SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

TA Emergency Coverage System

by

Ken Wakaba
Joseph Thèberge

Santa Clara, California
November 30, 2017

Table of Contents

List of Figures

1 Abstract

When a TA is sick or cannot make lab, they system will notify the professor of the absence and the professor will then select a TA that is available to cover. At the current moment, if a TA is unable to attend the lab for any reason, there are a string of steps in which the TA can notify the professor and the professor to find a replacement. Additionally, all the information the professor has (in terms of schedule and availability) is all kept locally and is done physically. For an update to be done, the professor has to contact the TAs via email or talk in person to get more information, which takes time. The solution we are looking for is a centralized location in which all the TAs contact information and availability is located, and a way for the professor to be notified if a TA is unable to come to lab so the professor may be able to find the next available person and contact them to cover for the lab. Professors and TAs need a usable system, preferably intuitive, that can be updated on changing schedules of TAs.

2 Introduction

When a student attends a lab, they expect a TA to be present. The TAs role is to assist and support students on their lab in any way they can. However, a problem presents itself when a TA is unable to attend their designated lab section. Currently, there is no efficient way for a TA to alert the students or the professor in charge of the class of their tardiness or absence from a lab. If a TA is sick or simply cannot make the lab due to an emergency, the only way they can relay that information to the proper parties is either through email or even text messages and calls. This could be potentially problematic if the professor does not receive the information in a promptly manner due to the fact that they will have minimal time to find coverage for the lab section. As a result, they run the risk of students missing information about the lab or even canceling the lab altogether, delaying lesson plans.

It is true that there are tools available, such as Google Calendar, that allow TAs to share their schedule with the professor so that in the event of an absence, the professor could notify the next available person for coverage. The issue with these solutions is that they still present the problem of notifying the professor in an abrupt manner so they can immediately find a replacement. As a result, the faculty are in need of an efficient and prompt way to alert both other TAs and professors in the event of an absence.

Our web based application will allow TAs to alert their professors of their potential absences so the professor may be able to swiftly find coverage. TAs will have the ability to input their schedule of availability for other TAs and the leading professor to view. If anything changes with their schedule, they can update it and the changes will apply automatically, keeping the master schedule live; any change to the schedule will also notify the professor so they too are kept up-to-date with their students availability. When a TA is unable to attend a lab, they can log into the system and note their absence for the lab. With this accessible knowledge, the professor can then easily pull up the master schedule to view who is available at the proper time, select all the TAs who are free, and are able to contact others to see who is able to come in on short notice. This greatly improves the efficiency in which the TAs and leading professor can communicate with each other because all of the information will be in one central location where all the availability and contact information will be located, as opposed to searching for contact information to email or text. Our solution will allow for smooth communication between professors and TAs by keeping a central online location for collaboration.

3 Requirements

We have established three types of requirements to make sure that our system works and meets our criteria: functional requirements, which describe how our system will work; non-functional requirements, which describe how a system should behave; and design constraints, which provides limits on how our system operates.

3.1 Functional

The system will allow:

- TAs to input/edit their schedule (Critical)
- TAs to notify the professor of their absence (Critical)
- professors to view TA availability and find replacements (Critical)
- professors to give TA permissions for labs (Critical)
- users to send email notifications (Recommended)

3.2 Non-Functional

The system will:

- show clear content and will be easy to use (critical)
- adaptable (critical)
- be low cost (critical)
- have an non-distracting intuitive user interface (recommended)

3.3 Design Constraints

- Run in the ECC servers at Santa Clara University
- Viewable on Chrome version 61.0.3163.100 (Official Build) (64-bit) and Firefox version 52.4.0 (64-bit), the browsers included in the ECC.

4 Use Cases

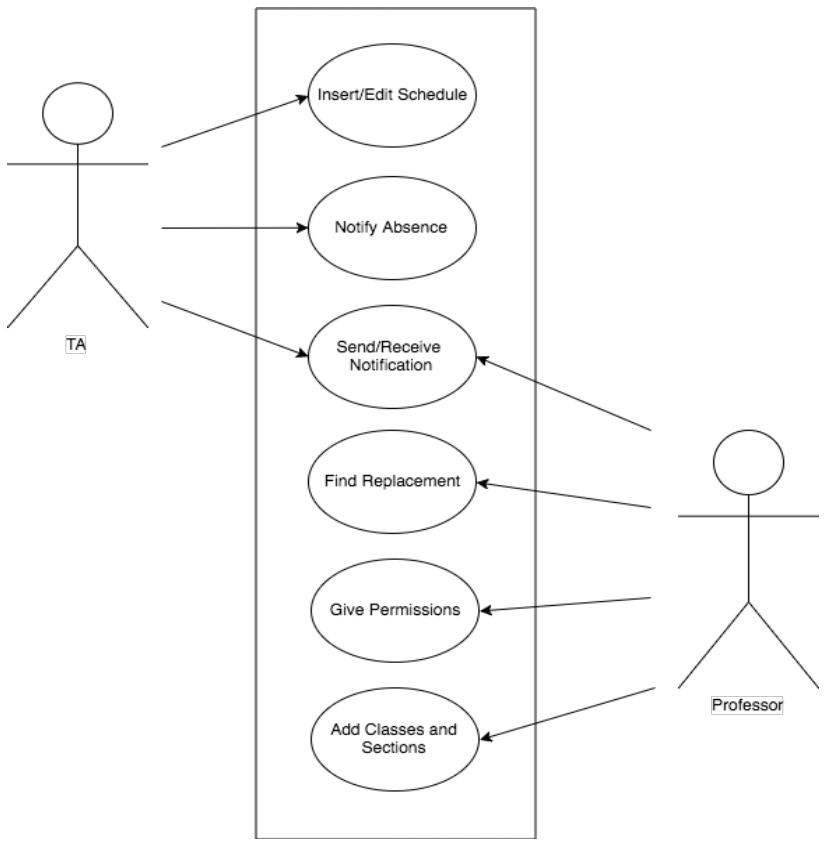


Figure 1: Use case diagram

4.1 Insert/Edit schedule

Goal: User will be able to input/edit their schedule

Actor: TA

Pre-Conditions: User must be logged into website

Post-Conditions: Users schedule is updated and uploaded into system

Steps:

1. User logs into the system
2. User selects option to input/edit schedule
3. User selects the classes from the listed options
4. User clicks Save button

Exceptions: TA has no permissions given to edit schedule

4.2 Notify Absence

Goal: The user will notify the professor of their absence from certain lab sections

Actor: TA

Pre-Conditions: TA must be logged into website and must be enrolled in certain sections

Post-Conditions: Professor has received notification of TAs planned absence

Steps:

1. User logs into system
2. User selects option to notify absence
3. User selects lab sections they will miss, providing a quick explanation
4. User submits form

Exceptions: User decides to cancel the form or is later able to attend lab after submitting absence notification

4.3 Send/Receive Notification

Goal: Professor/TA receives a notification

Actor: Professor/TA

Pre-Conditions:

- TA must have sent alert to professor
- TA must have changed their schedule
- Professor must have found replacement

Post-Conditions:

- TA schedule updated and notified professor of absence
- Professor has found replacement and notified new TA

Steps:

1. TA fills out form to notify professor of absence
2. Professor notifies the TA that is available to cover the lab section

Exceptions: There is no available TA to cover

4.4 Find replacement

Goal: Professor finds a replacement for the absent TA

Actor: Professor

Pre-Conditions:

- TA must have notified and posted of his/her absence
- All TAs schedule must be up to date

Post-Conditions: Professor will have found and notified the replacement TA

Steps:

1. Professor receives notification of TA absence
2. Professor goes into system and check schedule of when TA is missing
3. Professor searches and finds replacement
4. Professor sends notification to replacement

Exceptions: There is no available TA to cover

4.5 Give TA Permissions

Goal: Professor will give a TA the permissions to edit their schedule

Actor: Professor

Pre-Conditions:

- TA must have created an account

Post-Conditions: TA will now be able to input and edit a schedule on the application

Steps:

1. Professor logs into application and goes to Permissions section
2. Professor will select the TA and the classes they are allowed to TA for so they are able to select their availability

Exceptions: TA has not created their account

4.6 Add Classes

Goal: Professor will be able to add new courses and labs

Actor: Professor

Pre-Conditions:

- Professor must be successfully logged into system

Post-Conditions: New course and section times will be uploaded onto system for both the professor and TA to access.

Steps:

1. Professor logs into application and goes to Add Classes section
2. Professor will input the class name and select the days and times the lab sections will be held.
3. Professor will press save to store values in database

Exceptions: Class/Section is already inputted into system

5 Activity Diagram

Figures 2 and 3 show the flow of actions for our different users in the system. The flow of both the TA and User are highlighted below.

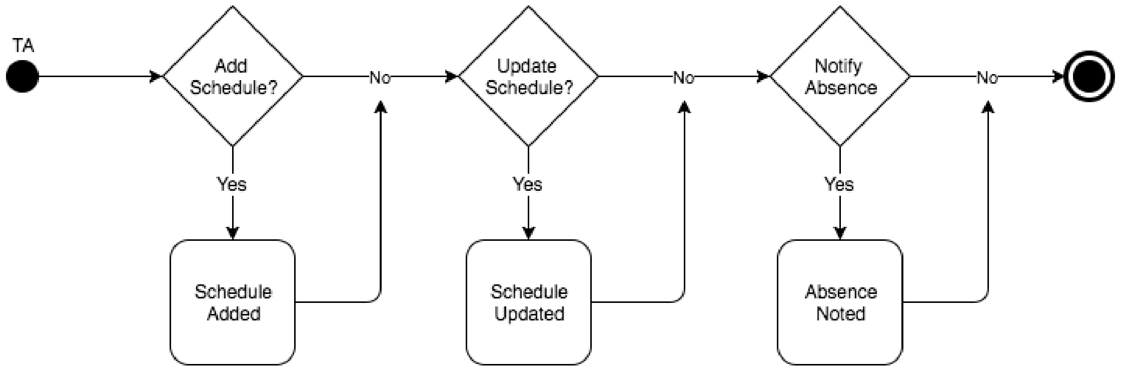


Figure 2: TA Activity Diagram.

