COMP 307
Course Project
Multiple Due Dates at The End of Document

This is your COMP 307 end of term team project.

You can optionally submit your team project for the **SOCS Website Competition**. The competition will be described in detail at the end of this document. If you choose to use your project in the competition, please tell the professor.

Your team must consist of 2 or 3 students. A group of 3 is preferable.

Your website must have as a **minimum**: a front-end, a backend, and a database (SQL or Mongo). Your website must use either a XAMPP-like stack that will run on-line using the SOCS mimi server, or a MERN-like stack that will run one-line using specially provided SOCS MERN servers setup by IT for our course. XAMPP and MERN starter code is available, however you are not required to use the starter code. You are free to explore additional tools, libraries and languages not covered in the course. You will need to get my okay for your technology stack before you start the project (more on that later).

Your project is to build a new School of Computer Science **TA Management website**.

You must code over 70% of your website by hand. This is important. We are computer science people; we need to know how to program. The remaining 30% can come from templates, frameworks, and libraries.

The best way to organize the work for the website is to divide the project into distinct areas and assign an area to one of your teammates. Make sure that each team member has been assigned an equal amount of web development work (creating PowerPoint slides and writing documentation does not count). **This is important so that the working relationships and grading can go smoothly**. If one team member fails to do their work, then **it will only affect their grade** since the work was divided into distinct equal sized units (more on this below).

Your goal is to create a functional and useful TA Management website. You are designing a professional looking website: **responsive**, **dynamic**, **interactive**, **functional**, and **pretty**. It must be easy to use by students, professors, and administrators. Be creative and design not only a good look but a simple and easy-to-use website. Simplicity of design includes **simple code**, **fewer libraries**, and **optimized functions**.

This document will describe the basic layout and features of the website. Your job will be to **design the look**, select the **technology stack**, and **build** a functional application.

If you choose to submit your project to the competition, then your website needs to run well with very few bugs.

The next pages describe the project.

Team Formation

To have a great team experience it is important to do the following:

1. Select a group of people you can work well with, and always be polite.
2. Divide the work from the start into non-overlapping areas. That way if someone fails to do their job, the prof can deduct those grades from only the offending student. When dividing the work, also make sure that dependencies are handled well. In other words, if a team member does not finish something it does not impact the other team members too much. Be smart in this and you will be very happy.
3. When something goes wrong in the team, do not let it go for a long time. Come talk to me. It is better to have a meeting with me than to have an end of semester "help us" meeting. There are many options when you come to me early (I am not scary).
4. Write a rough plan on paper. Doing this in writing is very important to make clear each other's jobs, **specify due dates**. Super, super important for a happy experience. Treat your deadlines like assignments. Don't be late in completing each deadline. If you need to be late, email your team mates to discuss (like you do with the prof when you are late with an assignment). Decide and redo the deadline schedule, tasks, workload, etc. One missed due date not handled well can cause many problems at the end of the course.

I suggest a team size of 2 or 3 students, with 3 being the optimal size. If there is a good reason why a student must work on their own, then they must contact me to get permission to work on their own. A student who works on their own must do the work of a **team-of-2**.

A **team-of-2** students must do the following:

- Design and build the website on SOCS server using either XAMPP-like or MERN-like stack
- All green and yellow boxes/circles from figure 1
- Minimum three orange and three blue boxes/circles from figure 1 (your choice)
- Optional: 1-4 bonus (red boxes/circles) elements from figure 1 (or other colours instead of red)

A **team-of-3** students must do the following:

- Design and build the website on SOCS server using either CAMPP-like or MERN-like stack
- All green, yellow, and orange boxes/circles from figure 1
- Minimum of four blue boxes/circles from figure 1 (your choice)
- Optional: 1-7 bonus (red boxes/circles) elements from figure 1 (or other colours instead of red)

Website Description

Figure 1 is a storyboard for the website. It describes the complete website, both front and back end. For this project you can build the site using the techniques covered in class or you can use other techniques that are common in website development that were not covered in class. The goal of this project is to: (1) give the student the opportunity to fully explore the skills they learned in this course, (2) work on a real-life application, and (3) explore advanced web development techniques not covered in this class, if they so choose. The project must run under **mimi** or the special MERN servers setup by IT for us.

