COSC 360 Documentation

Professor: Dr. Ifeoma Adaji

Ski-It: Ski discussion forum

Will Garbutt

Alrick Vincent

Walk through

Ski-it: a discussion site where you can post on topics related to skiing and snowboarding

home.php: Starting here, we are able to view hot posts, which are posts that are popular for the current day and popular posts which are most liked posts. From here we can select different kinds of posts on the side menu, either resort posts or backcountry posts. From the header we can search for posts by keywords and navigate to the login page.

searchpost.php: this page is reachable after pressing enter in the search bar in the header. This page will showcase all of the posts that match the keyword entered, this can be by user, topic, post title or post content.

login.php: Here we are able to login or go to the create account page where you can input user information to create an account, afterwards you will return to the login page. Here, sign in using username: 'will', password: 'will' for an admin account or username: 'test', password: 'test' for a user account. Let's start with the user account. After logging in successfully you will be directed back to the home.php page.

(User) Logged in home.php: Now that you are logged in as a user, the home page will reflect change by allowing you to comment and upvote or downvote on posts. Upvoting/downvoting will increment/decrement the amount of likes a post has. Pressing the comment button on a post will make a form pop up where you can write a comment and upon submitting the comment it will appear nested inside the post that was commented on. The home page will also reflect change by updating the header to have a logout button where you can log out and a 'My Account' button where you can go to your account portal.

account.php: here the side menu will update where you can navigate to 'your posts' which is the default, 'manage account', or 'new post'. From 'your posts' you can view all the posts you have made, and comments you have made on posts. From 'manage account' you can view all your account info and update it, this is where you can add your profile pic as well. When 'new post' is clicked a form is displayed where you can make a new post by inputting the title/subject of your post, selecting the type of post (backcountry or resort), uploading a photo if desired, as well as the content of the post. Clicking post will then upload your post to the site.

Now lets click 'log out' in the header which will bring us back to the home page, then let's click the same button which is now 'login' to go back to the login page and login as 'will' to walk through the admin features.

(admin) home.php: the home page will look and function the same as if a user is logged in, but instead of the header having a 'my account' button, it will have an 'admin' button to go to the admin portal. Let's click it.

admin.php: Here the side menu will update so you can navigate to ‘manage users’ (default), ‘manage posts’, or ‘User charts’. The ‘manage users’ page will display all users and their info by default but you can search for specific users as well using the search bar. From the displayed users you can select to enable/disable them. Here you can try disabling a user then try to log back in as the user you disabled and find that you will not be able to use the site. The ‘manage posts’ page will display all posts by default but you can also use the search bar here to search for posts by keywords. This page has the added functionality to allow an admin to delete posts. The ‘user charts’ page will display some charts for the admins to see. The ‘number of users registered over time’ chart which will show the amount of users registering on specific dates. The ‘posts distribution by post type’ chart will show the percentage of posts that are of type ‘resort’ and ‘backcountry’. And the ‘comment activity’ chart will display the comment activity done on the site by date.

Play around more with the site making different kinds of posts and adding comments to see how the charts change. The next page covers the detailed implementation of the site’s features.

Developer implementation details:

home.php
This is the page users get to when first getting on the website. This home page is styled with the header.css and home.css files, and are scripted with the votes.js files which contain the JavaScript functionality for handling user votes on the web sites. Almost every page on the forum includes a PHP file to include the header. The menu div is a navigation menu with links to the popular, resort, and backcountry pages of the web site. The main div contains two sections: Hot Posts and Popular Posts. The Hot Posts section displays the three most popular posts of the current day by upvotes. It determines today's date because everytime an user creates a post, it becomes time stamped with the current date and time. The Popular Posts section displays all posts that have at least 10 likes, sorted by likes in descending orders. The PHP code queries the database for posts with at least 10 likes and orders them by likes.

Limitations: The Hot Posts does not seem to work as intended, the code is there but does not display posts.

dbConnection.php

This file is included in almost every php file as it facilitates code reuse. This page contains the database connection details for both the localhost during development and the cosc360.ok.ubc.ca server. It creates a $pdo variable that is used in almost every php file.

backcountry.php & resort.php
Both files fetch relevant posts from the database, sort them by popularity, and display them along with user information, post images, and comments. From these pages, users can also add comments on the posts. The 'votes.js' is a JavaScript file which manages the upvotes and downvote buttons, the script updates the vote count in the database and refreshes the displayed vote count on the website. The way the file works is that it adds event listeners to all vote buttons on the page and updates dynamically the vote count when the user clicks it.