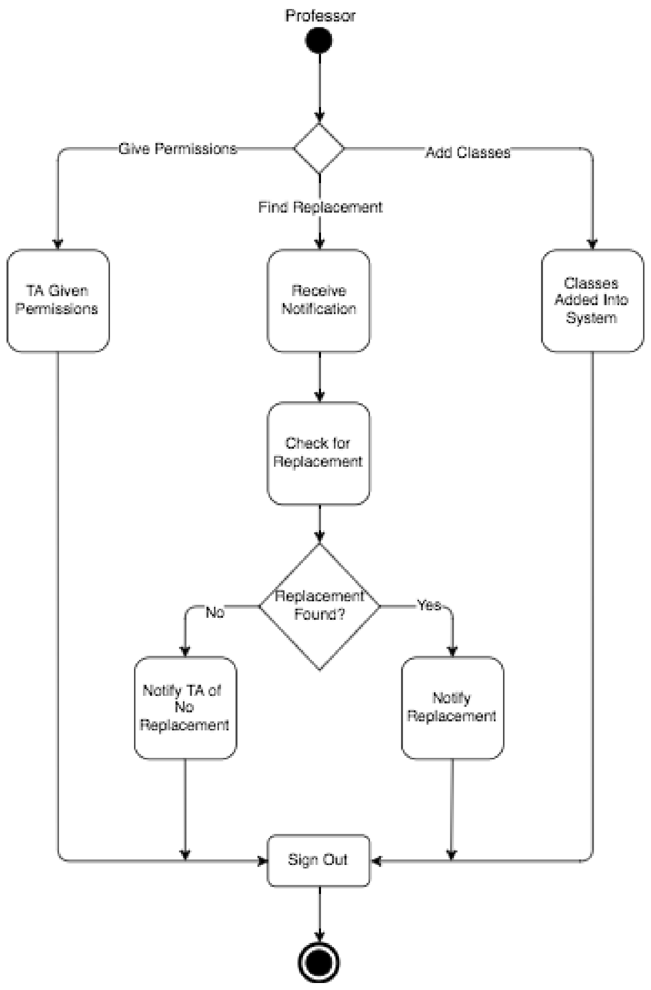


Figure 3: Professor Activity Diagram.

6 Concrete Models

Both the TA and the admin will be presented with a login page upon starting the application as seen in Figure 4. In the case of the TA, if a TA has yet to have an account, they may select to create a new account and enter the respective information as shown in Figure 5.

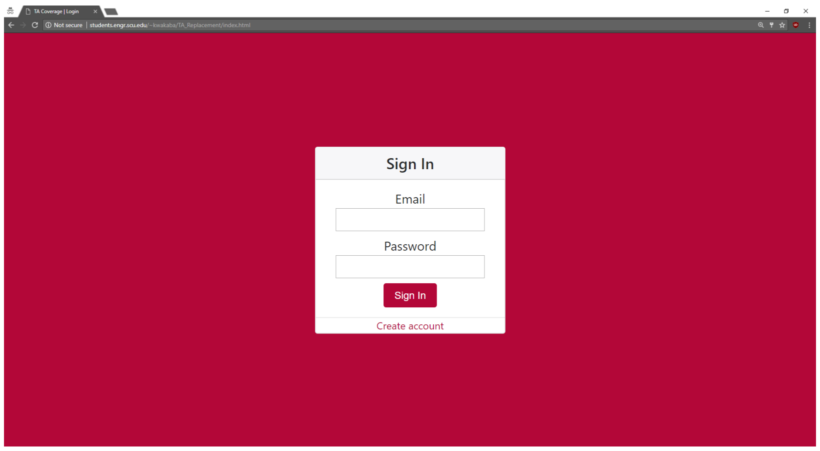

TA Coverage | Login
Not secure | students.engr.scu.edu/~kwakaba/TA_Replacement/index.html
Sign In
Email
Password
Sign In
Create account

Figure 4: Login Page

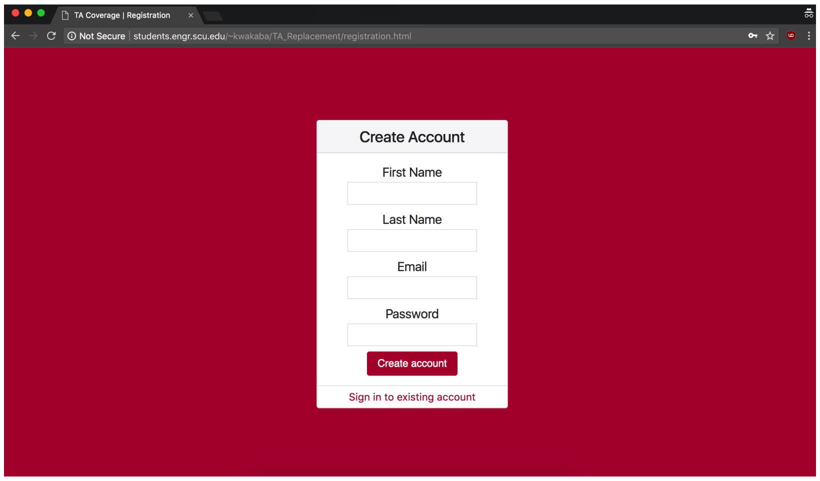
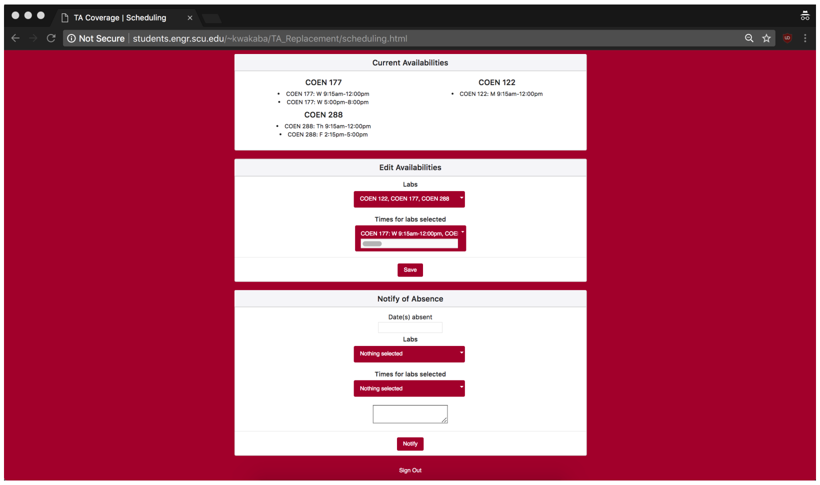

TA Coverage | Registration
Not Secure | students.engr.scu.edu/~kwakaba/TA_Replacement/registration.html
Create Account
First Name
Last Name
Email
Password
Create account
Sign in to existing account

