Detailed Implementation Document
nerdForum - COSC360 Project Group #31
Jason Schaad, Jasper Looman, Lily Stussi, Nick Schaad
April 14th, 2021

1. User Implementation

1.1 - Login/logout Page:

- The first implementation to the forum was the login page. Here users are able to login to their accounts using a username and password. Both username and password fields are validated using javascript, checking if both of these fields are not empty. Once the login form is submitted, and if these fields are non-empty, the form passes this information to a login processing page via a POST method. The login processing page grabs the POST variables sent by the form, makes a connection to the database, and uses prepared statements to check the username and md5 hashed password combination against those in the database. If a match is found, session variables are set for the user and they successfully log in, redirecting to the index page. On logout, the session variables are all cleared, and the session is destroyed. Users will see a message showing that they have successfully logged out.

1.2 - User Account Creation:

- In order to post replies and content on nerdForum, you are required to have an account. Under the login form, users will see a link reading “create new account” that brings them to the account creation page. This account creation page also serves as an edit/update user page, as it autofills the account information when accessed by a logged-in user. When creating an account, users are prompted for basic information as well as an image. When submitting the form, fields are checked to be non-empty and valid using javascript. For example, If the entered passwords do not match, or if a field is left blank, the form will not submit and the respective fields will turn red. If successful, the information is passed to a processing page where we insert the information into a ‘users’ table in the database. Once this is complete, the process page shows their account was created, and the user can now login to their new account.

1.3 - Editing profiles:

- In the header of the website, clicking “Welcome firstname lastname” will bring you to the edit user page. The user’s ID (stored as a session variable) is passed to this page via a GET method. On this page, we make a connection to the database, and sql prepared statements are used to retrieve all user information for the passed ID. This information then autofills in all user information. From here, users are able to change anything about their account (i.e. username, password, first/last name, email, image). Once the 'update' button is pressed, the form is submitted and the changes are made on a processing page. If successful, users will see a message confirming their changes went through, and that they may have to login/logout for some changes to take place.

1.4 - Creating a post:

- ❖ One of the main things users are able to do on nerdForum is create new forum posts. Once logged in, users can navigate to the posts page by selecting a category. On the forum posts page, all current posts are displayed in a table, and a form is displayed underneath prompting users for a title and content. Once the form is submitted, the form calls a process page that ensures the server method is a POST, and then uses sql to insert the submitted information into the database. CURRENT_TIMESTAMP() is used in the sql statement to store the date the post was made. If the post is successfully added to the database, the user is redirected back to the forum posts page. The new post will be visible at the bottom of the table on the forum posts page.

1.5 - Replying to posts

- ❖ Users are also able to add replies to any forum post. Once clicking a post, users are brought to a new forum reply page showing a post title, content, user who posted it, and the user image. All replies are shown below this, with each reply also showing content, the user who posted it, and user image. Underneath all of the replies is a form to add a new reply. If a non-user tries to access this page, the form to submit a reply will not be visible due to implementation of a securityLevel user attribute. Similar to creating a post, once the form is submitted, the form calls a process page to insert the reply into the database. If successful, the user is redirected back to the forum replies page, where they can see their new reply at the bottom of the thread.

1.6 - Collapsible posts/replies:

- ❖ At the top of both the forum posts and forum replies pages, users will see a button that reads: 'collapse posts. Upon pressing this button, a jquery script is called that changes the button to now read: 'show posts', and collapses all posts so the form to submit a new post is at the top of the page. Pressing the 'show posts' button causes the posts to reappear. The button functions the same on the forum replies page, but only collapses replies to the post, and not the actual post itself (i.e. title/content/user who created post are not collapsed).

1.7 - Search Function

- ❖ Regardless of if a user has an account or not, users are able to search the forum. Clicking the 'Search' text in the header will bring users to a search page. On the search page, two links are displayed, bringing you to pages to a search by user page, or a search forum page.
    - ➢ The search user page displays a form with all users displayed in a drop-down menu. Selecting a user and submitting the form calls the same page again, but this time with the selected user's ID passed to the page via a

    GET method. The page uses several sql statements to construct tables showing all replies/posts made by the selected user. General error handling and sql prepared statements implemented for security.
  - ➢ The search forum page allows users to search for posts or replies. On the search forum page, there is a form with a text box prompting you to search via keyword. Upon submitting the form, the same page is called again with the keyword passed via a GET method. The %like% operator in sql is used to find any post titles, post content, or reply content matching the given keyword. All matching posts/replies are then shown in dynamically-created, separate tables.
- ❖ For either search function, once the search finds results, clicking the icon in the "Details" tab will link you to the post that was selected.

1.8 - Breadcrumb Trail:

- ❖ On each page, variables named $PAGENAME and $BREADCRUMB are used to store the page name, and to store the location and name of the page in an associative array. In the header, php code is run to display the breadcrumb trail indicating your page location, and the path to get there. The php code runs a for loop that displays the name of the directory you are in as a link, and any subdirectories as a link if applicable. For example, clicking on the admin page, then the activity by date page will make the breadcrumb display: ' Home > Admin > Activity by Date'.

