Web Technology Assignment 5: Final

By: Diar Masri & Nate Fear

HTML

For the HTML part of the website we designed an image sharing web application which allows users to upload, catalogue and categorize their images. All images categories can be searched by visitors of our site, however users must register to upload images this is so that we can collate a user's images for them. Upon registering the user is able to log in to the web application to upload images as well view their already uploaded images. The HTML page consists of a header, main photo gallery and larger photo viewer as well as many pop up forms. Whilst there is server side form validation we mostly rely on HTML5 validation to sanitise data of our HTML pages; we also use the stricter XHTML typing to ensure that our pages will be displayed correctly on as many browsers as possible.

CSS

The CSS component of the website consists of typical stylings which allowed us to portray the web page according to our wireframe designs (See Figure 1). As well as to provide interactive features such as animation via hover on elements the user should interact with and also using the hover styling to create drop down menus. Moreover, CSS has also been used to allow us to provide a responsive web application which will work somewhat near optimally on any display size via the use of media queries rules which change the CSS of some content based on display size.

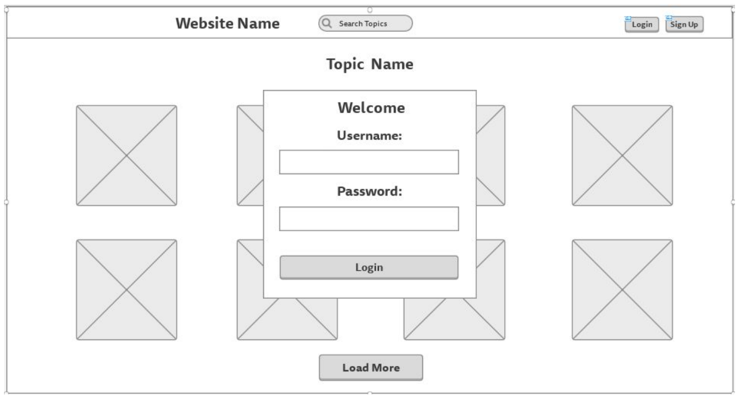

Figure 1: Wireframe Design

JS

Client-side JavaScript has been used to open and close pop up forms, perform form validation, index the gallery images and their descriptions, so that the images can be displayed in a larger viewer mode in which the user can cycle through the images and their descriptions using next and previous buttons. It is also used to recognise particular GET data strings in order to respond with the correct message box for the event, as well as to construct the appropriate GET data string and redirect in order to trigger server side functionality such as image deletion by passing an images ID.

PNG

The images created in Inkscape were only contours; therefore it was necessary to add a colour layer. Graphical adjustments were performed in Gimp and provided a large variety of colour editing such as airbrush, area fill and tools such as blur, smudge and burn that helped to create shades and 3D illusion. These adjustments to SVG images are present in all images in the section PNG in the image gallery.

(files: dolphin.png, duck.png,seagull.png,seagull2.png,8a.png)
We then took the coloured artwork and merged it into couple of Bristol images that we downloaded from the Internet.
(files: 1.png,2.png,4.png,5.png,6.png,8.png)
Mostly we applied different types of layer masks, transparency adjustment and transformation tools to replicate the artwork into various scales and orientations. The layer functionality allowed us to place artwork in between objects present in the original image as is seen in the image with the seagulls at the riverside. Moreover, the layer functionality allowed us to adjust only particular sections of the image such as colour palette, colour levels or create smooth transition between original image and artwork. The combination of downloaded images and our artwork is presented in section Mixture of the image gallery.
(files: 1a.png, 2b.png, 2c.png, 4b.png, 4c.png,5a.png,6b.png)
Adjustments:

- 1a.png - layer mask, scale tool, rotation tool, select tool.
- 2b.png - layer mask, scissor tool, color levels of the water adjusted, blur tool of the water under dolphin.
- 2c.png - transparency levels, colour palette adjustment, scissor tool.
- 4b.png - select tool.
- 4c.png - transparency gradient of each layer used to create smooth transition between layers.
- 5a.png - scissor tool, selection tool, colour palette adjustment.
- 6b.png - rotation tool, scale tool, layer masks, scissor tool, insertion of one seagull between the ring and sky.

