Laboratory Help-Seeking System
Final Report

COEN 174L
Group 4

Nick Goodpaster, Bryson Lee, Sean Karstein

Table of Contents

Table of Figures

Table of Tables

Abstract

We have built a web application that improves the efficiency of the question asking and answering process in engineering lab sessions at Santa Clara University. Currently, students raise their hands and wait for the teaching assistant to answer them in person. This hampers both speed and fairness for students. Existing solutions like email lack the speed and personalization of our system. Our solution allows teaching assistants to quickly asses the most common questions, making sure that students are answered in a quick and fair manner. Our application streamlines the question asking process and allows for a more enjoyable lab experience for both students and teaching assistants.

Introduction

At Santa Clara University, the format of Engineering labs usually comprise of a group of students on individual workstations facilitated by a Teacher Assistant (TA). Students progress through lab assignments at different paces and may require periodic help from the TA. As a result, feedback from TAs is given on a first-come, first-serve, as-needed basis. This system proves inefficient, as often times, there will be multiple students with similar questions. TAs must also respond to student concerns by physically visiting their workstation, potentially causing them to lose mental track of priority between students. Ultimately, this workflow prevents TAs from responding quickly to simple questions or fairly queueing student concerns.

Currently, no technical solution is in place to optimize the student answering process. Email serves as a possible way for TAs to respond to student concerns, but it is not a reliable platform to sort/prioritize questions by content, nor a service that allows for quick responses to common questions. Other ticket-based services exist, but are not tailored to the needs of Santa Clara University's Engineering labs, and are primarily meant for corporate IT departments.

There are a few technical requirements that our solution must meet. It must be a web-based application, it has to run in either Firefox or Google Chrome, and it must run on the Linux machines in the Engineering Computing Center (ECC).

The goal of this application will be to decrease the amount of time wasted by the TAs as they answer questions from the students. Our solution is a web-based application that allows students to more effectively ask questions and receive feedback from TAs. For longer questions that require the TA to go to the student, the application will tell them who has been waiting the longest so they can answer questions in an efficient and fair manner. Another problem this application will solve, is the issue of answering duplicate questions. Our implementation will allow the TA to group similar questions together, so that they can address the class as a whole and save time from answering each question individually.
By having our system sort questions by priority, TA's can conduct their labs more quickly and efficiently. The ability to answer similar questions at the same time will drastically improve TA response time, easing the question asking process both for students and TAs. Asking questions is an integral part of the lab experience. Our proposed solution will allow for seamless and quick response to those questions.

Requirements

Our system has a few functional requirements which will help outline our design and implementation. Some of the functional requirements are critical in nature, while some are merely recommended or suggested. Our critical functional requirements are as follows. The system will allow students to ask the TA of their lab section questions about the assignment. The system will allow the TA to respond to the questions directly in the web application if they so desire. The system will keep a timestamp of when each question is asked so the TA can answer them in order of when they are asked. And finally, the system will host several private lab sessions at the same time so that only the students in a session will be allowed to ask the corresponding TA questions. Following the critical requirements, we have also decided on a recommended functional requirement and a suggested functional requirement. The lack of these specific functionalities will not hinder the overall functionality of the system, but will their presence will greatly aid the client in their use of the system. The recommended requirement is a little more important than the suggested requirement. The recommended functional requirement is that the system will allow the TA to group questions by keyword, in order to address similar questions at once. The suggested requirement is that the system will save answered questions for later reference. These two requirements are important to the system, but not critical in nature.

The system also must meet a few non-functional requirements. These requirements do not affect the functionality of the system and will not change the implementation, but will determine how the functional requirements are achieved. The non-functional requirements our system will meet are as follows. The system will be intuitive so users can operate it without training. The system will be efficient. And finally, the system will be universally useable.

Finally, the system also has a few design constraints that will limit the ways in which our system will be able to operate. We have determined there are three main design constraints for our system. The system must be a web application written with basic web technologies. The application must run on Santa Clara University's linux web server. And the application must run on both firefox and chrome browsers. By keeping these requirements and design constraints in mind, we are able to continue on with the planning of our design, confident we will not run into any major issues deep into development.

