Project Report

(Group 14)

Derrick Park (1278970)
Youdong Ma (1382734)

System Architecture

In this project, three tier architecture is used - Model, Controller and View. Smarty template engine is used for implementing View Architecture. A separate folder “display” is created to manage the smarty files. All the templates files are located in “templates” under “display” folder. The extension of Smarty files is “.tpl”. The Smarty core code is placed in different folder “plugins”. In this folder, all the files need by Smarty to decode the template files are placed. It is very easy to setup Smarty in PHP, just include Smarty.class.php located in plugins folder and set the path of templates and cache. The Smarty core files translates tpl files to PHP files and process it. Smarty uses “{“ as left delimiter and “}” as right delimiter to guide Smarty that its syntax has started and needs to be parsed now. As, “{ }” are common usage brackets in Javascript and CSS which can be written in templates so it was changed to “~” as left delimiter and “`” as right delimiter because they are hardly used anywhere.

All the PHP files in web root directory works as model which reads data from HTML and sends data back to HTML for display. A file called “databaseQuery.php” works as controller which interacts with the database and sends the data to Model. There is no other file other than it, which interacts with the database. The database used is Mysql and PDO is used to interact between PHP and Mysql because it is more secure than normal mysql connection.

Configuration

A config.php file is created which is included in every Model PHP File. In config file, the login details for mysql is mentioned, Smarty file is included and all configuration parameters of Smarty is mentioned. In case user is logged in, cookies are refreshed. The controller class “databaseQuery.php” is included and its object is created so that it can be used by all the Model architecture files for contacting database indirectly through Controller. The constructor of Controller class creates an object of PDO which can be used by the functions of controller class only.

User Management Module

This module helps user to register and login into the system. A single PHP file is used to show both register and login form to the user. The PHP file which shows this module is “index.php” and template file is “index.tpl”.

Register

In case, user wants to register, the form is filled and submitted. After submission of the form, it is validated used Jquery and shows errors in case any field is blank and do not meet the requirements like contact number cannot have alphabets. After successful submission of the form, user goes to “register.php” where unique username is checked. If username is not unique, user is redirected to the register page with data pre-filled and an error message stating choose an unique username. In case, username chosen is unique, the data is saved into the database. After successful insertion of data in database, COOKIES is set for 1 hours and redirected to home page.

Query for Checking Unique user name
PHP Code - $databaseQuery->isUsernameUnique($username)
Mysql Code - SELECT * FROM ".DB_NAME.".users WHERE user_name = :username;

Query for Saving data in table users
PHP Code - $databaseQuery->insertDataUsers($username,$password);
Mysql Code - INSERT INTO ".DB_NAME.".users (user_name, password, date_registered) VALUES (:username, :password, now());

Query for Saving data in table persons
PHP Code - $databaseQuery->insertDataPersons ($username, $fname, $lname, $address, $email, $contact)
Mysql Code - INSERT INTO ".DB_NAME.".persons (user_name, first_name, last_name,address,email,phone) VALUES (:username, :fname, :lname, :address,:email, :contact);

Login

In case, user wants to login, the form is filled and submitted. After submission of the form, it is validated used Jquery and shows errors in case any field is blank and do not meet the requirements. After successful submission of the form, user goes to “executeLogin.php” where username and password is checked. If username and password both matches, user is logged into the system by setting cookies and redirecting to home.php.

Query for Checking username and password in table users
PHP Code - $databaseQuery->verifyUser($username,$password)
Mysql Code - SELECT user_name FROM ".DB_NAME.".users WHERE user_name = :username and password = :password

Uploading Module

The file upload module is developed using HTML5 file upload functionalities. The file uploaded is validated for extension and size using javascript. In case of an error, alert pop up comes and asks user to upload valid file. According to the requirement, only gif and jpg images are allowed. According to the database design given, the maximum file size which can be stored in database is 64Kb. Only browsers with HTML5 file upload support can use this website to upload the files. In case of folders, only Google Chrome gives the support and no other browser gives the support for uploading folder. Therefore, uploading module will work smooth in latest Google Chrome version. The uploaded files thumbnails are also shown using javascript.

The file name “upload.php” will show upload template with two options - Single file upload and folder upload which will upload all the valid images in that report. When a file or folder is uploaded, HTML5 file uploader is initiated and uploadFile.php is called using XHR Request. The file is sent to uploadFile.php, a random file name is generated and the file is stored by that name in upload folder. After successful upload, the random generated filename is returned to HTML.

After successfully uploaded the images, user can optionally fill up the details like subject, place, timing and description. After submitting the form, it goes to executeUploadFile.php, where data is stored into the database and user is redirected to home page with success message.

In executeUploadFile.php, the data of image is read by the file stored in uploads folder. GD library is used for creating thumbnail from main image. After getting thumbnail image and original image data, all the data is sent to database for insertion.

