CP476 FINAL PROJECT: PHASE IV – A+ TUTOR DOCUMENTATION

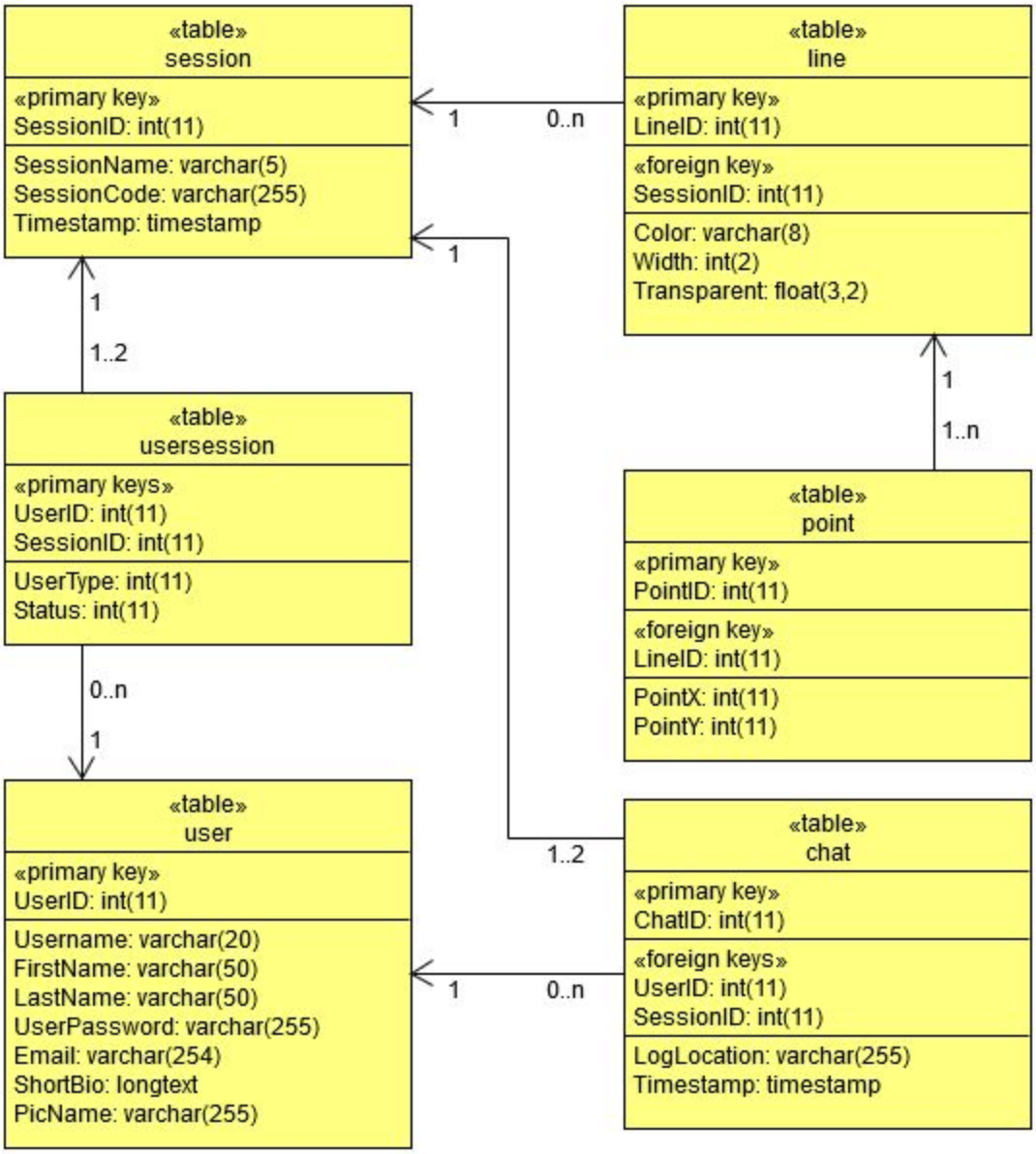
David Moreno-Bautista
151925580
more5580@mylaurier.ca