Figure 5: User Account Registration


TA Coverage | Scheduling

Not Secure | students.engr.scu.edu/~kwakaba/TA_Replacement/scheduling.html

Current Availabilities

COEN 177
- COEN 177: W 9:15am-12:00pm
- COEN 177: W 5:00pm-8:00pm

COEN 288
- COEN 288: Th 9:15am-12:00pm
- COEN 288: F 2:15pm-5:00pm

COEN 122
- COEN 122: M 9:15am-12:00pm

Edit Availabilities

Labs: COEN 122, COEN 177, COEN 288

Times for labs selected: COEN 177: W 9:15am-12:00pm, COE

Notify of Absence

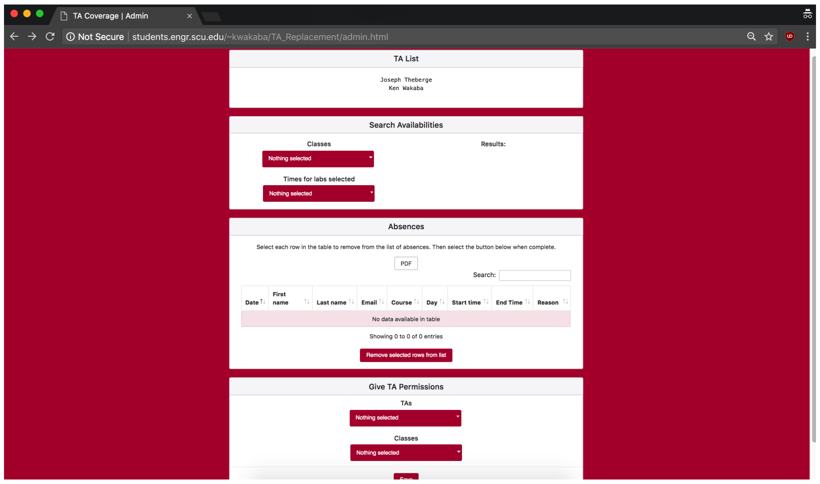
Date(s) absent

Labs: Nothing selected

Times for labs selected: Nothing selected

Figure 6: Main TA Page

Upon the TA login, they will be presented with a page similar to that of Figure 6. Within the TA page, they will be able to view and edit their availability, as well as communicate to the professor if they will be absent in any lab sections.


TA Coverage | Admin

Not Secure | students.engr.scu.edu/~kwakaba/TA_Replacement/admin.html

TA List

```
Joseph Theberge
Ken Wakaba
```

Search Availabilities

Classes: Nothing selected

Times for labs selected: Nothing selected

Results:

Absences

Select each row in the table to remove from the list of absences. Then select the button below when complete.

Search:

Date	First name	Last name	Email	Course	Day	Start time	End Time	Reason
No data available in table								

Showing 0 to 0 of 0 entries

Give TA Permissions

TAs: Nothing selected

Classes: Nothing selected

Figure 7: Main Admin Page

Once the admin has successfully logged in, they will be presented with a page similar to that of Figure 7. Within the admin page, they will be able to view a full list of TAs in the system, as well as the ability to search for a TA who is available at a given time. They will also be able to give TAs permission to select courses on their own page. The admin will also be able to view a table of all reported absences sent in by TAs so that they may be able to find a replacement. In addition, the admin will be able to add new classes and lab sections to be used within the system.

7 Technologies Used

HTML 5

- This is a requirement of the project, allowing modern browser support.

PHP

- This will allow us to communicate with our database, sending information over in order to run queries and receive results

SQL

- The data will be stored with SQL. SQL was chosen for its simplicity to manage with PHP and wide documentation and support.

Bootstrap 4

- This allows flexible, responsive design while making development easy and fast.

Popper.js

- This is another dependency for Bootstrap. This will be necessary for easily creating flexible drop-down menus.

Jquery

- This is a dependency for Bootstrap. We will mostly not need the library for our needs, but it must be included regardless.

SASS

- SASS compiles .sass to .css, and allows programmatic creation of stylesheets that allow easy cross-browser compatibility, variables, mixins, and more. It also allows compressed creation of stylesheets.

Git

- Version control will be managed with this during development and allow for easier cooperation.

8 Architectural Diagram

Figure 8 shows the architectural design we chose. Our system will utilize a client-server model that connects to the design center computer. This will handle the data processing using a data-centric architecture to store all the information passed in by the users. Once presented with webpage that will utilize HTML, CSS, and Javascript, the user will be able to interact with the server. The user can both submit data into the server, which will update dynamically, as well as request data, which will be passed back.

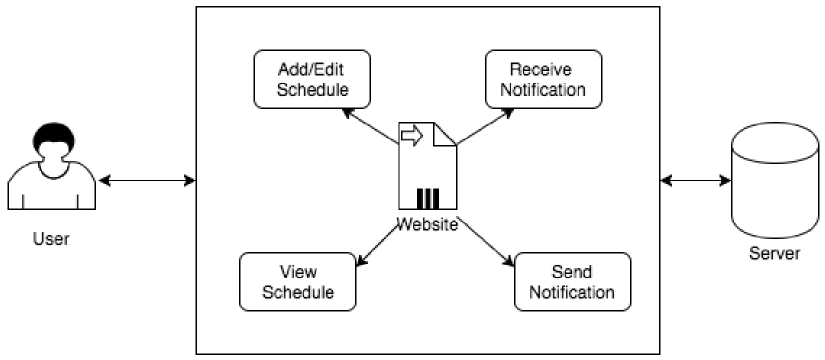


Figure 8: Hybrid Architectural Diagram

9 Design Rationale

The design decisions we made are most optimal for users because it offers a simple solution that is efficient, fast, and cross-browser compatible. We use the technologies listed in Section 7 for rapid development and easy-to-maintain code. Because of Bootstrap, mobile and desktop design becomes effortless. Having a client-server model hosted in the design center restricts us to some technologies, such as the latest browser versions, which is why CSS will be compiled to be cross-compatible between browsers to account for older obsolete browser versions located in the designer center. PHP is used as method of communicating with our SQL database. Through multiple queries, we are able to send information to the database from the website and vice versa in a fast and efficient manner.