SVG

The SVG part consists of graphical artwork created in Inkscape tool. The SVG artwork is presented under section SVG in the image gallery.
(files: dolphin_line.png, duck_line.png,seagull_line.png,bridge.png,seagull2_line.png)
Multiple features provided by this tool were applied such as curves, paths and their grouping and editing. Moreover, we managed to apply freehand drawing tool, corresponding smoothing tool and the calligraphic tool. All the presented techniques were applied to all the artwork in section SVG. The piece of work created in Inkscape served as a contour foundation for consequent colour editing of artwork in Gimp.

Server

The server side of the site is provided only in Node.js without usage of any framework.
Nevertheless, we managed to implement an image search based on the name of topic and ID of the user that uploaded the image. This allowed us to create image gallery based on HTML template that is filled in with data provided by respective predefined SQL query. Moreover, the SQL database allowed us to restrict access to particular sections of the website with passwords stored in the database. Therefore, only registered users are allowed to upload images and each user can view images from the database based on his username when on the home page. In the case, no images were uploaded it is marked

respectively. The uploading of files is performed using a form with predefined image categories and the image file itself which is stored in the upload folder among all other images.
Moreover, the server side contains validation of forms with respect to already assigned usernames and correspondence between the two password entries. Furthermore, the form validation is applied during login where entered username and hashed password is checked against data contained in the database. To avoid the user having to re-login every time when switch between the index and home page sessions are used for short term persistence by setting a cookie upon login with an obscured representation of the user's id. This cookie is saved by the client and sent back with each proceeding request, meaning that the web application can authenticate the user using the session data and look up their uploaded image upon accessing the home page without re-logging in providing this is within a given time window. Lastly, the server also provides the user with image deletion functionality because when a user clicks delete on an image the server receives an URL with GET data along with an image id, which the server then uses to verify that the image actually belongs to the user issuing the query and then deletes the image from the database causing it to disappear from the user's home page.

Database

The database for the website is provided by an SQLite database. Therefore, we created a script to initialize the database with respective tables and fill them with our artwork. The database consists of four pages:

- The image table stores image ID, text description, path to the file and ID of its owner.
- The user table stores user ID, hashed passwords and random data used for cryptography.
- The topic table stores all of our topics names with their IDs.
- The im2top table is an mapping table between image table and topic table.

The relationship between images and users tables is M:1 as each user can own various images but an image can be owned only by a single user. The mapping between images and topics is M:N as image can have two topics (assigned and “all”) as well as a single topic which can be assigned to multiple images. The security of SQL queries is managed by creating series of predefined SQL commands that allow to insert, delete and query the data in the database. Moreover, the database structure allows us to restrict access to the deletion procedure to only the owner of the image.

Dynamic Pages

The dynamic pages are delivered using EJS JavaScript template which allows easy insertion of data in JSON format into predefined template structure. The template allows us to create the image gallery with images acquired from the database in a way that the result of database query can be fed into the template. The structure of the template allows us to create loops and conditions in HTML code by special tags and when corresponding data is provided it merges all into final HTML page. This approach allows creation of similar pages using a single template structure. Moreover, it allows us to make changes in the image structure based on the request received or response to the current query.

Depth

Aims

The aim of web project was to display the intricate art work we created and altered using the various open source image manipulation tools on an artistic and art centric website. To achieve this goal we designed and implemented a web gallery which enables our artwork to be experienced from various perspectives. Upon being able to display our own artwork we built on this with categorisation so that collections of a specific type of artwork can be enjoyed at once by searching for categories. To share this with others and to allow them to share and display their artwork online we added user accounts, so that visitors could do more than just view the categories of our artwork by registering and uploading their own work which would then be added to public categories, as well as displayed all in one collection within the user's home page for the user's own enjoyment as well as so they can show their work to others. To conclude we're particularly proud of our detailed art work and achieved our goal of building an online art gallery and extended this many times over to allow other users to share their artwork with us.

Security Extensions

Hashing passwords along with random salt data has been used to secure passwords so that in the event the database is compromised user passwords are secure because attackers cannot easily obtain passwords by performing a reverse lookup using a hash dictionary due to the fact that hash dictionaries will not contain arbitrary salt strings.

Efforts have been made to prevent SQL injection and database compromise by using prepared SQL queries as these ensure that SQL code and data are not mixed by sending the SQL query and the user data to the server separately. This step prevents an attacker from being able to send malicious SQL code as user data and it being interpreted as code and executed.

Further endeavor has been spent obfuscating the way we use session data to extract user information, so that it will be less trivial for an attacker to be able to spoof the session data of other user for unauthorised access.