Query for inserting image details

PHP Code - $databaseQuery->insertPhotoDetails($photoId, $username, $permission, $subject, $place, $date, $description, $thumbnailData, $photoData)

Mysql Code - INSERT INTO ".DB_NAME.".images (photo_id, owner_name, permitted, subject, place, timing, description, thumbnail, photo) VALUES (:photoId, :username, :permission, :subject, :place, :time, :description, :thumbnail, :photo);

Security Module

In this module, only registered user can create a group and upload images. The owner of the group can only edit or delete the groups created. The owner of the images uploaded can only edit and delete it. Only users which are allowed to view the images can view them. In case user wants to create a group, createGroup.php is called. In this file, it is checked if user is logged in or not.

If user is logged in, create group page is shown otherwise user is redirected on login page. This has been implemented all the pages which can be accessed by only logged in users.

After the data on create group page is submitted, all the data is sent to executeCreateGroup.php file. In this file, unique group name corresponding to the username is checked. If it is unique, the data is saved into the database otherwise, user is shown an error with message indicating that group name is non unique. If the group name is unique for that corresponding username, data is saved into the database and group members are added into different database.

Query for checking Unique Group Name
PHP Code - $databaseQuery->isGroupNameUnique($username, $groupname)
Mysql Code - SELECT * FROM ".DB_NAME.".groups WHERE user_name = :username AND group_name = :groupname

Query for inserting group data in groups table
PHP Code - $databaseQuery->createGroup($username,$groupname, $groupId)
Mysql Code - INSERT INTO ".DB_NAME.".groups (group_id, user_name, group_name, date_created) VALUES (:groupId, :username, :groupname, now())

Query for inserting group members in groups_list table
PHP Code - $databaseQuery->insertGroupUserMapping($groupId, $user_name)
Mysql Code - INSERT INTO ".DB_NAME.".group_lists (group_id, friend_id, date_added) VALUES (:groupId, :username, now())

In case, owner of the group wants to delete the group, deleteGroup.php file is called in which it is checked whether the user is logged in or not. If the user is logged in, it is checked whether the group requested to be deleted is owned by the logged in user. If yes, group details are delete from groups and group_lists table.

Query for checking whether group is owned by the logged in user
PHP Code - $databaseQuery->getGroupName($groupId, $username)
Mysql Code - SELECT * FROM ".DB_NAME.".groups WHERE user_name = :username and group_id = :groupId

Query for deleting group details from groups table
PHP Code - $databaseQuery->deleteGroup($groupId)
Mysql Code - DELETE FROM ".DB_NAME.".groups WHERE group_id = :groupId

Query for deleting group details from group_lists table
PHP Code - $databaseQuery->deleteGroupMembers($groupId)

Mysql code - DELETE FROM ".DB_NAME.".group_lists WHERE group_id = :groupId

In case of group edit, file name - editGroup.php is called. Group Id is passed to this file. In this file, group name corresponding to the group Id and username is extracted from database. The username is used to make the system more secure. In case any user just change the group Id in the url so he/she wont be able to edit the group details of any other user. After receiving the group name corresponding to both group id and username, details of group members are extracted. All the details are passed to smarty template where data is shown with group members pre selected and group name pre-filled.

Query for checking group name
PHP Code - $databaseQuery->getGroupName($groupId, $username)
Mysql Code - SELECT * FROM ".DB_NAME.".groups WHERE user_name = :username and group_id = :groupId

Query for retrieving group members details
PHP Code - $databaseQuery->getGroupMembers($groupId)
Mysql Code - SELECT * FROM ".DB_NAME.".group_lists WHERE group_id = :groupId

After the group details are edited and form is submitted. The data is posted to executeEditGroup.php where unique group name is checked and details are updated if group name is unique. In this file, we delete all the group members of the that group and then insert the group members chosen now.

Query for Checking unique group name
PHP Code - $databaseQuery->isGroupNameUnique($username, $groupname)
Mysql Code - SELECT * FROM ".DB_NAME.".groups WHERE user_name = :username AND group_name = :groupname

Update the details in groups table
PHP Code - $databaseQuery->updateGroup($groupname, $groupId)
Mysql Code - UPDATE ".DB_NAME.".groups SET group_name = :groupName WHERE group_id = :groupId

Delete all group members in the group
PHP Code - $databaseQuery->deleteGroupMembers($groupId)
Mysql Code - DELETE FROM ".DB_NAME.".group_lists WHERE group_id = :groupId

Insert Group members
PHP Code - $databaseQuery->insertGroupUserMapping($groupId, $user_name)
Mysql Code - INSERT INTO ".DB_NAME.".group_lists (group_id, friend_id, date_added) VALUES (:groupId, :username, now())

After the images has been uploaded in uploading module, images can be edited or deleted also. In case of image delete, deletePhoto.php is called and passed the photo id. The image is deleted by matching photo id and logged in user so that only owner can delete the image.The option to delete image is only shown to the owner of the image by comparing the username of logged in user to the owner name of the image.