Xiang Ke
150617130
kexx7130@mylaurier.ca

Website Functionalities

Functionality	**Associated File**	**Included / Referenced Files**	**Associated Database Tables**
Signup and Login	landing_login_signup.php	-landing_process.php	user
Navigate Website (outside of classroom session)	include/header.php	-js/script_header.js -create_session.php -join_session.php -logOut.php	user session usersession chat
Navigate Website (within classroom session)	include/header_session.php	-browseInClass.php -endClass.php -js/script_whiteboard.js	user session usersession chat
View Active and Past Classroom Sessions	landing_home.php	-load_courses.php	user session usersession
View Profile	profile.php	----	user
Edit Profile	profile_edit.php	-profile_edit_process.php	user
Draw and Chat	whiteboard.php	-js/script_whiteboard.js -processChat.php -create_points.php -undo.php	user session usersession chat point line
Learn How to Use the Website	help.php	----	----

Database Design



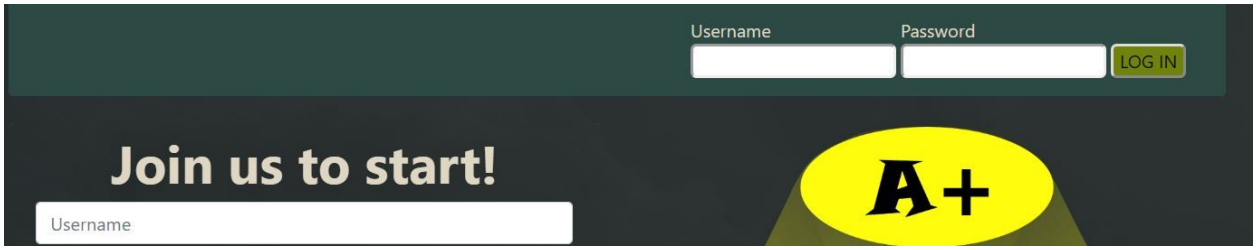
Notes

- We have exported the database to the file **127_0_0_1.sql** if needed for testing.
- In usersession table: UserType 1 is assigned to the creator (teacher) of the classroom session, and 2 is assigned to the guest (student) of the classroom session.
- In usersession table: Status 1 is assigned to an ongoing classroom session, and 0 is assigned to a terminated classroom session (it is terminated when the teacher clicks “end class”, logging out does not terminate it).
- In user table and chat table, “PicName” and “LogLocation” refers to the files associated with each, the actual files are stored on the server.
- Each classroom session has a maximum of 2 users (a teacher and a student) and therefore a maximum of 2 chat-logs associated with it (one for each).
- Each user has an unlimited number of classroom sessions and therefore an unlimited number of chat-logs.

Technologies Used

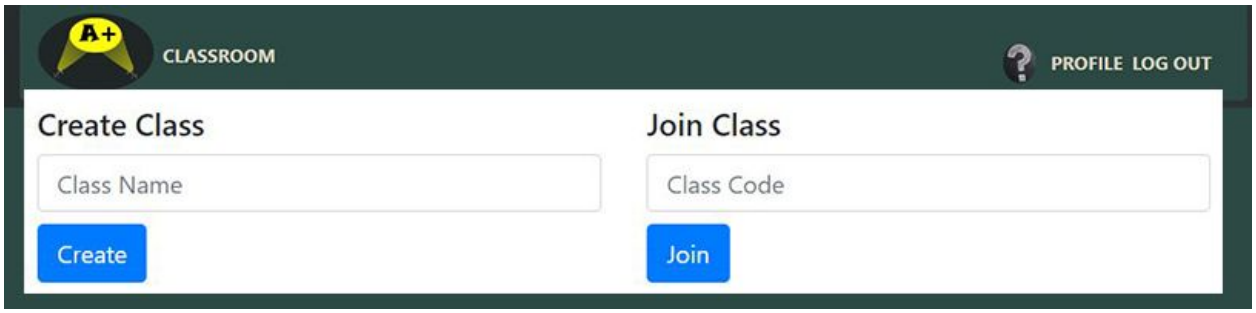
- HTML/CSS
- Bootstrap
- Canvas API (drawing)
- Javascript
- JQuery
- Ajax
- PHP
- MySQL