The TAs will be able to update and save their schedules at any point, which will then be copied over to the server we are using. This is beneficial because the information is never lost, meaning both the TA and Professor can view the schedule in real time.

When a TA wishes to send a notification, a form will be presented to them to select the labs they will be missing, as well as a section to provide reasoning as to their absence. For this to be possible, our server will contain all the Professors information such as Name and Email, as well as the courses they are teaching for that quarter. Once the notification is sent, the professor will receive an email indicating that their has been a reported absence from one of the TAs, which will then cause the professor to go into the system to find replacement.

10 Test Procedure

With every new implementation we included into the system, we ensured to test it rigorously.

With the initial login page, we ensured that the user was sent to the respective page, depending on if they were a TA or admin. In order to do so, once the user would enter their login credentials, we would then send the information to the data base to cross-reference with what we have stored. If the user is within the system, the proper response is sent back to indicate if they are a TA or admin. If there information is not in the database, an error is sent back to the front end indicating that the wrong credentials were inputted of the user is not within the system. With a successful login we also had to make sure that a session was established for both the TA and the admin. This allows extra security in the case of someone attempting to access either page directly through the URL without logging in first.

For the TA page, we ensure to test multiple cases. First, we ensured that upon successful login, the TAs schedule would automatically populate a section of the page, showing what they already inputted previously into the system. To do so, we had to tested to make sure that upon successful login, the database would send back the users schedule from a different table along with a success message. We also needed to make sure that we only sent back the specific TA schedule and not all the schedule inputs within the database, meaning we had to assign a key to each schedule value. When it came to the TA inputting a new schedule or editing a current schedule, we would test that each time the TA would press save that the data would be sent over to the back-end to update the database as well as updating the front-end visual, allowing the TA to see that the changes have been made successfully. This testing was done by inputting a few lines of code within the database entry to return a success or error with each new schedule update. In terms of the TA notification, we tested that the proper sections the TA selected as well as the message they send along with it was successfully passed to the professor page in order for them to view. This was simple to test because if the notification was sent successfully, the professor will be able to see new notification on their page and if the notification sent unsuccessfully, nothing would appear for the professor.

The admin or professor page was simpler to test for since there were fewer use cases to worry about compared to the TA page. The largest portion of testing that was done was on the notification portion. We had to ensure that the TA absence notifications were sent successfully and were appearing in their entirety on the admin page. Along with this, we also had to ensure that the admin could also send a notification to a TA to notify them that a replacement is needed. In addition, we tested for the case of the admin looking at TAs who are available for a certain lab section. In order to do so, we had to check that for each input the admin sends, inputs being the section name and time, that the appropriate TAs would be returned from the database.

11 Obstacles Encountered

Within the process of implementing our system we encountered quite a few problems along the way. The biggest of them all that caused a chain reaction was that our initial implementation was done using the wrong technologies. Prior to the first project demo, we had implemented our system using NodeJS, NPM, and Localhost. NodeJS is an open-source cross-platform JavaScript run-time environment. It is what allowed us to launch and maintain the server as well as serve content to the user. This was done in conjuncture with Localhost, a hostname that means this computer. It is used to access the network services that are running on the host via its loopback network interface, meaning we were able to access our server from NodeJs using Localhost. NPM, or node package manager, allowed us easy package management for our project. It ensured that the user running the project that launches the server uses the correct versions of packages that we declare fit for our website application. The issue with all this is that we were to run our application not on Localhost but on the school server. Because the school server does not support NodeJS as well as NPM, we were forced to scrap our entire implementation and start from scratch, leading to further obstacles.

In order to connect and correspond with the school server, we needed to use SQL as well as PHP, new technologies that our team had no prior knowledge of. This meant that we had to learn new languages, along with determining how to implement them into a design we had to come up with. We had difficulties figuring out how the two communicate with other as well as how they communicate with HTML to display the information but eventually we were able to get things going again.

An underlying obstacle that we faced was that of time. We were forced to redesign our entire implementation half way through our designated time period, meaning we only had half the time to produce a fully functional application. In addition to the time constraint, our team members had constant time conflicts when it came to schedule meeting times due to classes or work, resulting in late night meeting sessions.

12 Conclusion

Although our application was a success, there are still areas in which it could improve on.

Overall this project has been a great learning experience for the team. We have learned a great deal of things throughout the process of implementing our application. To begin with, we have learned that it is important to ask the client as many clarification questions as possible in the design portion of the implementation. By doing so, you minimize the number of issues you will encounter later on in the development. Perhaps if we would have asked for some more clarification to the specification of the design, we would have run into so many of the obstacles refereed to earlier in section 11. In addition, we have also learned of the overall structure and process that goes into a Senior Design. This means not only the implementation of your project, but the design that occurs before hand, as well as the documents that must be produced as well as the presentations that must be given in order to receive feedback and clarification. Along with this was the lesson of importance of team collaboration. We saw how beneficial it is to communicate amongst one another within a team, ensuring each and every member has a fair task as well as deadline to complete the task so that the team, as well as the project, can stay on track in terms of their timeline. We also learned a few technical technologies such as NodeJS, PHP, and SQL. In conclusion, the application and the process that went into designing and implementing it showed to be truly beneficial.

Appendices

A Installation Guide

To install you own version of the TA Emergency Coverage System, follow the steps provided below. Please note that you as a user have an SCU Design Center account, as well as have access to the computers.