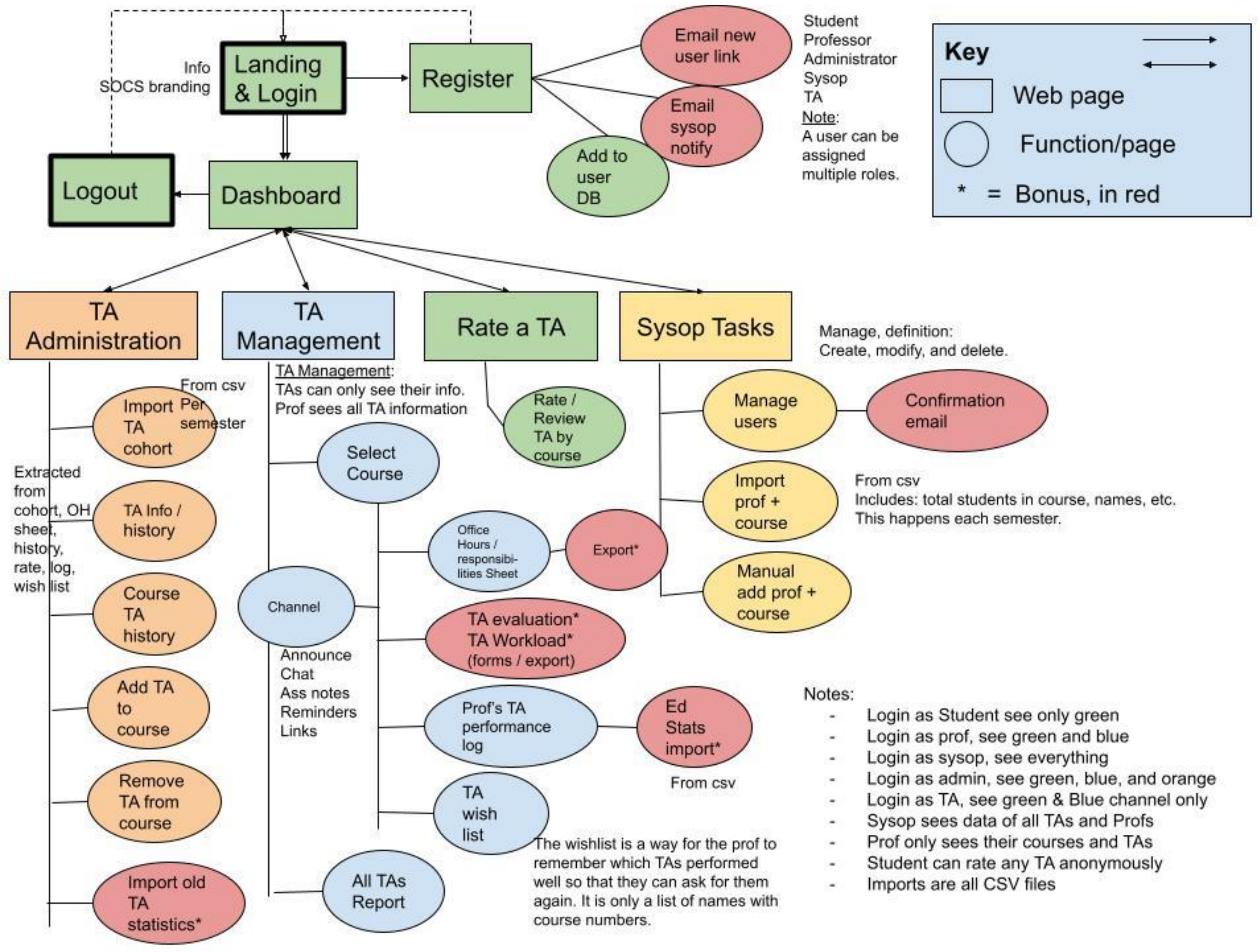
Key
Web page
Function/page
* = Bonus, in red
Info
SOCS branding
Landing & Login
Register
Email new user link
Email sysop notify
Add to user DB
Student
Professor
Administrator
Sysop
TA
Note:
A user can be assigned multiple roles.
Logout
Dashboard
TA Administration
TA Management
Rate a TA
Sysop Tasks
Manage, definition:
Create, modify, and delete.
Import TA cohort
From csv
Per semester
Extracted from cohort, OH sheet, history, rate, log, wish list
TA Info / history
Course TA history
Add TA to course
Remove TA from course
Import old TA statistics*
TA Management:
TAs can only see their info.
Prof sees all TA information
Select Course
Channel
Announce
Chat
Ass notes
Reminders
Links
Office Hours / responsibilities Sheet
Export*
TA evaluation*
TA Workload*
(forms / export)
Prof's TA performance log
Ed Stats import*
From csv
TA wish list
The wishlist is a way for the prof to remember which TAs performed well so that they can ask for them again. It is only a list of names with course numbers.
All TAs Report
Rate / Review TA by course
Manage users
Confirmation email
Import prof + course
From csv
Includes: total students in course, names, etc.
This happens each semester.
Manual add prof + course
Notes:
- Login as Student see only green
- Login as prof, see green and blue
- Login as sysop, see everything
- Login as admin, see green, blue, and orange
- Login as TA, see green & Blue channel only
- Sysop sees data of all TAs and Profs
- Prof only sees their courses and TAs
- Student can rate any TA anonymously
- Imports are all CSV files