Use Cases

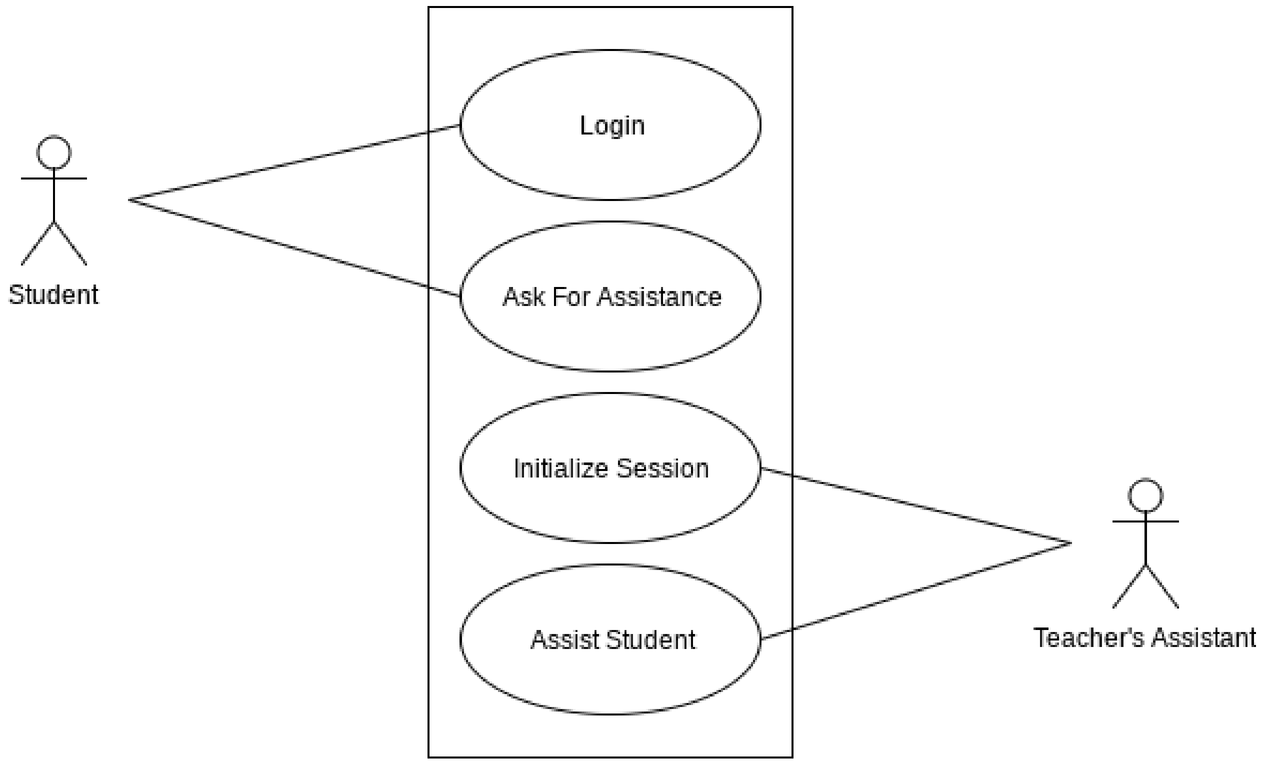


Figure 1 - Use Case Diagram

Table 1 - Use Case Descriptions

	Log In	**Ask for Assistance**	**Initialize Session**	**Assist Student**
Goal	Enter the correct lab Q&A session	Ask a question to the TA	Set up a Q&A session for the lab	Respond to a question that a student has asked
Actor	Student	Student	TA	TA
Preconditions	Student must have an account and know session number	Student must be logged in	TA must have an account	-TA must be logged in -Student must have asked question
Steps	-Enter username, password, and session number -Click login	-Enter question -Click send	-Enter username, password, and class name -Click login -Give generated session number to students	-Enter response -Click send
Postconditions	Student is added to session	Question added to database	Session is running	Answer added to database
Exceptions	Section number must exist	N/A	N/A	N/A

