CVWO Assignment 1

The aim of this small website is to create a simple blog with basic functionality. The use cases are as below:

Use cases

Login

Each writer of the blog will have their own account and password, which they will use to login into the blog. A logged-in user will have special privileges and available actions as compared to visitors who are not logged-in. Details on these actions are laid out below in each use case description.

Registration will not be implemented. All user accounts will be manually entered into the database initially.

Blog posts

Posts are the main feature of the blog. The landing page of the site would be a paginated list of the blog posts. Posts are shown in reverse chronological order. This request will be known as **list**.

In addition to **list**, clicking on the posts themselves will lead to a single page with the blog post itself, with it's accompanying details. This request will be the **show** request.

The above requests can be performed by any visitor. The below requests requires the user to be logged in as a writer.

Writers may **create** new blog posts, each with a title, a text of content as well as a recorded timestamp of creation. The new blog post will belong to them, and only they may edit or delete it. They may also **edit** a post that belongs to them, leaving a distinct timestamp for last edited time. Finally, writers may **delete** a post that belongs to them, removing it from the blog permanently.

Comments

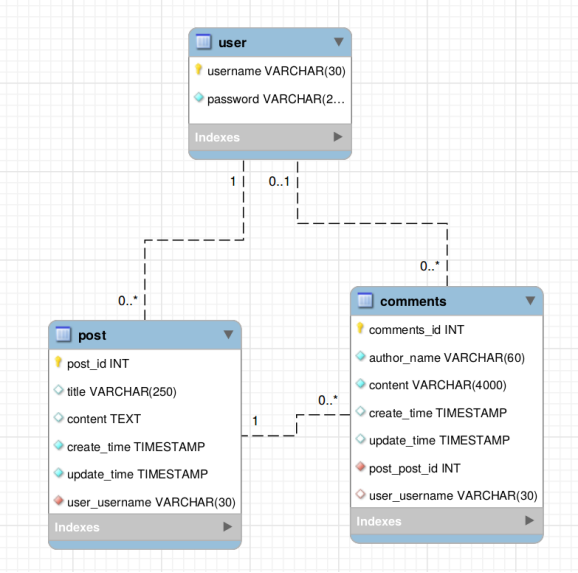
Each blog post will allow for comments from both the public and the writers. Comments will be **list**ed on the post's **show** page, at the bottom. Anyone may also **create** a new comment with a form on the post's **show** page. For writers, they may choose to comment with the identity of a writer, or they could enter a custom name, thus commenting as another visitor.

A writer may **edit** comments they have posted as themselves (if they choose to post as a visitor, they may not edit their comments). Furthermore, writers may **delete** their own comments and any other comments made by a visitor. They however may not delete comments made by other writers.

Implementation

Database

The schema for the planned database is located in `'db/blog.mwb'`, viewable in MySQL Workbench. An image is attached below for convienence.



Framework

The blog will use MINI (https://github.com/panique/mini), a very lightweight framework for PHP that follows MVC principles approximately.

Model

Each of post and comment will have a model. The database queries will be embedded within the model itself for simplicity.

Controllers

Both post and comment will each have a controller. Post will cater to all the requests above, namely **list**, **show**, **create**, **edit**, **delete**. In addition, two more methods for **create** and **edit** will be added to handle POST requests.

The comment controller will only have the POST versions of **create** and **delete**, in addition to 2 methods (GET+POST) for **edit**.

A final controller handles the login, by serving the login page, as well as processing the

login POST request.

Login

Logins will be handled with a session variable. The user will authenticate via a POST request. If the login was found to be valid, a session variable of the user's username will be assigned. Further page permissions will consult this session variable whenever neccessary.

Implementation will be roughly similar to what is laid out in http://www.phpro.org/tutorials/Basic-Login-Authentication-with-PHP-and-MySQL.html.

UI

The website will use Bootstrap for UI customization. Each of the GET request above will have a view associated with it.

Templates which specifies layouts like the sidebars/topbars will probably be defined in a seperate php file, and will be loaded via `require_once()` whenever neccessary, instead of being copied over to the various views.