Your First Blog

Your task is to create a Blog web application using the tools you learned about in class such as PHP, MySQL, HTML, CSS, JavaScript/jQuery. You must NOT use frameworks or PHP libraries (templating, database access, etc), or helper libraries other than jQuery.

Below you will find descriptions of pages and their functionalities that you will implement under each URL. Make sure you read all instructions carefully. You are responsible for asking the instructor for clarification as needed.

List of URLs in the application:

- `/login.php`
- `/register.php`
- `/index.php (also / )`
- `/articleadd.php`
- `/article.php`
- `/logout.php`

You are welcome to use helper files, at the very least you are required to have `db.php`.

NOTES

- In all cases of forms you should use the 3-state-form technology you learned about in class.
- In all forms you must inform the user of the reasons for an unsuccessful submission so the user knows **how** to correct it, not just that there was an error.
- For the pages that are only accessible to an authenticated user, if an unauthenticated user attempts to access them you should display a message similar to "Page accessible to authorized users only. You must log in or register to access it." with links to Login and Register pages.
- All validation must be done server-side, in PHP. If you like, you can add validation in JavaScript once you validate server-side.
- In all cases where user input is accepted from a form, (login name, password, article, etc.) or as a parameter to a URL (e.g. `/articleview.php?id=11` to view article #11) you **must validate** the input data and guard against parameters that would derail your application.

Please have a look **first** at the other document with website mock. Below you will find comments about each page but **only** on elements that are not obvious in the mock document.

COMMENTS ON SPECIFIC PAGES

LOGIN
No additional comments.

REGISTER / USER ADD

- User name must not already exist in the system. You must check that before you attempt to insert the new record as one of possible submission errors.
- User name must be at least 4 characters long and no longer then allocated database record (20 characters)
- User name must only consist of lower case letters and numbers.
- Password must be at least 6 characters long and must contain at least one uppercase letter, one lower case letter, and one number or special character. It must not be longer than 100 characters.
- Passwords must match for the user to be created.
- Upon successful creation of user display message confirming it and with a link to login page.

ARTICLE ADD

- Article title must be at least 10 characters long.
- Article content must be at least 50 characters long.
- User must be logged in to access that page.

ARTICLE (VIEW)

- The URL must contain the ID of article, e.g. `article.php?id=11` to view article #11.
- Only an authenticated user may submit comments. If a user is not authenticated then comment form is not visible but instead it displays text similar to "You must log in or register to post comments" with links to Login and Register pages.
- Comments are submitted using POST, and must be at least 5 characters long.
- You show all comments to that article (no limit).

INDEX PAGE

- The top right two items are exclusive, so the page either
  - Allows to login or register if the user is not authenticated
  - Allows user log out or submit a new article using /articleadd.php (not visible in the mock) if user is authenticated.
- Only the last 5 articles are displayed.

DATABASE STRUCTURE

Table 'user' with the following fields:

- id (integer, auto-increment, primary key)
- name (varchar 20)
- email (varchar 150)
- password (varchar 64)

Table 'article' with the following fields:

- id (integer, auto-increment, primary key)
- authorId (integer, foreign key linked to user.id)
- creationTime (timestamp, auto-assigned default value of current_timestamp at record creation time)
- title (varchar 100)
- body (varchar 4000)

Table 'comment' with the following fields:

- id (integer, auto-increment, primary key)
- articleId (integer, foreign key linked to article.id)
- authorId (integer, foreign key linked to user.id)
- creationTime (timestamp, auto-assigned default value of current_timestamp at record creation time)
- body (varchar 5000)

THE LOOK AND FEEL OF YOUR APPLICATION

Here are the guidelines of how to use CSS to format your content to obtain full marks:

- Content should be formatted to the center of the page, to width 960 pixels. Left and right margins of your web page should be filled in with a tiled graphics.
- Your blog should have a graphical banner on top or top-left of the page. You can generate one online using one of “text to banner” online services.
- Your blog should use colors palette that is pleasant to the eye. Consider using https://color.adobe.com/ color generator to generate different colors for your blog background, heading text colors, etc. Use of primary colours (red/green/blue) only is not enough.
- Formatting of elements should be done using DIV element and not TABLE tags.
- Every form should be inside a DIV with a border with rounded corners and a shadow.
- Error messages due to form submission should be presented in eye-grabbing color/background color that is part of the color palette chosen.

- Use a different font typeface for article titles.

VALIDITY OF HTML

You should validate the HTML generated by your pages using https://validator.w3.org/ service. Your HTML will be validated as part of your evaluation. You must use either HTML5 or HTML 4.01 Transitional document types. I recommend you do it as the last step of your application testing.