Activity Models

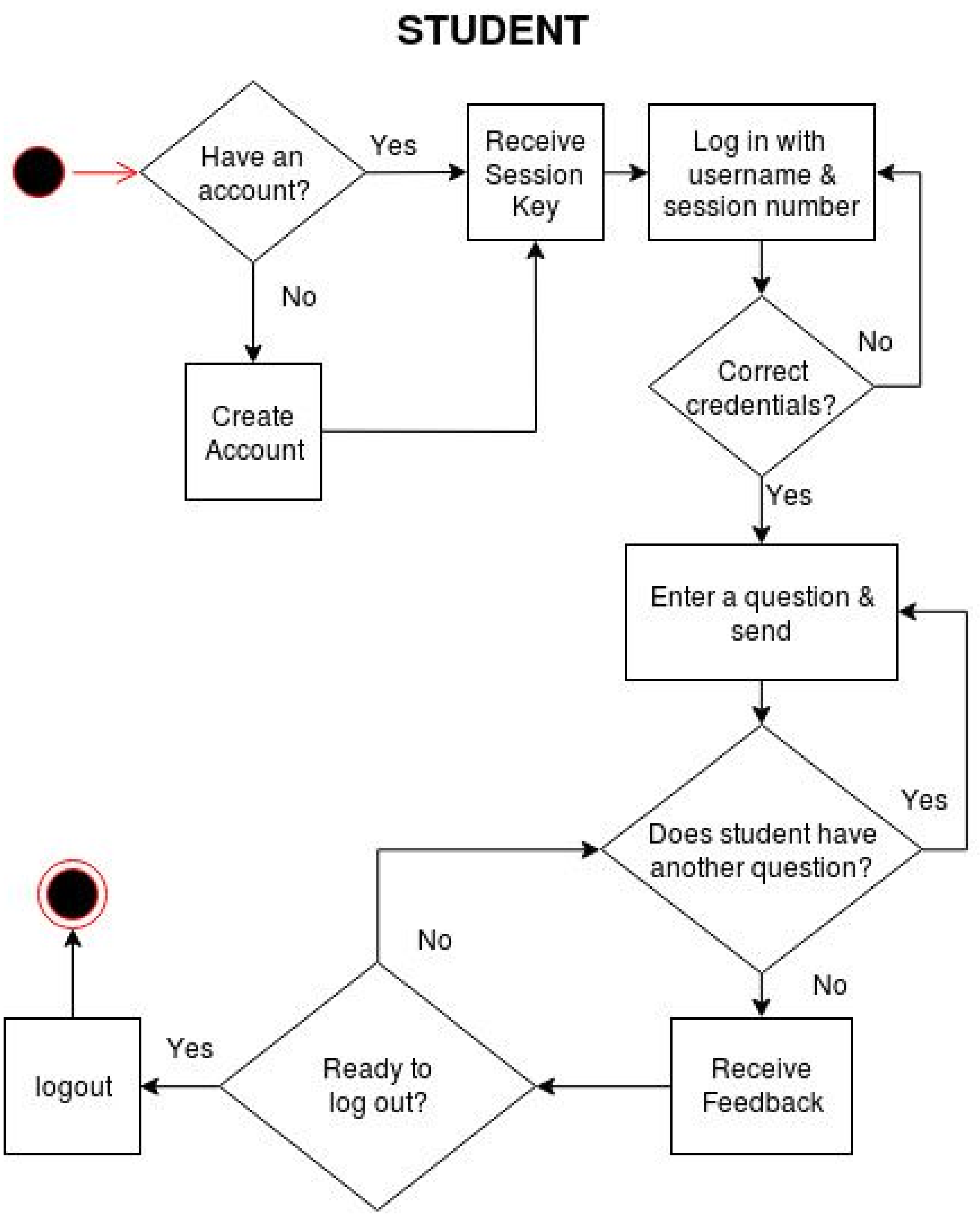


Figure 2 - Student Activity Model

TA

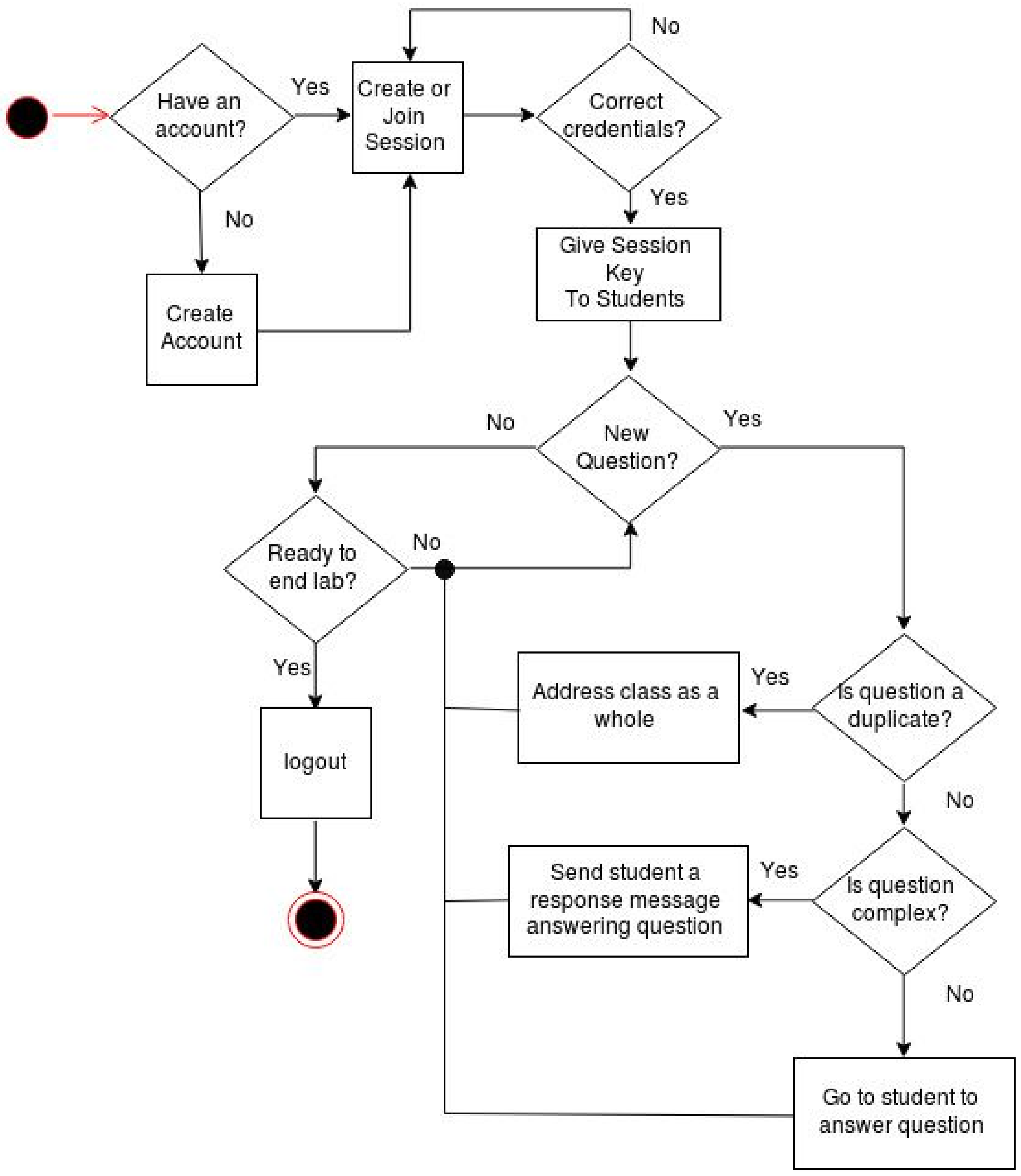


Figure 3: TA Activity Model

Technologies Used

- HTML
- PHP
- CSS
- JavaScript
- MySQL

Architecture Diagram

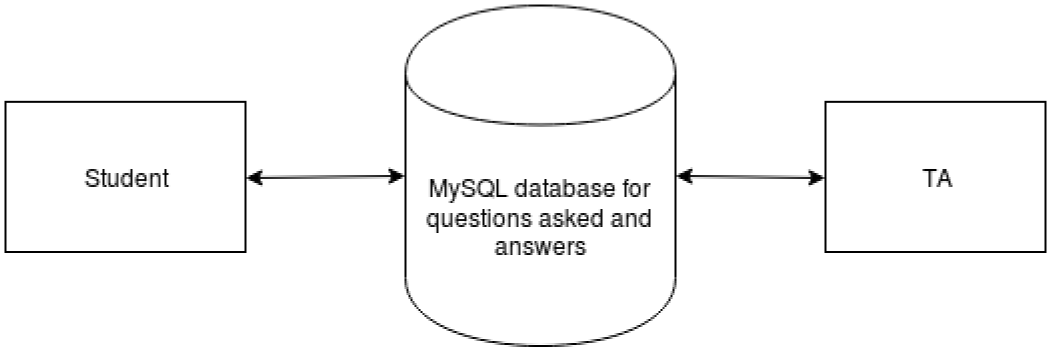


Figure 4 - Architecture Model

The architecture design the application is using is a data-centric design. The components are the database holding the questions and answers for each lab section, and the student and TA as clients using the database.