1.9 - Header:

- ❖ The header serves as an easy way to navigate to some vital pages on nerdForum. When the navigation links are hovered over, they are underlined. The header dynamically changes based on what kind of user is accessing the site. For example, if a unregistered user is accessing the site, the header will display a home link, a search link, and a login link. If a user who is logged on accesses the site, the header will contain an additional link where the user can edit their profile. This is done using a securityLevel attribute for users, and logging this as a session variable throughout the site.

1.10 - Hot Posts:

- ❖ On the nerdForum index page, under the post categories table is the 'hot posts' table. The SQL involved to create this table is based on the number of replies on a post in the past 24 hours. The posts will be sorted in descending order based on the number of replies. Only two posts are able to be shown at once, ensuring those posts are definitely the most active posts in the last 24 hours. The index page checks to ensure rows are returned in our SQL query, and if they are not, a message displays that there are no hot posts currently. Users are able to navigate to the 'hot posts' by clicking the text links that appear in the table.

2. Admin implementation

2.1 - Admin Panel

- The admin page can be reached by clicking the ‘Admin’ link in the header. This page allows admins to do many things that regular users are not able to do. This includes: a table view of all users with filtering ability, direct links to editing users, and viewing user activity by date.
    - On the admin page, admins will see a form at the top of the page, prompting for a username, first name, last name, and email fields. Below the form, each user’s information is displayed in a table by default. When information is entered and the form is submitted, the page calls itself, but passes the keywords in the URL as a GET HTTP method. The page now connects to the database and uses the SQL %LIKE% operator to filter the results. The table will now display any matching users.
    - Similar to the search page, there is a ‘details’ column of the user information table on the admin page, and clicking this icon will bring you to the edit user page, with the selected user’s information autofilling the edit user page. This allows admins to easily change other user’s information and resolve user issues.
    - Near the bottom of the admin page is the ‘View Activity by Date’ tool of our admin page. This form uses the HTML5 “date” input type, and allows admins to select a calendar date from a built in calendar. Upon submission of this form, the date is passed as vis method = POST to the activity by date page. The activity by date page accesses the database, and with use of prepared statements, two tables are created showing all posts, and all replies, that were made on the selected day. If an empty date field is entered, the page displays an error message and does not proceed. If no posts are made on the selected date, error messages are shown reflecting this.

2.2 - Edit/Remove posts:

- On the forum posts or forum replies page, if you are an admin, you will see an additional column appear in the table where posts are displayed. This is dynamically displayed based on the securityLevel session variable we set. In this column are buttons to edit or remove a post.
    - By selecting ‘edit’, admins are redirected to a new page where admins are able to edit the post selected. The post is accessed by passing the ID to the edit post page. The edit post pages looks like a form where you can change a post title or content, with the previous content autofilled in the form. Upon making changes and submitting this form, SQL is executed that updates the changed fields and makes the changes in the database. After completion, admins are redirected to the posts page using the header() function in php. The new changes will be visible once redirected.

- ➢ By selecting 'remove', admins are redirected to a page that receives the post ID from the previous pages, and deletes that post from the database. The deletion page then redirects back to the posts page, where the post will now be removed. These redirects happen so quickly that from the admins perspective, it will look like the post is instantly removed after pressing the 'remove' button.
- ➢ The same functionality can be seen when clicking on a post and accessing the replies page. Each reply will have two buttons that dynamically appear under each reply (if you are an admin). These buttons function very similarly to the delete post buttons, with the edit post button bringing you to a new page, and the delete post almost instantaneously deleting a reply and redirecting back to the same page.

2.3 - Enabling/Disabling a user:

- ❖ Admins are able to enable and disable users. Upon reaching the edit user page for a selected user, an additional dropdown menu reading: 'active' will appear on the page. If the dropdown menu is changed to 'no', and the form is submitted, the selected user will no longer be able to access the website. This is done by accessing the database and changing the 'active' attribute for that user. If the user whose account was disabled tries to login, they will no longer be able to do so until an admin changes their 'active' attribute back to 'yes'.

3. General Timeline of Implementation:

Our github repository for this project can be found here:

- ❖ https://github.com/jasonschaad/cosc360Project

All information below can be confirmed by looking through our commits.

- ❏ Github created, uploaded readme, gave everyone full permissions.
- ❏ Database design: created a structure for our forum database and uploaded an initial SQL file.
- ❏ Working index page and login page created
- ❏ Slow addition of main functionality, such as addition of a posts page, resolving forgotten passwords, simple header creation
- ❏ Sql modified to include extra columns in users table, some column names changed
- ❏ Basic css styling added to all currently created pages, mobile functionality with hamburger button added.
- ❏ Functionality added such as a replies page, additions to posts page, and the ability to edit users

- ❏ Functionality added such as account creation, searching users from as admin perspective, adding posts/replies, hot posts
- ❏ Minor Bug fixes
- ❏ More filters added to admin page, collapsible posts/replies added
- ❏ Several bugs fixed, sitemap added
- ❏ Much more CSS added, including styling on tables, the header, the breadcrumb trail, forms etc..
- ❏ Removed security needed for search, added activity by date functionality
- ❏ More bug fixes
- ❏ Deployment on Schaad.ca/nerdForum
- ❏ Tweaks done to make nerdForum run smoothly on a server
- ❏ Walkthrough documentation and project summary completed
- ❏ Detailed site implementation document completed.