1. Upon successful login into a Linux machine, open up a Terminal window.
2. If you do not already have an SCU student *webpage*, run the command webpage on the Terminal and follow the onscreen instructions.
3. You will need to unzip the files. To do so, run the command **tar -xzf KenWakaba_JosephTheberg_TAReplacement.tar.gz**
4. Once the webpage has been established, place all the files located on the USB Flash drive in /webpage/ ∼*username*. You can find more information on the SCU's wiki, found at:
   `http://wiki.helpme.engr.scu.edu/index.php/webpage`

   If connecting remotely, FTP and SFTP is supported. The guide to this is in the same link. Note that the filepath for FTP is slightly different, with the path:
   /DCNFS/users/web/pages/ ∼*username*
5. Next, you will need to setup a MySQL database. To do so, follow the instructions on SCU's wiki, found at: `http://wiki.helpme.engr.scu.edu/index.php/MySQL-5` Following the instructions provided, you will be able to setup the entire database either by running the commands provided, or using phpMyAdmin to manage the database through a website GUI (The login credentials for phpMyAdmin will be provided for you in the Wiki as well).
6. Once the database has been created, you will need to access all the .php files in the directory to change the host, user, password, and database variables to match your MySQL credentials.
7. You will need to follow the next few steps to set the permissions for all the files in the directory as well as setup PHP-CGI
   (a) To begin, run the command "**chmod 755 *directoryname***" on the directory that houses all the files
   (b) Next, you will need to ensure your PHP files are secure. To do so, you will need to make sure CGI is set. Follow the instructions provided on SCU's wiki link provided in step 1. Once those steps have been followed, you will need to go to every directory that houses any PHP files and run the command "**chmod 600 *.php**". This will set permissions for all PHP files in the directories. (NOTE: the access file that is asked to be created must be in the directory where all the .php files are located)

   The TA Emergency Coverage System should be fully functional on your system upon completing the steps above. To access the application, you may visit either `http://students.engr.scu.edu/~kwakaba/TA_Replacement/index.html` to access the main login index directly or you may visit `http://students.engr.scu.edu/~kwakaba` and follow the link description that will direct you to the main login index. Please refer to the SCU Wiki's for further assistance if needed. We hope that the guide has been helpful. For instructions on how to use the application, please proceed to the next section.

B User Manual

Below are instructions on how to use the application. It has been broken up into 3 main components" Instructions on how to Login, how to navigate the TA page, and how to navigate the Admin page.

B.1 Login

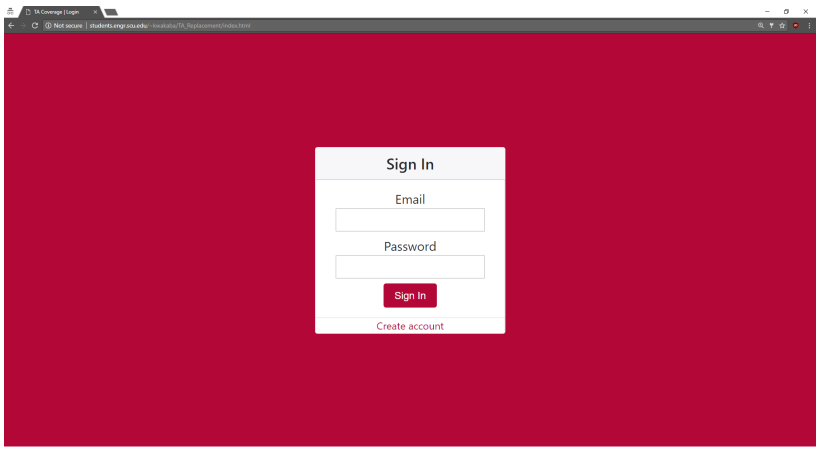


Figure 9: Login Page User Guide

In order to log in, you may visit either `http://students.engr.scu.edu/~kwakaba/TA_Replacement/index.html` to access the main login index directly or you may visit `http://students.engr.scu.edu/~kwakaba` and follow the link description that will direct you to the main login index.

Upon entering the login page you will be prompted to enter your email and password to log in. If you are a new user, you may select the **Create Account** option in which you will be directed to enter your information to create an account. Once an account is created, or if you are already within the system, simply input the **Email** and **Password** in which you registered with. If either one is incorrect, you will be prompted to enter once more. Only by entering the proper credentials will you be allowed a successful login. If logging is as a TA, you will then be directed to you respective TA Page and if you are a professor/admin, you will be directed to the admin page.

B.2 TA

Upon successful login as a TA, the user will be directed to a personalized page with their TA information presented to them. This includes showing their current schedule, as well as options to Inserting or Editing a schedule, as well as sending a notification to the professor reporting their absence.

B.2.1 Insert Schedule

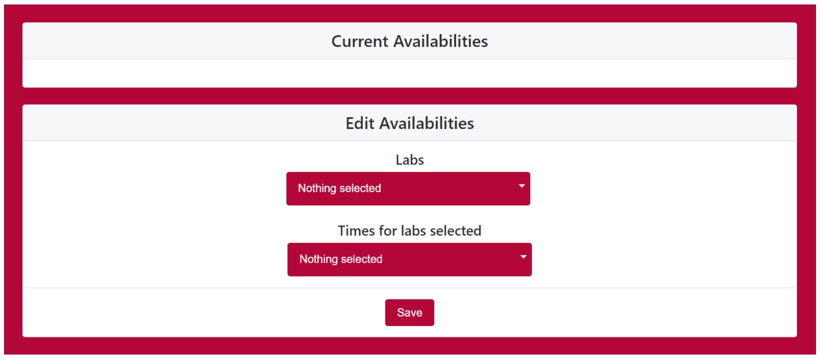


Figure 10: Insert Schedule User Guide

As a new TA in the system, you will be presented with a page very similar to the one shown in Figure 10. The section labeled **Current Availabilities** will be empty to you as a user since you have just created your account and nothing has been saved with your credentials within the system. To enter a new availability schedule, refer to the section below Current Availabilities labeled **Edit Availabilities**; this is the section you will be able to add you new availability schedule. You must first select the **Labs** option to open a drop down menu of all the labs you as a TA are allowed to teach (NOTE: This list is provided by the professor. If you see a class in which you are not registered to teach or you are unable to view a class in which you are allowed to teach, contact you professor so they may be able to give you permissions.)