account.php & accountinfo.php & makePost.php & post.php
From the pages which contain the specific header, logged in users who are not admin are linked to my account page, if not they will be redirected to home.php. From this page, users can find a navigation menu in the same style as other pages of the website, users are able to reach their account information, their posts, and this is where the users are able to make a post. The account information displays the user's account. Their username is stored in the php session and is used to make an SQL query to retrieve the information. An html POST form then allows the user to change his preferred information, updating it through a SQL statement. When the user wants to make a new post, he is prompted with a form which requires the post subject, location type, the image, and write the post content. Post.php is the actual php file which handles the creation of the post in the database. It simply consists of a INSERT sql statement.

login.php & processLogin.php
The login.php file is accessed from the header when the PHP session username variable is not set. This file displays the login form, which allows the user to enter their username and password. It includes form

validation. Upon submission of the form, the processLogin.php file is executed. This file checks if the request method is POST and if both the username and password fields are set. It then starts a new PHP session and compares the given username and encrypted password to the database through a select statement. If the statement returns rows, there is a match. At that stage, there is a check to see if the user's state is 'disabled', which effectively means the user is banned. If it is, it sets an error message in the session and redirects the user to ban.php. If not, the username is stored in the session and the user is redirected to the home.php page where he will find a new header with the ability to click on the account page. If no match is found, the error message is set to 'invalid email or password' and the user is redirected to login.php.

logout.php
This file is responsible for logging out a user from the website. It does this by starting a new session, unsetting all session variables, and then destroying the session. After destroying the session, the user is redirected to the home.php file where he will find a different header.

create.php & processcreate.php & create.js
This file contains the HTML markup, styling and Javascript necessary to render the "Create Account" page. It includes a form for users to input their first name, last name, username, email, and password. The form's action is set to processcreate.php, which processes the user registration form data. It first checks if the request method is POST and if all required fields are present. If the username and email are unique, the script inserts a new user into the database and redirects the user to the login.php page. If the username or email already exists, an error message is set in the session, and the user is redirected back to the create.php page. In addition, the create.js file is responsible for handling client-side form validation during the user registration process. The window.onload event listener is set to call the init() function when the page has finished loading. The script iterates through all required fields and checks if any are empty. If any empty fields are found, the empty variable is set to true. If the empty variable is true, the form submission is prevented, and an error message is displayed, the empty fields are given a red error styling using the 'error' class. If the empty variable is false, the script checks if the password and confirm password fields match. If they do not match, the form submission is prevented, and an error message is displayed and the fields are given a red error styling using the 'error' class. This allows the user registration process to be made more user-friendly and efficient by providing immediate feedback on input errors before submitting the form.

admin.php & admin.js & changestate.php & commentData.php & deletepost.php & registData.php & typeData.php
The admin.php file serves as the main interface for the admin dashboard. It contains sections for managing users, managing posts, and displaying user charts. Each section can be shown or hidden by clicking on the corresponding menu item. This is handled by the admin.js's showSection(id) function, which takes an element id as its argument. This sets the display style of all elements with the class name "section" to "none". Then it sets the display style of the section with the specified id to "block". The page also actively stores the active section, stored in localStorage to remember which page the admin was last on. By default, the admin is lead to the manage users section, which allows an admin to search for users by username, email or more and displays the result in a table. In that table, the state of the user is shown and can be toggled by a button. This button is handled by changstate.php which updates the state of the user dynamically and leads it back immediately to admin.php, showing a dynamic response for the user. In the manage posts section, admins are able to look for posts through a search bar and are able to delete them through deletePost.php,

which prepares an SQL query to delete the post. In the user charts section, the google charts API is used to draw the following charts. The function drawRegistration() fetches the results of an SQL query in a json format from registData.php, returning the number of user registrations over time in a line chart. The drawPostType() fetches json data from typeData.php, returning the distribution of posts by type in a pie chart. The drawComment() fetches data from commentData.php, returning the number of comments per post type in a column chart. All these functions are called in drawCharts() to display them to the admin. The admins are also able to go to their own account page in account.php

searchpost.php:
This file is responsible for displaying the search results when a user searches for a post using the header search bar. Error prevention is first performed, checking if the request method is POSt and if the 'search' input field is set. A SQL query is prepared to select all posts where the username, title, or content contains the search keyword using the LIKE operator for partial matches. If the search input is not set, the script fetches all posts from the database. If there are any results, the script iterates through the fetched posts and displays them in a similar fashion to the home.php file.

adminTest.php
This file is a PHPUnit test file testing the isAdmin() function defined in the shortcuts.php file. The setUp() method is implemented, assigning the $pdo property to a new PDO connection to the skiit database. The file basically tests if the isAdmin() function correctly identifies an admin user. It sets a known admin username, calls the isAdmin() function, and asserts that the returned value is true. A similar test is conducted to check if the isAdmin() function correctly identifies a non-admin user, setting a known non-admin username and checking if the function returns false for a non-admin. Finally, a test is conducted to check that the function returns false for a non-existent user.