Figure 1 – Website Storyboard

Figure 1 is divided into 5 colours. The storyboard format follows the convention described in class (database pictures are not shown to save space). Please make sure to adhere to the meaning of the arrows. Green represents those areas of the website that all users can interact with. Orange represents those areas that the TA Administrator can access. Blue represents those areas the Professor and TA can accesses. Yellow represents those areas the System Operator can accesses. Red represents bonus material. The bonus material is optional.

This website has the following user types: student, professor, TA administrator, teaching assistant, and system operator. Please note that a user's account may belong to more than one user type. For example, a professor could also be a TA administrator and/or the system operator. A teaching assistant could also be an instructor (professor). A student can also be a teaching assistant.

All users can belong to more than one course (professors and students belong to more than one course). The website must ask the user to select the course they want to interact with for a given feature (eg rate a TA, or chat on a channel, etc.) or automatically determine the course (if they are already within a course). How you want to handle this is up to you.

Users can only see those dashboard options that correspond to their user account type. For example, a student who is also an instructor can only see the green and blue areas in their menu.

GREEN AREA

This consists of the portion of the website that all users can interact with. It contains the landing page, registration and login, the dashboard, logout, and rate a TA. Notice that the **dashboard** is only reached after successfully logging in. To **login**, a database of usernames and passwords must be consulted. You must use an SQL or No-SQL database. As a minimum the database must contain these fields: first and last name, email, student ID number, username, password, and courses_registered_in (plural) and semester. The user will need to specify which courses they belong to. **Registration** adds the new user to the database. The user self-reports the courses they have for the semester when registering. The System Operator can edit this list later. For bonus, confirmation emails to the new user, and a notification email is sent to the system operator. The **landing page** must be professional looking, responsive, and informative to new users. The login is incorporated into the landing page (there is no login page). When **rating at TA**, the user must specify which course and term. Rating a TA gives a score from 0 to 5 (where 5 means the best). This option also provides a space to leave a short twitter-like comment.

YELLOW AREA

The yellow area is accessed by the system operator and consists of: the management of user accounts (edit, delete, add), quick import of profs and courses from a CSV file (normally provided by the SOCS office), and a manual way to input professors and courses (instead of using the CSV file). As a bonus feature confirmation emails are sent to the user when their account was modified, deleted, or added. The CSV file has the following format: `term_year, course_num, course_name, instructor_assigned_name`. For example, `“Winter 2022”,”COMP307”,”Intro to Website Design”,”Joseph Vybihal”`. A user account contains all the information provided during registration/login. Bonus: on following years students will need to re-register on this website to specify the new courses they belong in for rating a TA. How to do this without adding a new record?