Once you have selected the courses you are available for, select the sections in which you are able to teach the labs in the **Times for Labs Selected** Section (NOTE: The section times will be unable to be viewed or selected unless a Lab has been selected).

Once the desired labs and sections have been selected, simply press **Save** and the information will be sent to the database to be updated and stored, as well as the information in the **Current Availabilities** section will be updated accordingly.

B.2.2 Edit Schedule

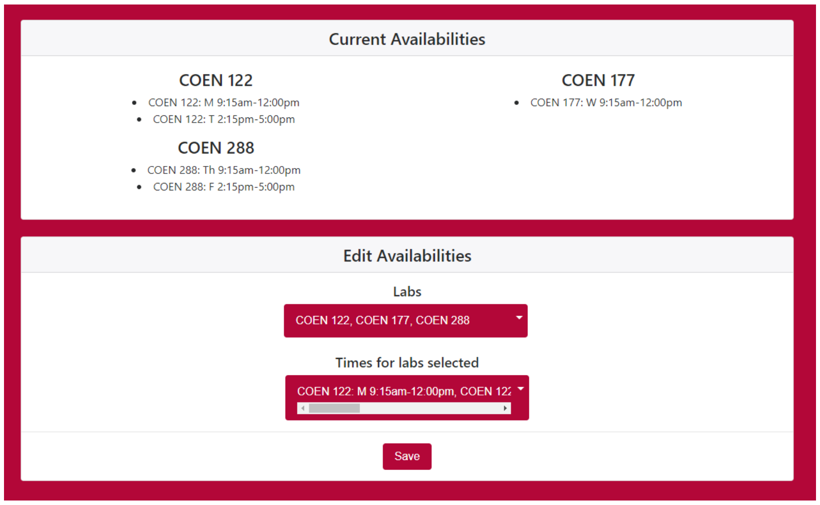

Current Availabilities
COEN 122
COEN 122: M 9:15am-12:00pm
COEN 122: T 2:15pm-5:00pm
COEN 288
COEN 288: Th 9:15am-12:00pm
COEN 288: F 2:15pm-5:00pm
COEN 177
COEN 177: W 9:15am-12:00pm
Edit Availabilities
Labs
COEN 122, COEN 177, COEN 288
Times for labs selected
COEN 122: M 9:15am-12:00pm, COEN 12
Save

Figure 11: Edit Schedule User Guide

When logging in as a returning TA, you will be presented with a page similar to that seen in Figure 11. To edit your schedule in order to add or remove a class/section in which you are no longer available for, refer to section ***C.2.1 Insert Schedule*** and follow the same procedure.

B.2.3 Send Absence Notification

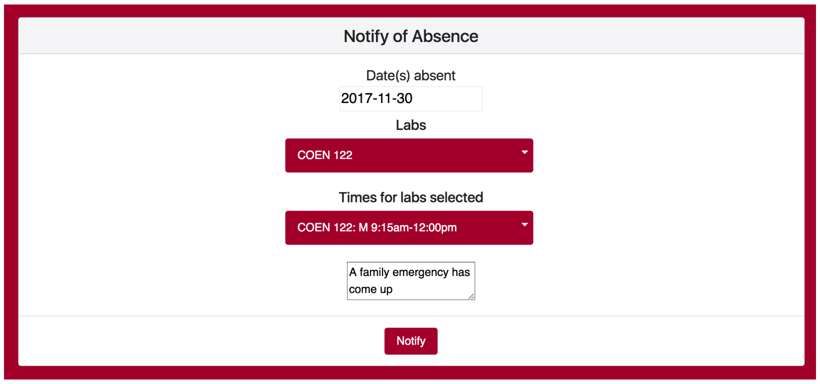


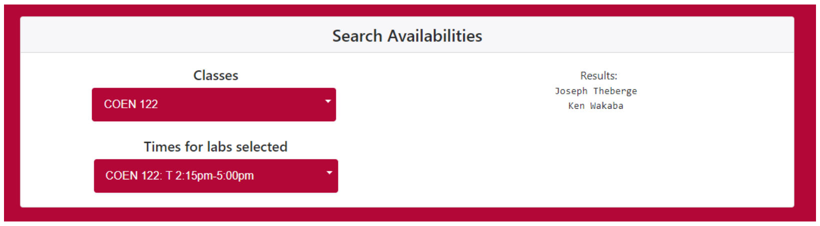
Figure 12: Send Absence User Guide

To notify the professor of a lab section you will be unable to attend, the TA will go to the **Notify Absence** portion of the page. To notify, all that is needed to be done is select the date(s) in which you will be absent and then similar to editing the schedule, select the lab and the section in which you will be absent. You must provide a reason of why you will be missing the lab section so that the professor may have a record of it. Once satisfied, press **Notify** and your entry will be sent to the professor.

B.3 Professor

Upon successful login as an admin, you will be directed to an admin page which will give you a the user options such as searching for an available TA, giving TA lab permissions, entering new labs and sections, and sending notifications to TAs.