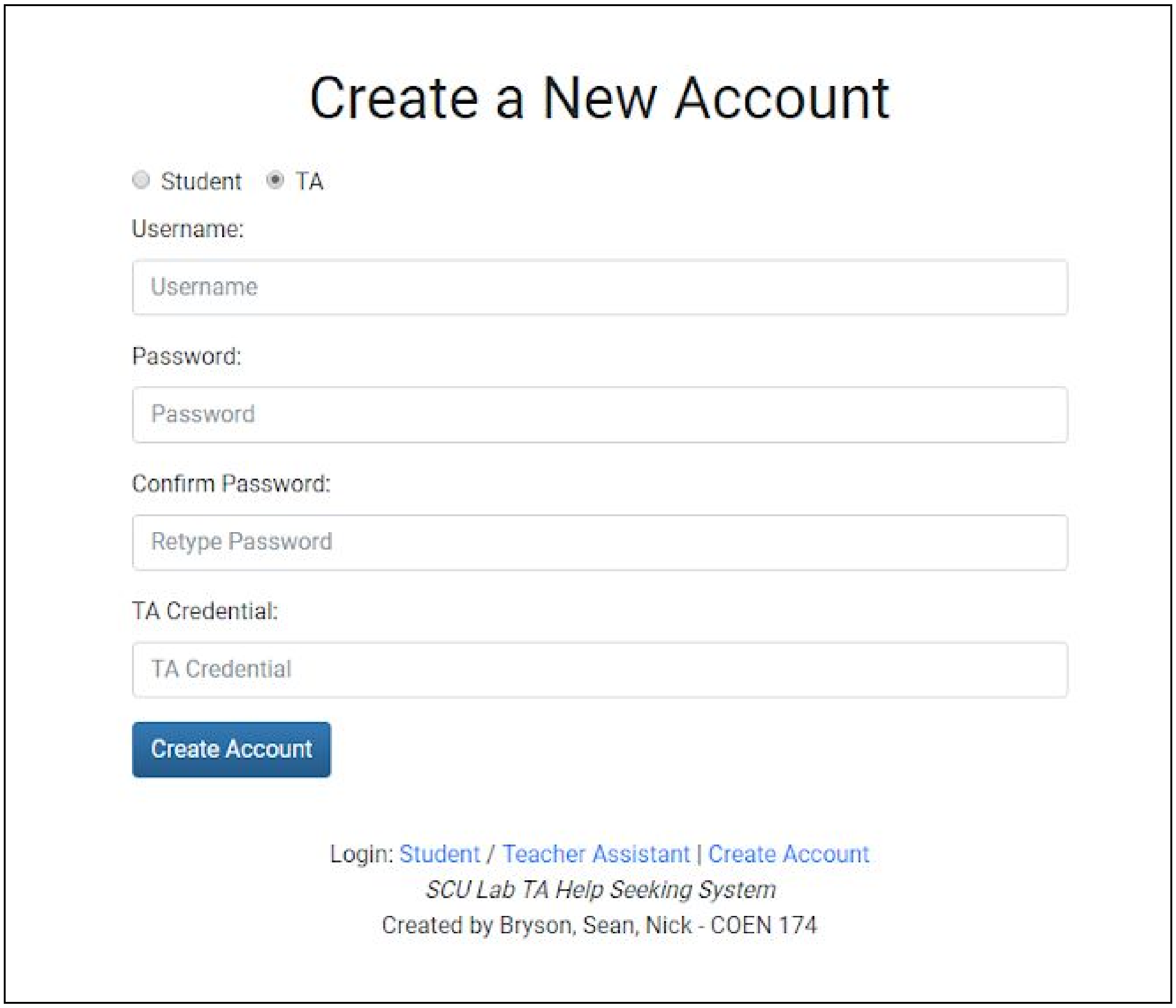
Design Rationale

The design of this application is primarily dictated by the requirements set forth by the project. We are strictly limited to using HTML, PHP, CSS, and JavaScript, despite more flexible and robust alternatives, and due to major restrictions in library and framework usage, we are only permitted to utilize a static PHP website paradigm and cannot explore REST architectures or server-based applications. In addition we will use MySQL for storing questions and student information. This relational database is already installed on the Engineering Computing Center Linux machines and will work well for storing information like questions with tags.

Description of System Implemented

Account Creation

Our system fulfills all of the critical functional requirements mentioned. The first page is the account creation page. This page allows either a student or TA to create a new account with a unique username. TAs will have to enter a TA credential set by the system administrator.



Create a New Account

Student [✓ TA]

Username:

Username

Password:

Password

Confirm Password:

Retype Password

TA Credential:

TA Credential

Login: Student / Teacher Assistant | Create Account

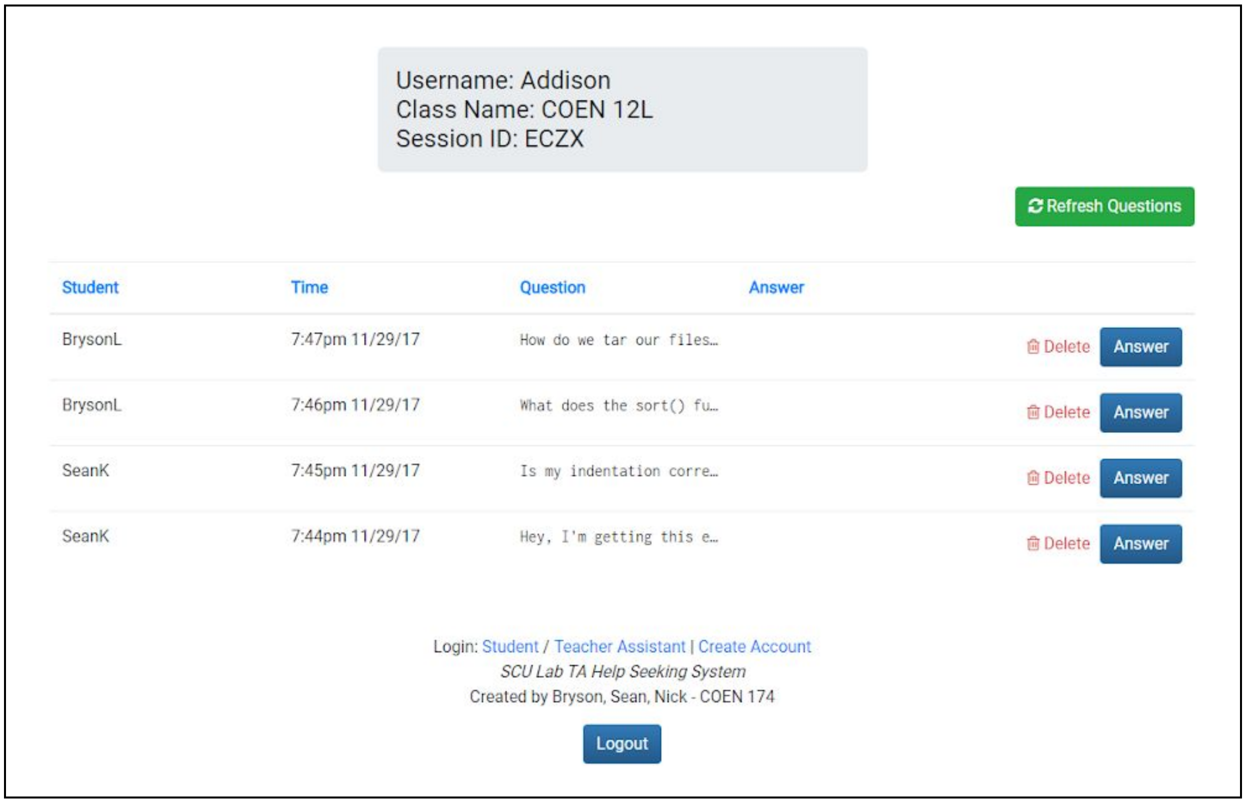
SCU Lab TA Help Seeking System

Created by Bryson, Sean, Nick - COEN 174

Figure 5 - Account Creation Page

TA Interface

Upon creating an account, the TA is prompted to log in. They can choose to either join an existing session or start a new one. Upon creating a session, a session ID is randomly generated. The TA gives the session ID to students so that they can login to their session and start asking questions.