BLUE AREA

The TA Management area is visited by both the professor and teaching assistant. To enter this area the user must specify which course they want to interact with. After selecting the course, all the options interact only with the selected course. To switch to another course, the user must exit to the dashboard and then return to this area (or some other method can be used). The **office hours and responsibilities feature** allows the professor and teaching assistant to define the office hours, office locations, and duties of each TA and Prof. Look at the information contained in our course TA coordinates spreadsheet for an example of the information that must exist in this feature. The data must be stored in the database. It has a bonus feature to export (or print) something that looks similar to the TA Coordinates spreadsheet but as a report using a table format, which would be shared with students by the professor on myCourses. The **TA performance log feature** can only be accessed by the professor. It stores the professor’s notes about each TA in the course. The professor selects the TA from a dropdown and then is given a twitter sized space to write a note about the TA. This is appended to the database and time stamped (date & time). The professor can append multiple performance notes per TA. This DB record must have as minimum the following fields: `term _year, course_num, TA_name, comment, time_stamp`. The **TA wish-list feature** can only be accessed by the instructor. The instructor can identify to the TA administrator which TAs they would like to have next semester. This information is appended to a DB record that has as minimum the following fields: `term_year_this_is_for, course_num, prof_name, TA_name`. The **all TAs report**

feature can only be viewed by the professor. Selecting this feature displays an on-screen report that collects and displays all the stored information about each TA in the course. The following information is displayed: `TA_name, assigned_responsibility, student rating average, performance log comments, and student rating comments`. The **channel feature** is used by both the professor and TA and functions like a Slack channel where the teaching team can speak together. It has only one channel per course. The **bonus feature TA Evaluations/Workload** is an important element of this website. If you are interested in creating this feature, please email the professor.

ORANGE AREA

This is the TA administrator area. The administrator decides which TA should be assigned to a course. To do this well, the administrator needs to know the TA's performance history, which courses they have been assigned to in the past, and in which responsibilities they excelled in. The administrator also needs to be aware of the professor's TA wish list and which courses the TA wants. The **import TA cohort feature** is the way the administrator receives from McGill the list of teaching assistants hired by the university for that semester. It contains the courses the TA said they are interested to TA. The import is from two CSV files called `CourseQuota.csv` and `TACohort.csv`. Course Quota has the following fields: `term_year, course_num, course_type, course_name, instructor_name, course_enrollment_num, TA_quota`. TA_Quota is the maximum number of TAs that can be selected for the course. Course_Type refers to Regular, Seminar, Lab, Other. The website then automatically compute enrollment_num divided by TA_Quote and flags anything where the number of students per TA is lower than 30 or higher then 45. These are problematic numbers and the Administrator will need to fix that. TA Cohort has the following fields: `term_year, TA_name, student_ID, legal_name, email, grad_ugrad, supervisor_name, priority(yes/no), hours(90/180), date_applied, location, phone, degree, courses_applied_for_list, open_to_other_courses(yes/no), notes`. The administrator will use this information to manually match the TA to a course. The **TA info/history feature** gathers all the information about the selected TA from all sources: TA Cohort, student rating average, professor performance log, student rating comments, prof wish list membership, the courses they are currently assigned to TA this term. This functions as a report. The administrator will use this feature to help determine the qualification of the TA and assign/remove them for courses. The **course TA history feature** displays a table of each TA with the courses they have been assigned to this term and the courses they have been assigned to in the past. Note: Course History and Info History could be combined into a single report/interactable table (a single feature). Also note: the administrator wants to be able to see: (a) select a TA to see all the courses they have been assigned to, and (b) select a course number to see all the assigned TAs. The **add TA to a course feature** and the **remove TA from a course feature** both modify the same database having the following information as minimum: `term_year, course_num, TA_name, student_ID, assigned_hours`. The **bonus feature import old TA statistics** provides additional historical information from external sources. If you are interested in implementing this bonus feature, please contact the professor for more information. **IMPORTANT**: The Orange Area needs a lot of creativity on making it a useful place for an administrator to inspect prospective TAs, assign/reassign them to courses, and track statistics. This Orange area is the one place that you can change the organization and look from what I have described to make it more useful. For example, the orange area could be transformed into a type of dashboard/spreadsheet that displays information with buttons to select what the administrator may want to do.

You are given a lot of liberty as to how you can implement the website. For example, I have not specified database names only fields. You may even combine different records, except do not do that for importing (these come from the university as separate files).

I suggest that you only do bonus features after you complete the minimum regular features you decided upon.

Project Instructions

Please follow these steps:

Step 1: Team and project decision (No later than November 18, 2022)

- Select a team leader who will be responsible to communicate with the professor, schedule events, and fill out forms.
- Using the following link (https://docs.google.com/spreadsheets/d/1m9MfZBNw8LCGlIK2eEamrRCu6ZRPq3bffeixgfOMRpY/edit?usp=sharing) tell me who your team members are, who the team leader is, your technology stack and whether you will be submitting your project in the competition (you can say yes and later decide not to).