B.3.1 View TA Availability and Find Replacement



Results:
Joseph Theberge
Ken Wakaba

Figure 13: Searching for Available TA User Guide

As an admin you have the ability to search for a TA who is available at a given lab section(s); This may be because another TA has reported that they will be absent at that given time and need a replacement. To search for a TA, you must first select the **Class(s)** in which the TA will be in. Upon selection, the next section labeled **Times For Lab Selected** will automatically update with all the sections for the selected class. Once a lab section has been selected, the right hand side of the Search section will automatically update a list of all TAs free at the selected class section as shown in Figure 13. Depending on what sections and classes have been selected, the list will continue to populate with the appropriate names (NOTE: this search option is dependent on the fact that their are TAs previously inputted into they system as well as that their schedule is up-to-date).

B.3.2 Give TA Permissions

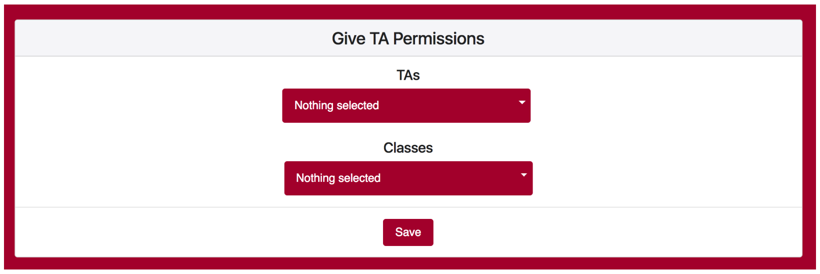
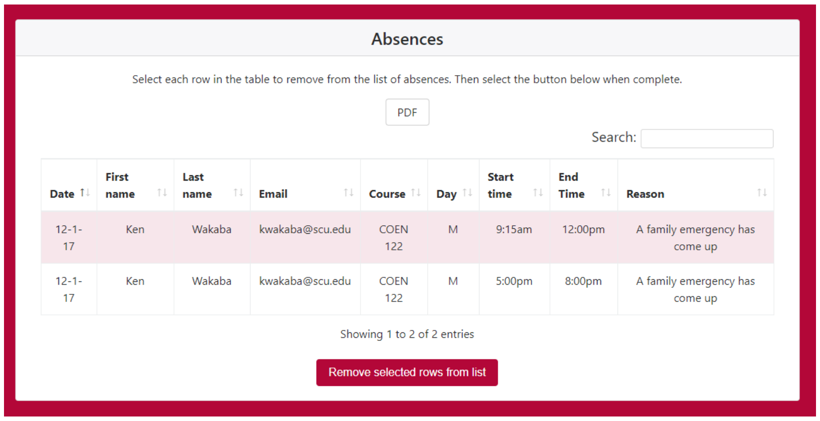


Figure 14: Give TA Permission User Guide

When a new TA creates an account, they are not able to do anything such as insert a new schedule because they will not have the ability to access selected classes. For them to be able to select classes, the professor must give them permission to do so. For the professor, they must log into the system and go to the **Give TA Permissions** section of the application. Once here, select the TAs you wish to give permission as well as the classes they are able to TA for. When satisfied, press Save to push the information onto the database. The TA will need to refresh the page and once they do so, they will have the ability to select the labs and times they are available, given they have proper access given by the professor.

B.3.3 View Absences



Absences

Select each row in the table to remove from the list of absences. Then select the button below when complete.

PDF

Search:

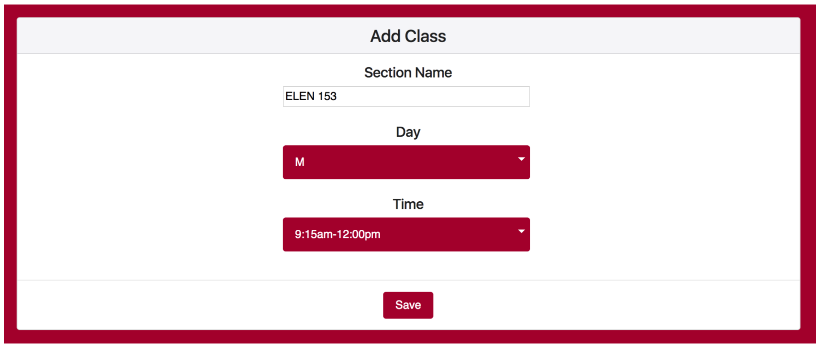
Date	First name	Last name	Email	Course	Day	Start time	End Time	Reason
12-1-17	Ken	Wakaba	kwakaba@scu.edu	COEN 122	M	9:15am	12:00pm	A family emergency has come up
12-1-17	Ken	Wakaba	kwakaba@scu.edu	COEN 122	M	5:00pm	8:00pm	A family emergency has come up

Showing 1 to 2 of 2 entries

Figure 15: View Absences User Guide

When the admin is logged in, they will be able to view a table of all the reported absences sent in by TAs. Using this table, they can see the dates and sections in which TAs will be unable to attend a lab. The admin, using the **View TA Availability** portion of the page discussed in C.3.1 will be able to find the replacements. Once the admin has found a replacement, they will have the ability to click on the email provided in the **View TA Availability** section, which will launch their native Mail application so that the admin can contact the TA in the application. To update the table of absences, simply select each row in which a replacement has been found, and click the button indicating to remove the buttons. This portion also has the added functionality of being able to download the entire table as PDF and saving/printing the file.

B.3.4 Add Classes



Add Class

Section Name: ELEN 153

Day: M

Time: 9:15am-12:00pm

Figure 16: Add Classes User Guide

Within this section, the admin will have the ability to add new classes and sections into the system. These classes are used for the TA when they are using the scheduling options or the notifications. The admin as well uses these classes in giving permission and finding availabilities. To add a class, simply input the class name (e.g. COEN 123), as well as the day the lab section will be held and the time. By selecting Save, the values are updated in the database and will be able to be used throughout the system.