CMPUT 391 Image Sharing Project

BY: PREYANSHU KUMAR, JUSTIN WONG, GURMEET DHILLON

INTRODUCTION

Our Online Image Sharing System allows users to upload *jpeg*, *gif* files, share them with their friends, and browse for images. Users can also create groups that help to filter the images that can be seen. Our system is available at consort.cs.ualberta.ca and the databases are hosted on the University's Oracle servers. We use a combination of HTML, CSS, JavaScript, PHP and SQL queries in order to implement our system.

FILES

INITIAL DATABASE SETUP

- sql.sql: This file is the sql file used to create the initial database. Essentially, it contains all the sql commands present in setup_w2016.sql as per the project specifications and additionally includes other sql commands required for the setup of our database.
  - **SQL Statements used**:
    - [INSERT INTO users values('admin','admin',sysdate);]
    - [CREATE INDEX placeindex on images(place) INDEXTYPE IS CTXSYS.CONTEXT;]
    - [CREATE INDEX subjectindex on images(subject) INDEXTYPE IS CTXSYS.CONTEXT;]
    - [CREATE INDEX descriptionindex on images(description) INDEXTYPE IS CTXSYS.CONTEXT;]

USER MANAGEMENT MODULE

- login_submit_final.php: The login page of our website. Here, a person may click register to create a new user account, or login with an already created user account.
- landing_page.php: This file is the website home page. From here, a user may view a gallery of all uploaded photos, their own uploaded images, upload a saved image from their computer, manage their groups, access the help page (User Documentation), search for images based off of keywords and dates, and logout from their account and be returned to the login page. If the user is 'admin', the user may also access administrator options that include OLAP operations.

- logout.php: Logs the user out of the website and clears the session username and session password. Then returns you to login_submit_final.php.

SECURITY MODULE

- PHPconnectionDB_final.php: This file is an intermediary file between login_submit_final.php and landing_page.php that acts as a security measure which checks if a valid username and password are provided. If a valid username and password are provided, a user is logged in. If a valid username and password are not provided, the user is redirected back to the login page (login_submit_final.php).
  - **SQL Statement Used**:
    - [SELECT user_name, password FROM users WHERE user_name = $username AND password = $password;]
- group_homepage.php: This file is the hub for group management, where you can create groups and change members of each group.
- group_signup.php: This file lets the user enter the Name of the group that they would like to create.
- group_registration_confirmation.php: This file makes sure the group name is available and puts it into the database if it is unique while also generating the group creation date and id.
  - **SQL Statements Used:**
    - [SELECT * FROM groups WHERE group_name =:group_name;]
    - [INSERT INTO groups VALUES (:group_id, :leader_name, :group_name, :datereg;]
- groupManipulation.php: This file shows you all the groups you are a part of and all the users that have been added to each one. From this page you can click on one of two buttons to add/remove users from your groups.
  - **SQL Statement Used:**
    - [SELECT * FROM group_lists, groups WHERE groups.user_name =:username AND groups.group_id = group_lists.group_id;]
- addUser.php: This file lets the user enter in who they want to add to the group and which group to add them to. The user being added has to have an account before hand.
- add_person_confirmation.php: This adds the person specified to the database. It also creates a value from when they were added and gives them a notice.
  - **SQL Statement Used:**
    - [INSERT INTO group_lists VALUES (:group_id, :friend, :date_added, :notice;]
- removeUser.php: This file lets the user enter in who they want to remove and from which group they want to remove them from. The person must be in a group beforehand.

- remove_person_confirmation.php: This file removes the specified user from the database.
    - **SQL Statement Used:**
        - [DELETE FROM group_lists WHERE group_id =:group_id AND friend_id =:friend;]
- updatePhoto.php: This file lets the user change the subject, place, description and the permitted groups that can access it.
- updatePhotoConfirmation.php: This file updates the changed information about the photo and uploads these changes to the database.
    - **SQL Statement Used:**
        - [UPDATE images SET permitted=:permission, subject=:subject, place=:place, description=:description WHERE photo_id=:photo_id AND owner_name=:username;]

UPLOADING MODULE

- getPhotoInfo.php: This file is used to input information for a photo to upload. Here, a user may choose the photo's permissions, give the photo a subject value, give the photo a place value, select a date to associate with the photo, enter some descriptions about the photo, and select a photo(s) from their computer to upload. Photos must be in either *jpeg* or *gif* image formats.
- upload.php: This file is the backend file for uploading an image. Here, the information given by the user in getPhotoInfo.php is inserted into the Oracle database.
    - **SQL Statementes Used**:
        - [INSERT INTO images VALUES($photo_id, $owner_name, $permission, $subject, $place, $timing, $description, $thumbnail, $photo);]

DISPLAY MODULE

- image_gallery.php: This file is used to display the thumbnails for all images that a logged in user has uploaded. By clicking on a thumbnail, the user will be redirected to a page where they can view the uploaded image.
    - [SELECT * FROM images where owner_name = :owner_name;]
- all_images_basic.php: This file display's all of the thumbnails for the images of all users, this should theoretically be ranked in such a way that the page would show based off of popularity and it would also show the images based off of preferences. This could not be done but we got pretty close to finishing it.
    - SQL Statements Used:
        - SELECT * FROM images
- getImage.php: This file is used when the user clicks a thumbnail. Essentially, this file will redirect the user to a page where they can view their uploaded image.
    - **SQL Statementes Used**:

        - [SELECT photo FROM images where photo_id = $photo_id;]

- getImageThumb.php: This file is used to display the thumbnails in image_gallery.php.
    - **SQL Statementes Used**:
        - [SELECT thumbnail FROM images where photo_id = $photo_id;]

SEARCH MODULE

- search.php: This file is used by the user to give them options on which to search for their photos. They are given a start and an end date to choose from, and they are allowed to use keywords that will be sorted through and shown based off of frequency unless they choose the most recent or the least recent option on the radio buttons. Then, based off of what they select, they will be given those types of photos. This is basically the “front end” of our search module as it is everything that the user enters to be able to get what they want. This mostly is just a bunch of html and bootstrap that ends up being used to be sent to search_for_images.php:

- search_for_images.php: This is our “back-end” and the “display” for our search module. This does all of the searching for us, it basically gets all of the photo.id’s that match the search conditions based off of what the users options has entered. An SQL statement is started and then certain statements get added on at the end based off of what the user has entered for (their dates, keywords and most recent or least recent) This also makes sure to check that the search follows the security module based off of whether the photo is public, private, or if the person is in a group and then adds that to the SQL statement.
    - **SQL Statementes Used**:
        - This entire statement is very large, and changes depending on what the user wants exactly. Therefore, we will be going through each part individually:
        - To start we have to rebuild the indexes up so that we are able to grab from them so we have the statements:
            - $rebuilds = "alter index subjectindex rebuild ";
            - $rebuildp = "alter index placeindex rebuild ";
            - $rebuildd = "alter index descriptionindex rebuild ";
        - Initially for searching we will just have a select from as that’s what we are going to get from, we then add a variable called $sqlsearchcheck which will then take what the needs as search criteria:
            - SELECT i.photo_id FROM images i WHERE '.$sqlsearchcheck;
        - 1. if the user has a keyword (or multiple keywords) only that they want to search from it will search the indexes based off of what word that we search for

                - '$sqlsearchcheck .= ‘ (contains(i.place,\''.$keywords.'\', 1)>0 or contains(i.subject,\''.$keywords.'\', 2)>0 or contains(i.description,\''.$keywords.'\', 3)>0)'
            - 2. If the user only enters a start and end date (both of which are needed for this statement to work):
                - $sqlsearchcheck .= ' i.timing between to_date(\''.$start_date.'\', \'mm/dd/yyyy\') and to_date(\''.$end_date.'\', \'mm/dd/yyyy\')'
            - 3. If the user has both keyword (or multiple keywords) as well as a unique start and end date :
                - '$sqlsearchcheck .= (contains(i.place,\''.$keywords.'\', 1)>0 or contains(i.subject,\''.$keywords.'\', 2)>0 or contains(i.description,\''.$keywords.'\', 3)>0) AND i.timing between to_date(\''.$start_date.'\', \'mm/dd/yyyy\') and to_date(\''.$end_date.'\', \'mm/dd/yyyy\')'
            - After this we have to take into account permissions in every case. That is, if the picture is public, or if the picture is private and the user is the one who is seeing the image is the one who is logged in, and finally if the user viewing the image is part of a group, or the leader of the group.
                - $sqlsearchcheck .= ' and ((i.permitted = 1) or (i.permitted = 2 and i.owner_name = \''.$user.'\') or (i.permitted <> 1 and i.permitted <> 2 and i.permitted in (select group_id from group_lists where friend_id = \''.$user.'\') or i.permitted in (select group_id from groups where user_name=\''.$user.'\')))';
            - After doing this, all that is left is how the user wishes to order their files, there are 3 different ways that a person wants to order their pictures. 1. The default case, which is newest first.
                - $sqlsearchcheck .= " order by i.timing asc";
            - 2. The next case will be if they want it to be sorted by oldest first
                - $sqlsearchcheck .= " order by i.timing desc";
            - 3. The last case is if the user has chosen the none, we will then use the given formula in the assignment page of [Rank(photo_id) = 6*frequency(subject) + 3*frequency(place) + frequency(description)]:
                - $sqlsearchcheck .= " order by(rank() over (order by(6*score(2)+3*score(1)+score(3)))) desc";
        - All of the things above, depending on what the user wants when they are searching will be added to $sqlsearch check, and then it will be executed.

DATA ANALYSIS MODULE

- admin_data_analysis.html: This file is used by the ‘admin’ user to do OLAP operations. What information is outputted is dependent on what parameters are selected. If the user parameter is selected, then the OLAP operation will display each username and how many images they have uploaded. If the subject parameter is selected, then the OLAP operation will display subjects and how many images are associated with that subject. If a time parameter is selected, the OLAP operation will display the time in accordance to the selected time parameter and how many images are associated with each time. If no parameters are selected, the OLAP operation will simply return the total number of images in the Oracle database.

- admin_output.php: This file makes up the backend part of the Data Analysis Module and outputs the specifics of the OLAP operation depending on what parameters the admin had selected in admin_data_analysis.html.
    - SQL Statementes Used: [SELECT owner_name, subject, timing, COUNT(photo_id) FROM images GROUP BY owner_name, subject, timing;] – This query is changed depending on the parameters the admin had selected in admin_data_analysis.html.