INFS3202/7202 Practical 9

You must present this practical to your lab tutor during your scheduled lab sessions in week 12 that starts 21/05/2012. The prac could be done either in the lab, or at home.

You may use either PHP or Java for this prac but it is recommended that you continue with the with the same server-side language that you used in Prac 8.

This task will involve refactoring much of your existing code such that it conforms to an object-oriented design. **A skeleton for PHP will be provided separately later.**

Task 1 - Design Patterns (2 Marks)

In this course we have covered a wide range of topics. Modify the code handling SQL operations to use a fully object-oriented approach that creates an API to be used on the system. All SQL code should be contained within the class and not appear in on the page itself though the code will need to be included (PHP) or imported (Java) so as to use these methods.

Task 1.1 (1 Mark)

Implement the following objects. You may implement further functions if desired. You should follow the capitalisation and naming conventions of the language that you use.

<u>Images</u>

get_image(image_id) - Returns image information as in get_images for the specified id (i.e., *image_id*) but as a single dimensional Array(PHP) or ArrayList(Java)

get_image_with_tag(tag_text) - Returns a multi-dimensional Array(PHP) or ArrayList(Java) containing the image id, image name and path (as a single string), and timestamp of all images with the tag "*tag_text".* Note that, *tag_text* is a single tag.

get_image_tags(image_id) - Return an Array(PHP) or ArrayList(Java) containing all of the tags associated with the image whose id is "*image_id*".

<u>Albums</u>

get_album(album_id) - Returns an Array(PHP) or ArrayList(Java) of integers containing each image_id in the specified album.

get_album(tag_text) - Returns an Array(PHP) or ArrayList(Java) containing ids of all albums which have matched tag "*tag_text*". Note that, *tag_text* is a single tag.

You should be able to import the classes implemented in the Task 1.1 into any PHP page in your project. Use the methods defined in Task 1.1 to implement the following tasks.

Task 1.2 (1 Mark)

Create a new page of “Image Search - Search by tags” (Example 1). Given a tag, display all images associated with the tag. The complete tag set of each image should be shown under the image. Use the methods **get_image_with_tag(tag_text), get_image(image_id)** and **get_image_tags(image_id)** in this task**.**

