Function: browse.php

browse.php is designed to allow users to browse through displayed media. It displays the user’s username and account number, as well as links to their profile page, the page to update their profile, and the page that holds their created playlists.

Below these are the options to upload a file, the ability to keyword-search media files, the ability to category-search media files, and the ability to sort media files by views or upload date.

Media is displayed at the bottom of the page. Display includes media ID, the title with a link to the view page, the upload date and uploader, the view count, a download link, a link to add to favorites, and a link to add to playlists.

Implementation: browse.php

browse.php will redirect any users who are not signed in to the login page.

browse.php uses $\_POST with HTML forms to process keyword searches, media categorization, and media sorting. For this reason, it is only possible to index media on one of these three options at once.

browse.php uses the $\_SESSION variable to get the user’s account\_id and username when displaying these at the top of the page. The session id is also used to generate the link to the user’s profile page.

browse.php uses a hyperlink to link to a page for uploading media.

browse.php selects media\_id, name, path, type, upload\_time, and views from the table media, account\_id and username from account, and media\_blocked from interaction. media\_blocked will prevent media from rendering for any user who has been media blocked by the uploader.

browse.php will check to see if any media sorting has been used with the isset($\_POST[]) function. If a specific $\_POST variable is set, it means the user wants the media organized a certain way. Depending on which variable was set, a statement is appended to the SQL query to organize the data on a trait, such as upload\_time or views.

browse.php uses html forms to present the search bar and the media categorizations. All of the processing is done in the back end by PHP.

browse.php uses a while loop to print every row returned by the SQL query. If the uploader of the media on that row did not media block the current viewer, the media is printed out in a row. This process repeats until mysql\_fetch\_row() cannot fetch any more data.

Function: config.php

config.php holds the configuration data for our database. It is not rendered to the user.

Implementation: config.php

config.php simply stores a few values into variables, such as the database host, user, password, and database name.

Function: create\_table.php

create\_table.php is used to create the database tables. It contains the structure of all of the tables we used for this project, in the form of sql queries.

Implementation: create\_table.php

create\_table.php simply holds some commands to create sql tables. It is not meant to be rendered to the user, and exists only to be run once if tables need to be initialized.

Function: favourites\_process.php

this function is deprecated. It is not used.

Implementation: favourites\_process.php

this function is deprecated. It is not used.

Function: favourites.php

this function is deprecated. It is not used.

Implementation: favourites.php

this function is deprecated. It is not used.

Function: function.php

this page contains several functions used throughout the code.

* user\_exist\_check($username, $password, $email) queries a database. If the user in question is not found, it inserts into the table, creating an account with that username, password, and email. Otherwise, it returns an error, usually because the username was already taken.
* user\_pass\_check($username, $password) queries a database. It looks for a user with the specified username. If it finds one, it compares the password for that username with the one passed to the function. If they do not match, or if the username does not exist, they return error values.
* updateMediaTime($mediaid) – deprecated. This function is not used.
* upload\_error($result) takes in an integer as an upload error code, and prints out the corresponding error string.
* get\_account\_id\_from\_username($username) queries a database. It fetches the account id associated with that username and returns it as a string. This returns an empty string if the username is not found.

Implementation: function.php

* user\_exist\_check($username, $password, $email) first queries the database to retrieve the account information of a user with the specified username. If this returns a nonempty result, it means the user exists already; an error is thrown. If this returns an empty result, a new query is created. This creates an account by passing in the username, password, and email, as well as account type (1 for normal user, this is default. Admins are type 0.)
* user\_pass\_check($username, $password) first queries the database to retrieve the account information of a user with the specified username. If the result is empty, it means that user does not exist, and an error value is returned. If the user DOES exist, their password is compared to the password passed in. If they do not match, a different error value is returned. If they do match, the function returns a success value.
* updateMediaTime($mediaid) – deprecated. This function is not used.
* upload\_error($result) uses a SWITCH CASE statement that switches the media error on the passed-in error value. This returns a string with the error in it.
* get\_account\_id\_from\_username($username) selects information from the account of a user with username equal to the passed-in username. I the user does not exist, the function passes back an empty string. Otherwise, the function fetches the data with mysql\_fetch\_row and returns the account id.