Step 2: Your presentation is due December 7, 2022

- Your website does not need to be complete on this date.
- Submit a video of the presentation and a Power Point of the presentation.
- Each team member must speak in the video describing the portion of the website they created. Each person speaks for 3 minutes (points deducted for longer than 3 minutes) about the technology stack and feature in their portion of the website using the PowerPoint. Then the leader will give a 3-minute demo of the website.
- Submit this to myCourses **Project Demo** assignment slot.

Step 3: Website Submission December 11, 2022

- Submit your project as a ZIP file to myCourses, but also have a working website in the public_html directory of the **team leader**.
- Submit a readme.txt file with:
  - your team member names,
  - URL to the public_html landing page at SOCS
- **All team members submit the entire project**. This is important because it is easier to enter a grade in myCourses when a student has submitted something than when they have not. The TA will grade the Leader's project and then use the readme.txt file to distribute the grade to the other team members. Due to team issues, the readme.txt file may not be the same as the Google doc you filled out above.

COMPETITION INSTRUCTIONS

If you registered for the competition and then decline at the end, make sure to tell the professor.

At the end of December, the professor will select 3 best projects. These 3 projects will be announced to everyone in class by email. The professor will then submit these 3 projects to the School of Computer

Science. The school reserves the right to not select any of the 3 projects. If they do select one of the projects, then an email will be sent to the class announcing the winner. This will probably happen by the end of January. At the same time, the winning project will be placed on Professor Vybihal's Hall of Fame webpage. The winning project has two levels of win: (1) SOCS will adopt the students project as-is!!!, (2) the winning project will go through additional development (through a COMP 400 project or as a paid position). The winning team might be invited to help deploy the website.

REPO STARTER CODE

- Your team captain will need to email their mimi username and email address to me. I will add the captain only to the repo. The captain will then receive an invite email. The email should take then to the login URL (see below) to access the repo. If not, follow the instruction below:
- Go to URL https://gitlab.cs.mcgill.ca/users/sign_in
- Select LDAP tab and use your mimi login credentials.
- After logging in you will see the repo named: TA Management. Select that repo to enter.
- You must **git clone** the above repo (**https://gitlab.cs.mcgill.ca/jvybihal/ta-management.git**), then **switch branch to MERN** (or XAMPP), and follow the rest of the installation instructions (it is in the **installation-instructions.md** file in the repo itself).
- The captain will then share the code with the rest of the team. Do not use the repo I shared with you to develop on. Create your own repo. You will only have access to the repo until December.

HOW IT WILL BE GRADED

- TOTAL: 100 points

- POINTS:
  - (Pass/Fail) Professor approves team's tech stack
  - (Pass/Fail) Team of 2 or 3 students (special permission & conditions for other sizes)
  - (Pass/Fail) 70% of project was coded from scratch
  - (Pass/Fail) Front-end (existence of)
  - (Pass/Fail) Back-end (existence of)
  - (Pass/Fail) Database (existence of)
  - +10 points for a fully running website (partial running loses all 10 points)
  - +10 points for a professional looking responsive website (graded proportionally)
  - +10 points for good coding style (comments, indentation, modularity, reuse)
  - +70 points for meeting the 2- or 3-person team requirements (graded proportionally)
    - Front-end code quality (HTML, CSS, JS and related languages)
    - Front-end layout quality (responsive, interactive, page design, easy to use)
    - Front-end color and theme (pretty?)
    - Backend code quality (good COMP 303 techniques, good SE techniques)
    - Backend usefulness quality (does it work, is it working well, is it easy to use)
    - Good database usage (meet project requirements)

- BONUS:
  - 7 bonus points. One point per red circle.
  - You cannot receive more than 100% on this project.
- REDUCTIONS:
  - -10 points for not following instructions (proportional)
  - Standard late penalty (note there are multiple due dates)
- COURSE OUTLINE:
  - Project demo is 25% of the course grade and is divided into two parts:
    - Video presentation 15 points.
    - Code running on SOCS 10 points.
  - Project code is 25% of course grade, described above marked out of 100 points.

- Points are awarded proportionally unless otherwise stated.