How It Works



Signup and Login

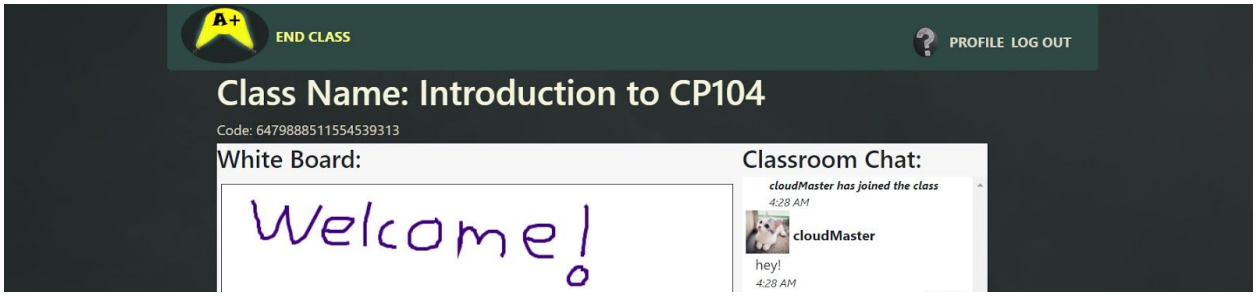
- Notable technology: PHP, MySQL
- When a user enters the website, they have the option to sign up or to login. When this is successful, the MySQL database is accessed and a PHP session is created to store user-specific information for the remainder of the user's browsing.
- During a sign up, the user information is stored in the database; and during a login, the user information is retrieved from the database for both validation and session creation.
- This relationship between PHP and MySQL is integral for the user experience, which is why it was chosen for this purpose.



Navigation (outside the classroom)

- Notable technology: Javascript, JQuery, Ajax
- The navigation bar links to the various pages of the website: home, classroom, help, profile, and logout. The classroom link implements a Bootstrap Javascript modal, which has the create and join class form.
- Create Class sends a POST request to set up the database tables for the classroom session and the chat table, which will be used to store logs for the conversations. The user is given the privilege to end class, since they are the creator.

- Join Class sends a GET request using jQuery's Ajax. If the code entered is valid, the user's PHP session is given the appropriate privileges and chat log location taken from the database. The chat-logs of both members of the classroom session are updated with the user-joined status, and the user is sent to the classroom after the appropriate JSON response is sent back.
- All of the links in the webpage that lead to the classroom have a Javascript event listener, which adds responsiveness by lighting up the spotlights upon hover.
- The asynchronous Ajax requests needed for the join session form make it essential for our project, and that's why it was chosen.

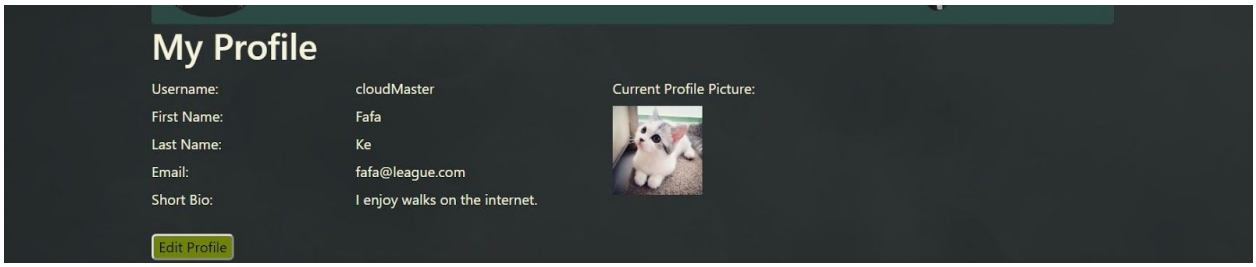


Navigation (within the classroom)

- Notable technology: PHP, MySQL
- The navigation bar is different when inside the classroom session; the spotlights in the logo are lit up, and every link away from the classroom first updates the chat-logs of both members of the class with a "user is not paying attention" message. Then the user is taken to the particular page that they clicked to access.
- The creator of the classroom session also has an "end class" link, which terminates the classroom session by updating its status in the usersession table of the database.
- All of the links in the webpage that lead away from the classroom have a Javascript event listener, which adds responsiveness by dimming the spotlights upon hover.

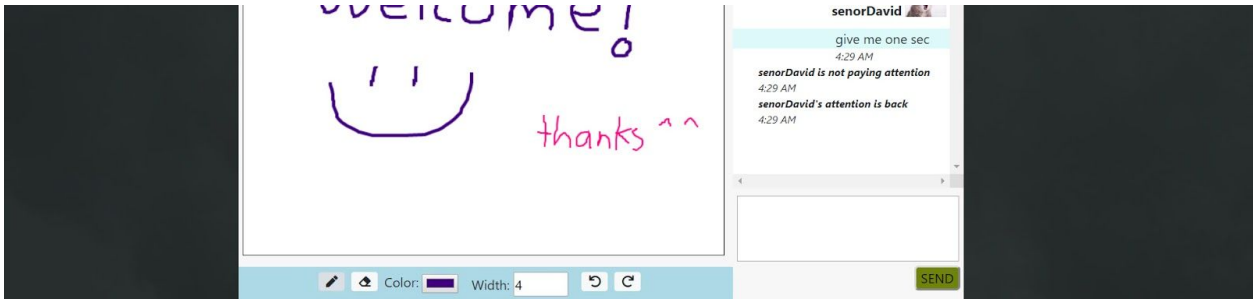
Home Page

- Notable technology: PHP, MySQL
- The homepage is where a user's classroom sessions are displayed for them. There is a welcome message and a table for for each type of session: session attending, session teaching, and session completed. Sessions in the first two categories have links to their appropriate classrooms.
- All of the information displayed is taken from the database with the aid of the user's PHP session keeping track of their credentials.