Username: Addison
Class Name: COEN 12L
Session ID: ECZX

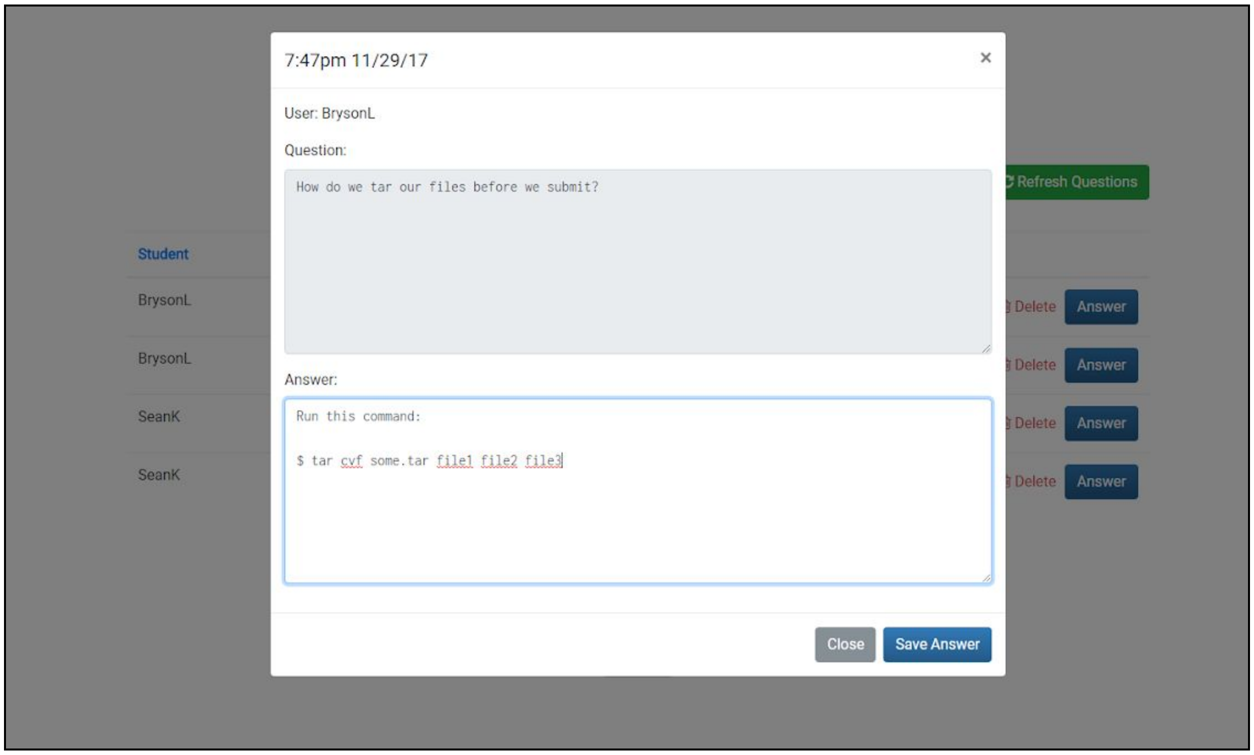
Student	Time	Question	Answer	
BrysonL	7:47pm 11/29/17	How do we tar our files…		Delete Answer
BrysonL	7:46pm 11/29/17	What does the sort() fu…		Delete Answer
SeanK	7:45pm 11/29/17	Is my indentation corre…		Delete Answer
SeanK	7:44pm 11/29/17	Hey, I'm getting this e…		Delete Answer

Login: Student / Teacher Assistant | Create Account
SCU Lab TA Help Seeking System
Created by Bryson, Sean, Nick - COEN 174

Figure 6 - TA Homepage

The TA homepage pictured above is student’s questions appear in a queue. Clicking the “Refresh Questions” button refreshes the page and displays new questions. Upon receiving questions, they can click on any of the headings to sort questions by that category in either ascending or descending order.

The TA can click the “Answer” button next to any question to begin their response. After clicking “Answer”, the window pictured below appears.



```
How do we tar our files before we submit?
```

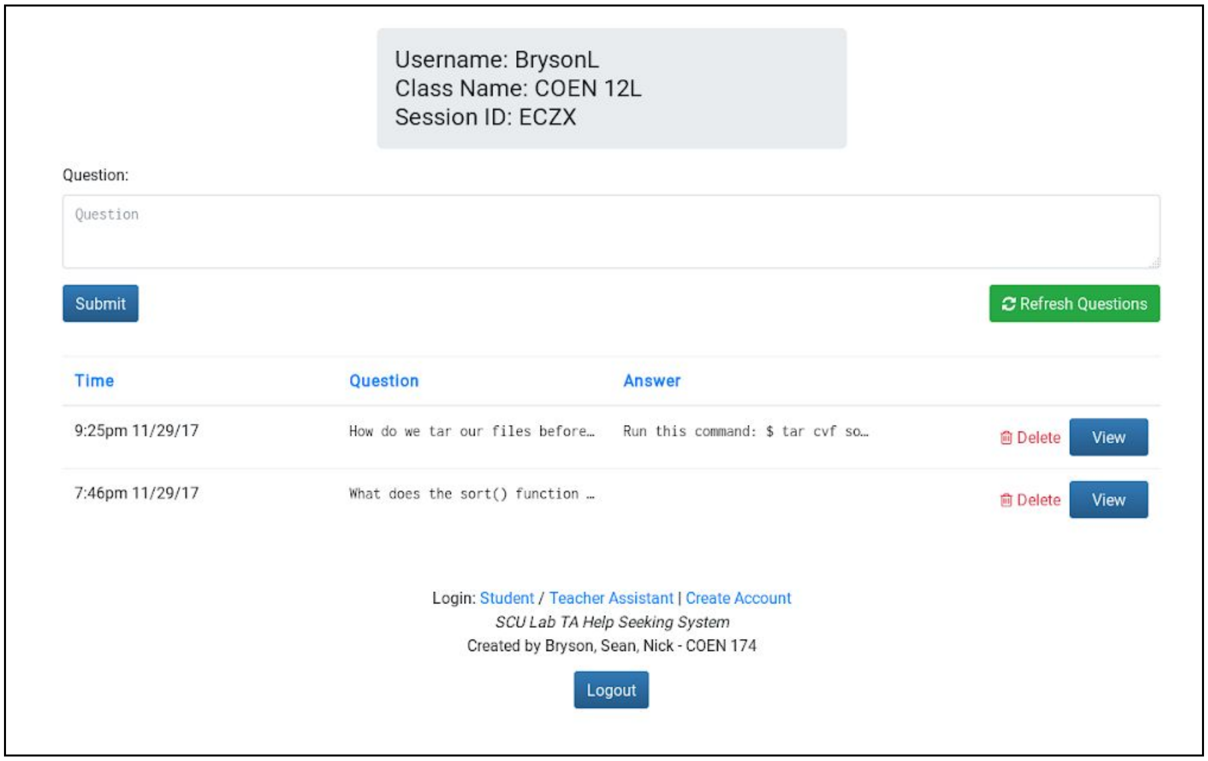
```
Run this command:

$ tar cvf some.tar file1 file2 file3
```

Figure 7 - TA Answer Window

The TA can then compose their response in the Answer section and click “Save Answer” to send the it to the student. On the homepage, if the TA wants to delete a question, they can click the red delete button to the left of the “View”/”Answer” button. This will open another modal, asking the TA to confirm their deletion. When the TAis done, they can click the “logout” button at the bottom of the screen to exit their session and return to the login screen.

