COSC 360
Blogster - Implementation Document

blogster

Siddesh Nambiar, Felipe Portela Chalhub, Jarin James, Yidu Guo

Details:

The implementation of the website, Blogster is as follows. We chose to implement the website with the help of the languages, HTML, CSS, PHP, JS and SQL. This was used in multiple different approaches and included a front end that was established using HTML and CSS. The backend was established using PHP, JS and SQL. This was done using the help of PHPMyadmin. Below is a list that was provided as a rubric in our submission document. The list will enable us to further discuss the implementation and how the coding was conducted. This document hopes to serve as the summary document of the implemented features.

Feature:

1. Layout Document (Page Diagrams) - The layout document as a part of our page diagrams is available to view from our GitHub repository. The Page diagrams however include a few visuals that are different from the final implementation. These were changed along the way. The page diagram was a good start for us to establish a base.
2. Organization of Pages, Site map - The site map can be found in the Page diagrams folder as seen above. Here there is a flow of how the pages are linked and can be traversed through.
3. Layout with contextual menus (i.e., when user has logged on to site, menus reflect change) - The menus reflect the changes of what the site page is, logged in and logged out. It reflects the change.

4. 2 or 3-column layout using appropriate design principles (i.e., highlighting nav links when hovered over, etc.) - The buttons and pages highlight and show when the mouse is over it.
5. Navigation Links on the top - All the pages on the website include a navigation bar that allows you to traverse through pages.
6. Masthead and footer - Where applicable, every page has a mast head and a footer.
7. Navigation links regardless of where the user is - The navigation links will always be available to users on all pages.
8. Breadcrumb strategy - Many pages have a sense of the “where the users are in the threads” this would be evident through the nav links.
9. Page for user registration - The signup page is available for users to register and the user will be stored in the database.
10. Form entry along with image - The user can enter their information along with an image upload through signup.
11. Form validation through JS - Where applicable and because this is a requirement, the implementation' through JS is shown on the login page. For some other pages, the “required” form capability is used.
12. Responsive design - The design however cannot be implemented on a mobile phone, will still work, since most of the work is relative to the user screen in css.
13. Preliminary summary document for server side indicating implemented functionality - this document wishes to serve the purpose of the summary document. The server side implementation was done as mentioned before using the PHPMyadmin and xampp. The database was built on the server and an sql code was provided for the same. This enabled us to connect to the database and transfer data between the server and the local form. Using this, we were able to communicate data between the servers and pages. The database is shown in a screenshot in the walkthrough document.
14. Post Storage/Discussion Thread storage in database - The post that a user posts, is stored into the database. We chose to use a Twitter style approach and this would mean that we allow for users to create 240 character posts!
15. User account information stored in database - All the users information including, username, id, email, pass, blog content all of these are stored in the database in tables
16. Appropriate security for data - For the safety of the users, we implemented hashing for the user password, the user’s password will never be visible to us on the server side or the front end.
17. Site must maintain state - The site maintains the status of being logged in or logged out, if the user logs in, they get to post, however if they are logged out they can only see the other users posts but not be able to post.

18. Error handling - Where appropriate, the errors have been handled to ensure flow.

User Functionality -

1. Browse Discussions without registering - The users can view posts without having to register onto the page, this is done by Home -> Explore and they can view posts but not post.
2. Register on site using username and email and image - The user can register on the site by creating a username, email and image, the username is always unique amongst users.
3. Allow login using username and pass - The user will be able to login with the credentials they provided.
4. When logged in, the user will be able to create posts on their user-blog page, and they can view and comment on posts there as well.
5. View and Edit profile - The user will be able to edit their profile and profile picture on their user page and here they can change their email and password.
6. User password recovery - the user will be able to change their password on the profile page, this is not done using their email but can be changed directly on this page. Once again, we will not be able to see the user's password at any time.

Admin Functionality -

Additional Functionality -

1. Hot threads, Post tracking - this was done by using the number of likes and was displayed using "order by" in mysql, based on the likes stored.
2. Alert on page changes - When a page is updated, if the user enters an incorrect value, it automatically updates the page and displays the required informations
3. Tracking comment history - the user's comments are tracked by the user blog page and the user can see all the comments they have posted.
4. Activity by date - All the blog posts will have the functionality to be tracked by the date, when the user posts, the date that the user posted will be displayed
5. Resolve user problems - The user will be able to change their username and email on the profile page along with the fact that they can also change their passwords on this page.

There are a few limitations to the website which is mentioned in the explanation as well. One of which is the password recovery via email. This could not be figured out by us for some reason, since we would have to use a mail server too, we tackled this by using a direct recovery instead. In the future, we would love to make the pages a little more efficient in the functionality and how it was implemented. Some things like, automated new pages, when the page exceeds the blogs, or adding a better UI for the comments section, even using better coding methods, like prepared statements, interfaces for methods or much better password encryption is something that we would love to work on in the future.