Query for deleting image
PHP Code - $databaseQuery->deletePhoto($photoId, $username)
Mysql Code - DELETE FROM ".DB_NAME.".images WHERE photo_id = :photoId AND owner_name = :username

In case of edit image, editImage.php file is called where the data corresponding to the photo id is extracted and passed to the template editImage.tpl to show all the details. In editImage.tpl, the data is shown pre-filled and when the form is submitted, all the data is posted to updateImage.php where all the data is updated.

Query for updating Image
PHP Code - $databaseQuery->updatePhotoDetails($photoId, $username, $permission, $subject, $place, $date, $description)
Mysql Code - UPDATE ".DB_NAME.".images SET permitted=:permission, subject=:subject, place=:place, timing=:time, description=:description WHERE photo_id=:photoId AND owner_name = :username

Display Module

In display module, the thumbnail of the images are shown, the complete details of images are shown with original image and top 5 most viewed images are shown. On homepage of the user logged in system, thumbnail of the uploaded image by user is shown and top 5 most viewed images are also shown.

For displaying thumbnail image, photo id along with parameter - thumb is passed to file showImage.php. In this file, the data of image is extracted using photo Id and according to parameter thumb, the thumbnail image data is shown using php headers. Similarly, in case of complete image details, viewImage.php is called which extracts all the data corresponding to the photo id and passes to viewImage.tpl. In this case, photo Id with parameter - full is passed to the file showImage.php to show the original image uploaded. The username of logged in user is checked with the owner of the image, if both are equal, option to delete and edit image is shown. If the permitted value of the image is not 1 and 2, the name of the group is extracted.

Query to retrieve photo data
PHP Code - $arrPhotoData = $databaseQuery->getImage($photoId)
Mysql Code - SELECT thumbnail,photo FROM ".DB_NAME.".images where photo_id = :photoId

Query to retrieve all photo details
PHP Code - $databaseQuery->getPhotoDetailsById($photoId)
Mysql Code - SELECT * FROM ".DB_NAME.".images where photo_id = :photoId

Whenever, viewImage.php is opened, it is checked if the user has seen the image before or not. If the user has not seen the image before, a new entry into the table unique_views is made. In this table, photo id and username is stored which tell that the user has seen the image with the corresponding photo Id.

Query to check if user has seen the image before
PHP Code - $databaseQuery->isImageAlreadySeen($username, $photoId)
Mysql Code - SELECT * FROM ".DB_NAME.".unique_views where photo_id = :photoId AND user_name = :username

Query to insert unique views
PHP Code - $databaseQuery->insertUniqueView($username, $photoId)
Mysql Code - INSERT INTO ".DB_NAME.".unique_views (photo_id, user_name) VALUES (:photoId , :username)

On homepage, top 5 most viewed images are extracted from table and their thumbnail image is shown.

Query for extracting 5 top most viewed images
PHP Code - $databaseQuery->getTopPhoto()
Mysql Code - SELECT photo_id, count(photo_id) as views FROM ".DB_NAME.".unique_views group by photo_id order by views limit 5

Search Module

In search module, the user can search the image by keyword(s) and the range of time. They can sort the image according to the time. If no option to sort the image is chosen, images are shown according to the algorithm - Rank = 6*subject + 3*place + Description

When search option is clicked, search.php is called which show all the options using search.tpl. In this file, keywords is mandatory. When details are filled and form is submitted. All the data is posted to searchResult.php. In this file, all the groups in which user is a member is calculated so that only those images can be shown which the user has right to view. After this data is extracted from database according to the parameters selected. If no sorting is selected, frequency of the keywords is find out for each result and assigned to an array. After that, array is sorted according to the ranking and result is displayed.

Query for extracting data

PHP Code - $databaseQuery->getSearchResult($strGroupIds, $keywords, $condition, $from, $to, $username, $sorting)

Mysql code - $where = "(subject like '%".$keywords."%' OR place like '%".$keywords."%' OR description like '%".$keywords."%')";

```
                    if ($condition == 'or') {
                            $where = $where. " OR (timing >= '".$from."' AND timing <=
'".$to."')";
                    }elseif($condition == 'and') {
                            $where = $where. " AND (timing >= '".$from."' AND timing <=
'".$to."')";
                    }
                    $where = $where. " AND (permitted IN (".$strGroupIds.") OR
owner_name = '".$username."')";
                    if($sorting!='') {
                            $where = $where. " ORDER BY timing $sorting";
                    }
                    $sql = "SELECT photo_id, owner_name, permitted, subject, place, timing,
description FROM ".DB_NAME.".images WHERE $where";
```