Student Interface

In the student's case, upon creating an account, the student is prompted to log in. They enter their username and password, as well as the session ID given to them by their TA. Once they log in, they are taken to the student homepage pictured below.



Username: BrysonL
Class Name: COEN 12L
Session ID: ECZX

Question:

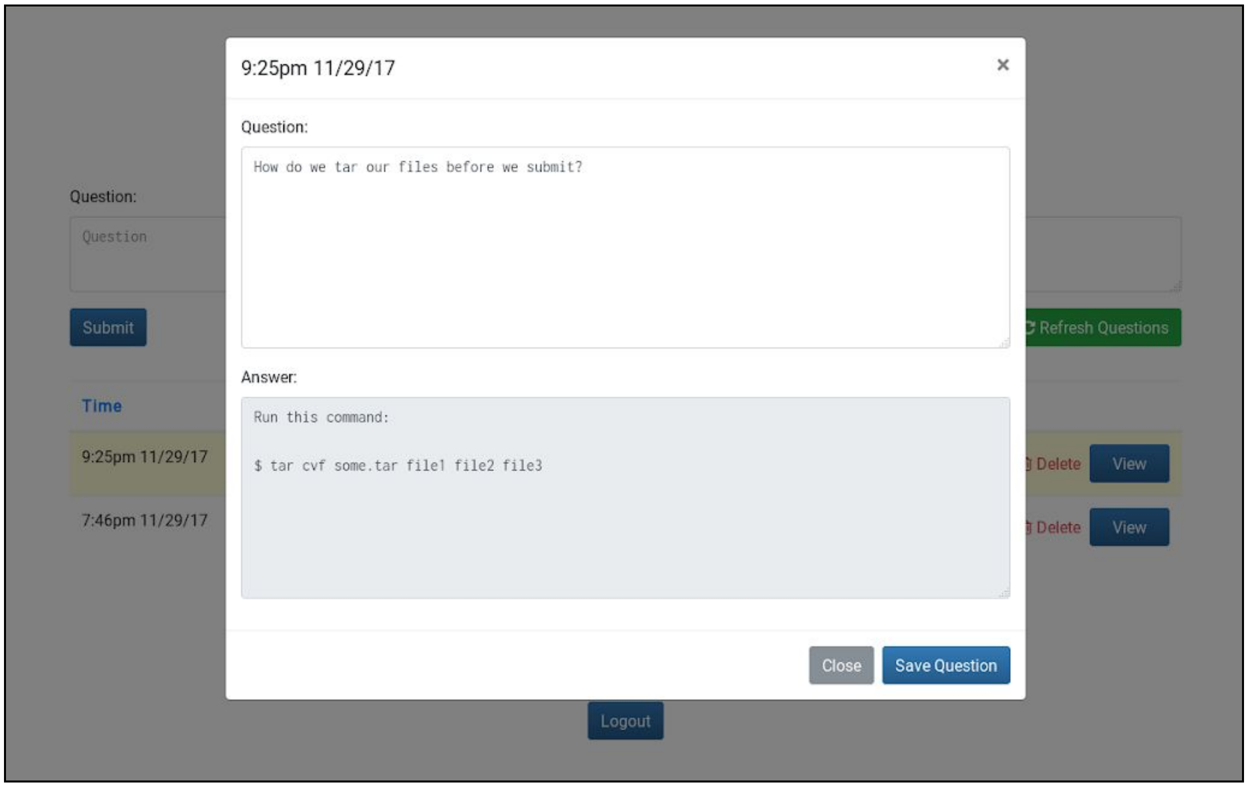
Time	Question	Answer
9:25pm 11/29/17	How do we tar our files before…	Run this command: $ tar cvf so…
7:46pm 11/29/17	What does the sort() function …	

Login: Student / Teacher Assistant | Create Account
SCU Lab TA Help Seeking System
Created by Bryson, Sean, Nick - COEN 174

Figure 8 - Student Homepage

On the student homepage (pictured above), the student's questions appear in a queue. Clicking the "Refresh Questions" button refreshes the page and displays any new answers form the TA. The student can click on any of the headings to sort questions by that category in either ascending or descending order.

If the student wants to edit their question or view an answer, they can click the "View" button next to any question. After clicking "View", this modal window appears:



```
How do we tar our files before we submit?
```

```
Run this command:

$ tar cvf some.tar file1 file2 file3
```

Figure 9 - Student Question Window

The student can then edit their question in the Question section and click "Save Question" to send the update to the TA. The student can also view the TA's answer within the Answer section of the modal. On the homepage, if the student wants to delete a question, they can click the red delete button to the left of the "View" button. This will open another modal, asking the student to confirm their deletion. When the student is done, they can click the "logout" button at the bottom of the screen to exit their session and return to the login screen.

Test Results

Regression Testing

Our project followed a very strict feature implementation workflow that was heavily version controlled and tested incrementally to ensure that our code properly adapted to new additions. Utilizing a Git and three separate development environments running three integration environments, we were able to run several instances of the same application in a live environment to properly test new features from a user and programmatic perspective.

Beta Testing

In order to test our system from a user perspective, we conducted beta testing. This was accomplished by allowing other students, who do not have knowledge of our code, to use our system. Without any instruction, we asked the students to create accounts, login, ask, and answer questions. We found that in some cases students were confused about the function of certain buttons. We used this data to change some of the names of our buttons to improve usability.

Difficulties Encountered

There were several difficulties that we encountered during this project. The most prevalent problem was our team's lack of PHP experience. Although we were familiar in developing with other languages and technologies, PHP follows a different paradigm of web architecture which was difficult to adjust to given our experience. We were also limited to PHP as one of the few viable web technologies that would allow us to build this application.

The second difficulty was adjusting to the requirements of ECC servers for hosting web applications. In particular, we had major difficulties designing and maintaining a flexible workflow for our development environment. Because deployment of our application was entirely dependent upon proper file permissions and structure within our ~/webpages directory, creating a workflow that allowed Git version control proved hard in practice.

Suggested Future Changes

There are several future changes we hope to implement that will improve the functionality and performance of the application. The first is a more robust TA validation system to ensure that only authorized TAs have the ability to create TA accounts and create new lab sessions. Currently, we utilize a modifiable environment variable for the TA validation key, however, TA accounts should be properly managed through some form of an administrative interface. Second, session deletion should be built into the TA question-and-answer interface. Ideally, this is to prevent old sessions from interacting with newer ones. Third, question keywords should be added as an additional field for when students ask questions so TAs can easily sort through existing questions. Fourth, a more standardized address routing should be introduced such that the SCU Lab Help-Seeking System does not need to rely on the ~/webpages URL and consequential system subdirectories.

