‘.travis.yml’ file is used which also contains the script for using the ssh key to deploy the code.

# Known Issues

* File permissions get reset on statusFile.ini when Travis CI pushes to server
* LocalHost development on admin-course-list.php loads very slow due to Matrix code
* HTML output not parsed from Server when displaying to views
* Component to process automated scheduler results assigns preparation hours per section rather than per course (not a problem with automated scheduler but with logic for saving assignments)
* No component to manage sessions: the years are currently hardcoded and as such there is no means to automatically close applications for a session after a certain date