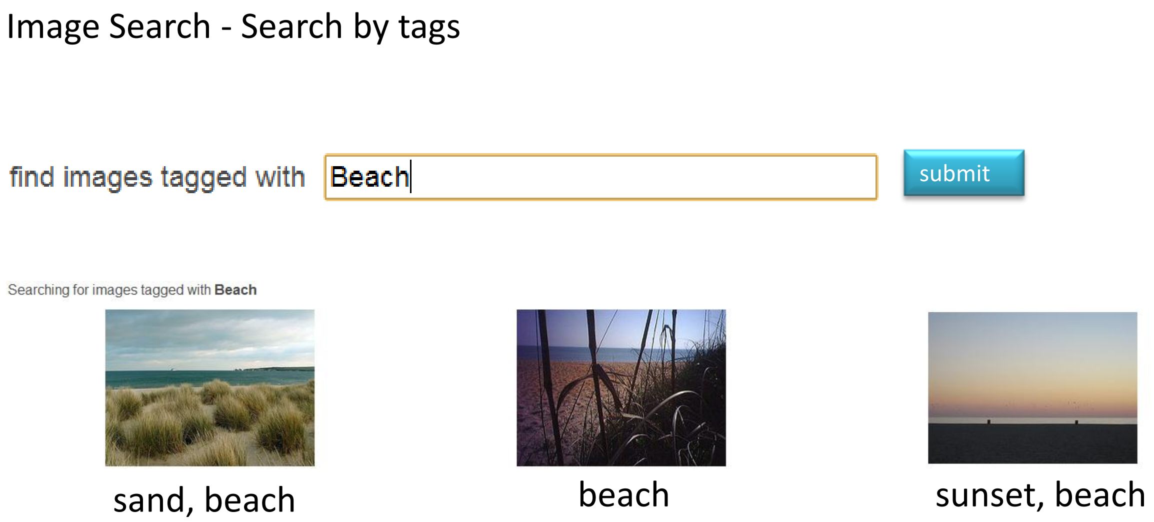


Image Search - Search by tags

find images tagged with Beach

Searching for images tagged with **Beach**

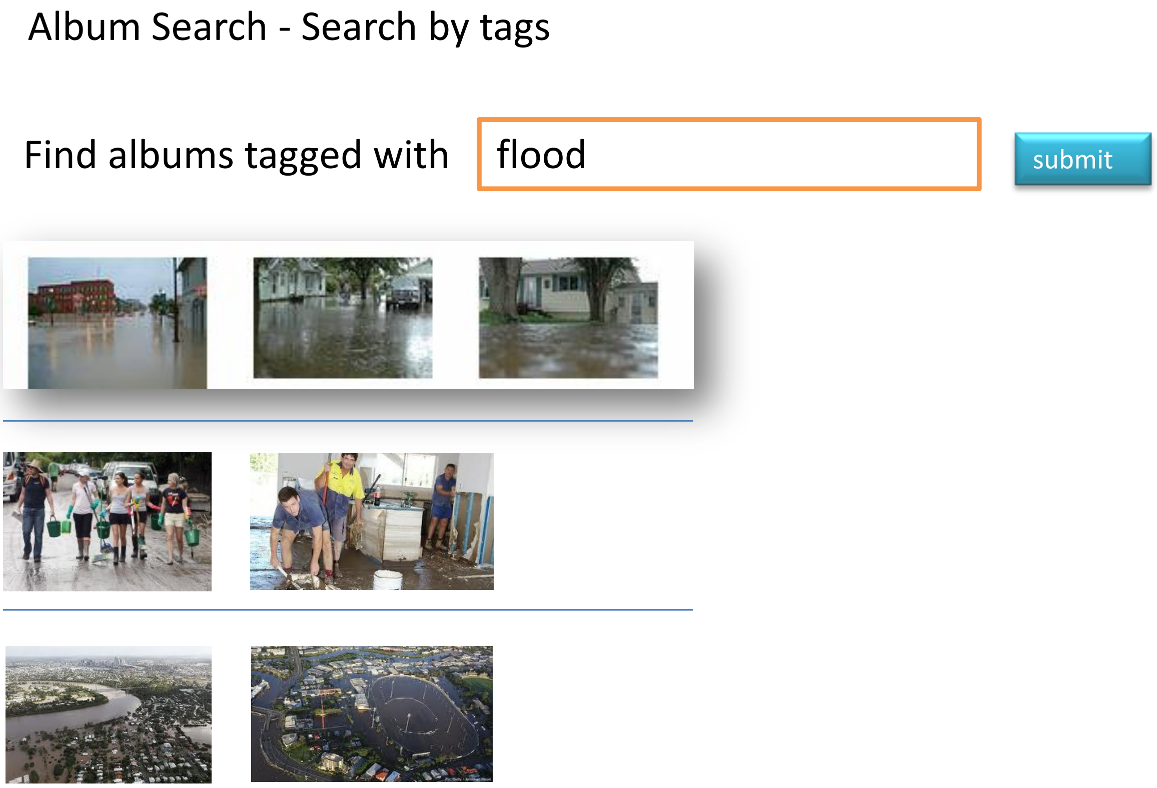
sand, beach

beach

sunset, beach

Example 1: Image Search – Search by tags

Create a new page for “Album Search - Search by tags”. Use **get_album(tag_text)** and **get_album(album_id)** and **get_image(image_id)** in this task.



Album Search - Search by tags

Find albums tagged with flood

Example 2: Album Search – Search by tags

Task 2 - Add Support for Multiple Users in MySQL (2 Marks - 1 Mark for the Database Implementation and 1 Mark for the Class Implementation)

Next you will need to create a table in your database called “users”. It should have the following fields. Populate the database with a few username and password combinations.

Field Name	Type	Notes
ID	INT, Primary Key	Auto Increment
Username	VARCHAR(50)	Stores usernames
Password	VARCHAR(50)	Stores passwords

Now modify your login code so that it allows any username/password combination present in the database. You should create a further class for this set of functions called users. Your login page should continue to use Ajax and JSON.

Users

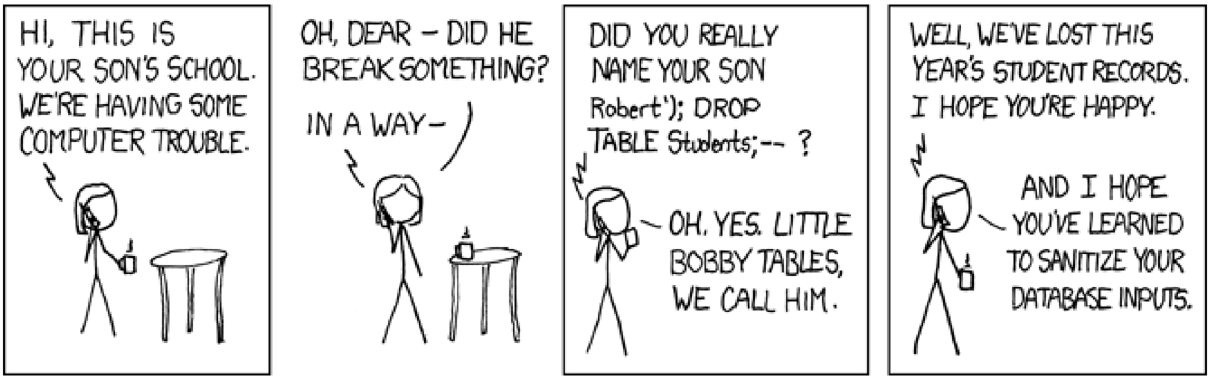
create_user(username, password) - Creates a user with the specified details unless user already exists. Returns 1 on success and 0 on failure.

login(username, password) - Logs on the user if the correct details are specified and sets up session for the user. Returns 1 on success and 0 on failure.

logout() - Clears the user’s session.

Task 3 - Secure Queries (1 Mark)

One of the most common ways of hacking into website is by exploiting unsecured SQL queries are vulnerable in order to commit SQL injection attacks.


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

Courtesy: http://xkcd.com/327/

For this task, you should investigate SQL injection and implement sanitation for the inputs on the Sign-Up page. Explain to your tutor how your database is protected from malicious inputs. You will be required to show the code to your tutor to ensure understanding.