Profile Page

- Notable technology: PHP, MySQL
- A user's profile page displays all of their information that they've entered during sign up. They have the option to change everything except for their username by clicking on "edit profile".
- Similarly to previous pages, the database is accessed and updated with the help of PHP and MySQL queries.



Classroom Page

- Notable technology: Canvas API, jQuery, PHP, MySQL
- The classroom page is the main feature of the website. The chat works by accessing the database and updating the chat-logs of both the classroom host and the guest (also referred to as the teacher and the student). Your own personal message is aligned to the right with a blue background on your own chat-log, while it is aligned to the left with a regular background on the other member's chat-log. The chat-log for yourself is loaded on the page and refreshed every 2500ms with the use of jQuery, providing a real-time experience.
- The whiteboard works by first setting an appropriate line width, colour and transparency as Canvas API attributes. Mouse events, handled by jQuery, save the mouse coordinates during a user's strokes, which are sent to the database. Points are grouped into lines, which make up the whiteboard content.
- The lines are also pushed onto a stack on the javascript side. Users have the ability to undo or redo their strokes, which is implemented by setting the transparency of the lines kept track of by the stack. Erasing is implemented by drawing white color as the stroke response for the eraser. Similarly to chat, the content of the board is refreshed but at a faster rate of 100ms.

How To Use

- Documentation for the users on how to use the application is presented on the help page of the website.

Known Problems

- The real-time drawing **does not work as expected on Firefox**; there seems to be an issue that causes the refreshing of the board and chat to lag and flicker.
- However, everything is **verified to work as expected on Google Chrome**.

The Development Experience

The Idea

- This website was inspired by Python Tutor; they have a “Start private chat” option, which gives you a link that can be sent to another person. You are then both able to code in real-time on the same window as well as chat.
- This is similar to our website, as the user is able to share the code of the classroom session with someone else and have their real-time involvement within the same window.
- On the other hand, our website focuses on drawing as the interactive component rather than real-time programming, since it is designed to better resemble a real-life classroom.
- Furthermore, the chat in our website is more personalized, since users are able to create their own profile and have their username and picture displayed.

The Challenges

- There were multiple challenges that we faced while developing the website. As an example, HTML comments within a PHP file seem to interfere with AJAX calls coming from Javascript. We had to remove the header comments in those files, but it took a while to realize the issue.
- Both of us had little experience with real-time chat and drawing board implementations; everything we know about internet computing is what we’ve learned in class. Thus, we had to do some research in order to fill in the gaps.
- Online tutorials helped us find the Javascript functions that proved useful in the refreshing of both chat and drawing board components (setInterval). We also discovered Canvas API, included in HTML, as the right tool for the implementation of drawing.
- Time management was another obstacle, since other courses and responsibilities often interfere. As a result, it was difficult to coordinate with each other, but we were able to get our designated tasks done on a revised timeline.
- As a whole, the most time-consuming phase of the project was the initial design of the overarching website structure. Afterwards, implementing Ajax calls and making the classroom work in real-time were the next difficult phases.

The Future

- As this development phase comes to completion, we have ideas on how to further extend the features of the website. With the drawing board, It would be great to add more options within the toolbar.
- Some potential inclusions are the drawing of different shapes, adding text within the board, and adding images to the board.
- Also, since each drawing session has only one board, having multiple boards on different panels would be another consideration. With this option, going back and forth between different boards would further simulate a real-life classroom environment.
- Finally, another great addition would be the ability to export the board and chat. This would allow the user to save any discussions or drawings locally for future use.

Website Demonstration

- The following is a list of sample users that we have added to the website hosted on hopper; they have ongoing classroom sessions with each other, and have their own unique profile. The purpose is to demonstrate how the website performs with a populated database.
- The password for all of them is **password**

Fafa
David
LikesToLearn
Gamer
Hacker
CuriousLlama
CatLady
SportsFan