Experience and Lessons Learned

Throughout the design, development, and testing phases of this project, we gained a great deal of experience and learned many lessons in the process. Before beginning, we had little to no experience with PHP, so this project allowed us to learn a new language and gave us the opportunity to gain valuable experience using it. We also gained important experience in developing a project from the ground up. We were able to both design and implement the system and see a conceptual model become a final product.

The experience we gained working on this project was greatly impacted by a few lessons we learned in the process. The first main lesson we learned is the importance of not rushing into implementation. When we first started implementing our design, we had not yet fully completed the design, and had not decided on the structure of the system. This caused us to have to backtrack a few times later on down the road - an avoidable issue if we had taken the extra time to fully complete our design to ensure we were prepared to begin implementation. Another lesson we learned was of the importance of clearly defining all requirements before beginning to design the system. We realized fairly late into our design phase that we asked far too few questions of the client during our requirement elicitation. Because of this, we found out we needed to meet more requirements than we originally thought, and as a consequence, we had to change aspects of our design late in the process. Finally we also learned to create a realistic development timeline and to stick to it. As we began to implement our system, we realized that we had set some unrealistic expectations regarding when we would finish each portion of the project. This caused us to have to re-think and re-plan our project to ensure we finished it in the given time. We also didn't do a great job of sticking to our timeline which may have caused the issue just mentioned. We procrastinated a few times, which cause a few late nights and extra hours leading up to deadlines.

All in all, this project taught us a lot about the importance of each step in the software development process and gave us the opportunity to gain real software experience, developing a product from the ground up.

Appendix

Installation Guide

Requirements

Installation of the SCU Laboratory Help-Seeking System is dependent upon five conditions:

1. *The installer or administrator has an active user account on the ECC system, or is permitted to set up an account on the ECC system*

By default, all SCU faculty, SCU students enrolled in a class in the Engineering school, and SCU students enrolled in a Computer Science class are given permission to set up a user account on the ECC systems. This user account is required to continue with the instructions in this installation guide, and must be properly set up in accordance with standards established by the ECC system administrators. The user account can be set up at any time by consulting the ECC help desk or ECC system administrator and following the steps that they provide.

2. *The installer or administrator tasked with installing the system is a student, a faculty member, or an authorized person permitted to host web pages in the Santa Clara University ECC ~/webpages domain*

In order to run the application itself, the student must have a /webpages subdirectory created under their specified ECC username, such that the /webpages/*username* folder exists (where username is the ECC username of the installer or administrator). More details concerning how to set up a webpages directory can be found at:

http://wiki.helpme.engr.scu.edu/index.php/Webpage

If the instructors provided by the link above do not work properly or you face difficulty in any part of the guide, one should consult the ECC system administrator for more help. This guide assumes that the Webpages directory has already been established. One can test whether the webpages directory is setup correctly by attempting to access, where *username* is the web pages directory of the ECC account that is/was set up. This is your login username by default.

http://students.engr.scu.edu/~username/index.html

If the above link shows an HTML page with “YOUR NAME’S Web Page” then your webpages directory has been set up properly.

3. *The installer or administrator is authorized to host an instance of the SCU Laboratory Help-Seeking System*

Only authorized individuals are allowed to host an instance of the SCU Laboratory Help-Seeking System. This system was created to only require one instance of the application to run for the entire school, and therefore multiple instances do not need to run and could potentially conflict the use of the application.

4. *Authorization to host an instance of a MySQL 5 database on the ECC center*

While the required technologies that are needed to effectively run the SCU Laboratory Help-Seeking System is already provided by the ECC system, you must have permission to run and have an active ECC MySQL remote database account and associated database instance. Creation of the MySQL account and database instance must be done in accordance with the ECC system administrator. More instructions can be found at:

http://wiki.helpme.engr.scu.edu/index.php/MySQL-5

After creation of this database, the installer will have been given a username, a domain for the remote database, a password for the account, and a database instance name for the application to run on.

5. *Proficiency with Linux Commands and the Terminal*

While not entirely necessary, it will help if the the user is familiar enough with Linux and the Terminal to follow the installation guide outlined here.

Downloading and Extracting the Source Code

Installation of the source code into your live webpages environment must be done on the ECC Linux Machines, as this is where the dependencies of the code live and where the commands can be run.

First, login to the Linux machine and insert the provided USB drive into your computer, transfering the scu-lab-seek-source.zip compressed folder to any subdirectory of your choice. This will be referred to as your *Working Directory* for the entirety of this installation guide. Your working directory should preferably be a directory that you have full permissions to read, write, and execute from in order to better interface directly with the code, which you will have to change during configuration. Open a terminal window and navigate to the working directory that holds scu-lab-seek-source.zip, and unzip the folder using the unzip command:

```
unzip scu-lab-seek-source.zip
```

This will extract the folder from your zip file and create a folder called scu-lab-seek-source that contains all the files to run the application, properly formatted in the correct file structure. Note: do not move any files in this directory, or change the name of any files. This is unnecessary and may cause problems when installing the source code. If you chose to you, you can run

```
chmod -R 777 scu-lab-seek-source
```

Although this sets universal permissions for your scu-lab-seek-source folder, a provided deploy script changes the permissions of all files to what it should be during deployment.

Configuring your Files

Navigate into the scu-lab-seek-source directory. There are 5 files that must be configured for your application to run properly after it has been deployed (assuming the current directory is the top level of the scu-lab-seek-source directory). Open the files below and follow the instructions for each:

`./php-cgi/dev/settings.php`
Change the `DEV_WEB_HOME` php variable to be your ECC webpages directory name. For example, if you access your webpages site is located at students.engr.scu.edu/~jsmith, `WEB_PAGE_HOME` is should be changed to “jsmith” (without the quotes).
Change the `TA_CREDENTIAL` php variable to be a secret TA credential key that is only to be given to TAs to create new accounts.

`./php-cgi/.htaccess`
Change `WEB_PAGE_HOME` to be your ECC webpages directory name. Do not remove the “~”.

`./php-cgi/partials/.htaccess`
Change `WEB_PAGE_HOME` to be your ECC webpages directory name. Do not remove the “~”.

`./php-cgi/db_php/.htaccess`
Change `WEB_PAGE_HOME` to be your ECC webpages directory name. Do not remove the “~”.

`./php-cgi/db_php/connect.php`
In this file, change the following fields depending on your MySQL account information provided earlier:
`username` should be your username for the MySQL database that was set up earlier.
`password` should be your password for the MySQL database that was set up earlier.
`dbname` is your database instance name you were provided by the system administrator.

Setting up the MySQL 5 Database

To use MySQL, you must first have a MySQL account created for you. This account is separate from your normal ECC account. To request one, please e-mail *support@engr.scu.edu*. Your MySQL username will be identical to your ECC username, but the password is independent. (By default it is set to your Access Card ID with all the leading zeroes).
Your database name will be of the format:

```
sdb_<username>
```

Once you have created a MySQL account, start by setting up your mysql environment with this command:

```
setup mysql5
```

Before running MySQL navigate to this directory:

```
/php-cgi/sql
```

Now run MySQL by executing:

```
mysql -h dbserver.engr.scu.edu -p -u <username> <db_name>
```

Then login with your password. (By default it is set to your Access Card ID with all the leading zeroes). Once the MySQL prompt is open, execute this command:

```
source create_tables.sql
```

Deploying the Source Code to your Webpages Directory

Note that in the top level directory of the scu-lab-seek-source source code folder, a script called deploy.sh has already been provided to automate the deployment of the application.

Within the terminal, navigate to the top level directory and run deploy.sh by using the command below, where WEB_PAGE_HOME is your webpage home directory name:

```
./deploy.sh WEB_PAGE_HOME
```

This will automatically install the application to your /webpages/WEB_PAGE_HOME directory, transferring your files from your current working directory to your webpages directory. The webpages directory automatically detects changes in the corresponding user's subdirectories and updates the live site.

The application should now be successfully deployed and accessed at:

http://students.engr.scu.edu/~WEB_PAGE_HOME/php-cgi/student_login.php

where WEB_PAGE_HOME is your webpages home directory. Open any browser of your choice and go to the link provided above, with the correct WEB_PAGE_HOME directory.

User Manual

Create An Account (TA and Student):

1. Open a web browser (either Firefox or Google Chrome).
2. Navigate to *http://students.engr.scu.edu/~**<YOUR_WEBPAGES_USER_NAME>**/php-cgi/create.php.*
3. Select either the Student or TA radio button at the top of the form.
4. Fill in the username and password fields with values of your choice.
5. If you selected the TA button, enter the TA Credential given to you by your System Administrator.
6. If the username is not already taken, the password meets the length requirement, and the TA Credential is correct, your account will be created.

TA User Manual:

1. If you already have a TA account, navigate to *http://students.engr.scu.edu/~**<YOUR_WEBPAGES_USER_NAME>**/php-cgi/ta_login.php*, and skip to step 4.
2. If you do not have an account set up, see the above section titled "Create An Account."
3. After creating an account, click the login button at the bottom of the page labeled "Teacher Assistant."
4. On the TA login page, begin by entering your username and password, and then selecting the action you want to perform.
    a. If you want to join an existing session, select the "Join" option.
    b. If you want to create a session, select the "Create" option, and skip to step 6.
5. If you selected "Join", enter the Session ID of the session you would like to join in the next box.
6. If you selected "Create", enter the class name of the session you would like to create in the next box.
7. Once all fields have been completed, click the "Login" button.
    a. If all the information entered is correct, you will be logged into the system and redirected to the TA questions page.
8. Inside the questions page, you have the ability to view and/or answer existing questions (step 9), delete existing questions (step 10), refresh the question table (step 11), and logout (step 12).
9. View/Answer a Question:
    a. To view and/or answer a question, first click the blue button either labeled "View" or "Answer" at the end of the table row of the question you are interested in.
        i. The button will say "View" if the question already has an existing answer.
        ii. The button will say "Answer" if it has not been answered yet.
    b. Upon clicking the button, a dialog box will appear containing the following elements:
        i. The time and date the question was asked.
        ii. The username of the student who asked it.
        iii. The question content.
        iv. A text box for the answer.
    c. If you want to answer the question, or update the existing answer, type the answer into the answer text box, and click the "Save Answer" button. Skip to step 10.
    d. If you do not want to answer the question, you can either click the "Close" button, or the "x" in the upper right corner of the dialog box.
10. Delete an Existing Question:
    a. To delete an existing question, first click the red button labeled "Delete" directly to the right of the answer content for the question you are interested in.

    b. Upon clicking this button, a confirmation dialog will appear.
    c. If you are sure you want to delete the question, click the blue “Delete Question” button at the bottom of the dialog. Skip to step 11.
    d. If you do not want to delete this question, you can either click the “Close” button or the “x” in the upper right corner of the dialog box.

11. Refreshing the Question Table
    a. If at any point in time you want to refresh the questions table to see if new questions have been asked, simply click the green button labeled “Refresh Questions” located directly above the questions table.
12. Logout
    a. If you want to logout at any time, click the blue button labeled “Logout” located at the very bottom of the page.
    b. Upon logging out, you will be redirected back to the TA login page.

Student User Manual:

1. If you already have a Student account, navigate to *http://students.engr.scu.edu/~**<YOUR_WEBPAGES_USER_NAME>**/php-cgi/student_login.php*, and skip to step 4.
2. If you do not have an account set up, see the above section titled “Create An Account.”
3. After creating an account, click the login button at the bottom of the page labeled “Student.”
4. On the Student login page, enter your username, password, and the Session ID of the session you wish to join.
    a. The Session ID should be provided to you by your TA at the beginning of the lab session.
5. Once all fields have been completed, click the “Login” button.
    a. If all the information entered is correct, you will be logged into the system and redirected to the Student questions page.
6. Inside the questions page, you have the ability to view existing questions and answers (step 7), delete existing questions (step 8), refresh the question table (step 9), and logout (step 10).
7. View Questions and Answers:
    a. To view your questions and their answers, first click the blue button labeled “View” at the end of the table row of the question you are interested in.
    b. Upon clicking the button, a dialog box will appear containing the following elements:
        i. The time and date the question was asked.
        ii. An editable text box with the question content.
        iii. The answer content (if it exists).
    c. If you want to edit your question, type the changes into the question text box, and click the “Save Question” button. Skip to step 8.

    d. If you do not want to edit your question, you can either click the “Close” button, or the “x” in the upper right corner of the dialog box.

8. Delete an Existing Question:
    a. To delete an existing question, first click the red button labeled “Delete” directly to the right of the answer content for the question you are interested in.
    b. Upon clicking this button, a confirmation dialog will appear.
    c. If you are sure you want to delete the question, click the blue “Delete Question” button at the bottom of the dialog. Skip to step 9.
    d. If you do not want to delete this question, you can either click the “Close” button or the “x” in the upper right corner of the dialog box.
9. Refreshing the Question Table
    a. If at any point in time you want to refresh the questions table to see if one of your questions has been answered, simply click the green button labeled “Refresh Questions” located directly above the questions table.
10. Logout
    a. If you want to logout at any time, click the blue button labeled “Logout” located at the very bottom of the page.
    b. Upon logging out, you will be redirected back to the student login page.