Coursework Report

Scott Postlethwaite
40281026@live.napier.ac.uk
Edinburgh Napier University - Module Title (SET08101)

1 Introduction

The aim of this assignment is to design and implement a blogging platform which will allow the user to create, delete and edit blog posts with an element of persistence which will save blog posts to a database. For the server side elements I have used Node.js, Express and Embedded Javascript (EJS). For persistence I have used a Realm database thanks to the ease of use and plethora of tutorials.[1]
This blog has been set up for a single user to write their blog, myself. This keeps the focus of the blog centered on one topic and experience making for better styling.
The topic and content of this blog were inspired by a current trend in the cycling world where BMX riders are swapping out their small trick bikes to large slopestyle mountain bikes. I have a lot of views on this trend as I am making the switch myself and I feel that it is better to have real life posts populating the blogging platform than just placeholders. This also allows for a theme to focus on while styling. This can be seen in my site by the wheel acting as a menu in the top left corner (Which matches the wheel on the banner) and the continuous colour matching to the banner.

2 Software Design

2.1 Server Side

The server aspect of the blog is upheld through node.js. The html is generated by an ejs document used as a template to be filled in with posts from a database.
Throughout development I have used nodemon to refresh the server any time a change is made to any of the files. This ensures that any edits I make to the code is displayed on the screen. This gave me a far greater understanding of node.js as I learned which of my changes made it tick and from there I learned what to and not to do.

2.2 The HTML

The main page's HTML will be generated with embedded JavaScript in order to display the posts taken from the database by node. The other pages of the site will be directly created through html. The general look and feel of my site will be very simplistic taking design tips from the likes of facebook, twitter and instagram where all posts are displayed in chronological order for the user to scroll through. All of the relevant information will be centered on the screen with a logo at the left corner. The logo will act as a hidden menu as upon hovering it will display each function available to the user in the form of links. To make use of hidden menus and text I will first inspect websites that I use myself in order to see how they have implemented similar features. I will also make use of the W3 schools tutorials[2].

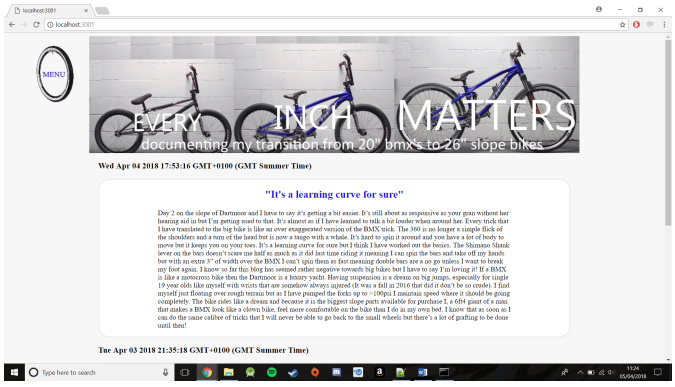

Figure 1: **Main Page** - The main page of my blog)

2.3 The Javascript

The password protecting of each function were simple JavaScript queries. if the text input matches the string initialized in the js file you are allowed access to the site. There are better and more complicated ways to implement this such as allowing users to register, change their passwords and log their sessions however I felt that as this site is only intended for me to post, I have only locked these functions behind a single hard coded password. There could be a database entry containing the user name and password however I felt that this would over-complicate a simple system and would slow down the site.

2.4 The CSS

The design of the site was inspired by Microsoft Word where posts appear as pages in front of a slightly off white background. I have complimented this with a slightly darker boarder around the post to give the allusion of three dimensions.
The colour scheme of the blog is simplistic yet effective. The colours present in the banner, the slightly off white wall used as the background, the blue of the frames used for the titles and the white walled tires matching the background of the post. While at first glance it may seem too basic however I would argue that over styled blogs such as "Belle de Neige"[3] distract the reader's attention too much and make the content itself harder to read and enjoy.
As far as the menu items are concerned they have a hovering logo that stays in the top left corner which hides the menu items beneath it. This is done through a container holding

the logo and each menu item. The menu is then hidden and upon hovering over the container, the menu is released.
The menu is fixed in position and size where everything else on the page is dynamic. This ensures that the menu is always accessible by the user no matter where on the page they are looking.

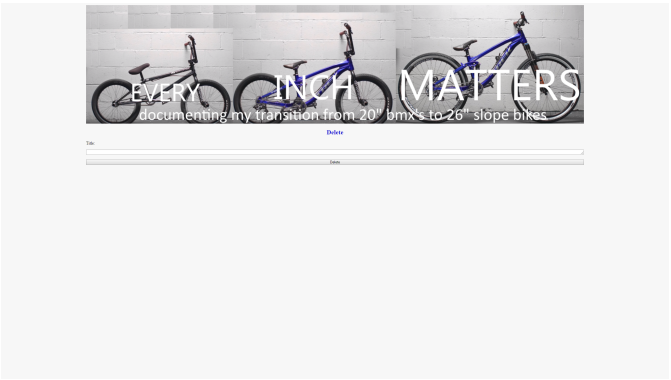

Figure 2: **Delete Page** - The delete method of my blog)

2.5 The Database (Realm)

The database used was chosen thanks to Realm's plethora of tutorials[1] on how to create entries, edit them and delete them. The blog consists of a series of 'Post' entities, containing the TimeStamp (Current date and time of the post), The Title which is used as the primary key and the Content of the post it's self. As the title is used as the primary key it cannot be edited but can be used to edit the content of a blog post or to delete one.
I achieved this by referencing Realm's JavaScript tutorial[1] and by following their simple Blog tutorial[4] for writing.
Each feature was written with a lot of experimentation. I ran into a series of problems when writing the Delete function. Initially I used a filter to find the blog post which had a title matching the title provided by the user however this ended up deleting all blog posts. To get around this I actually edited the post before deleting it to ensure that the correct post was being deleted. This worked effectively and made the implementation of editing far easier.

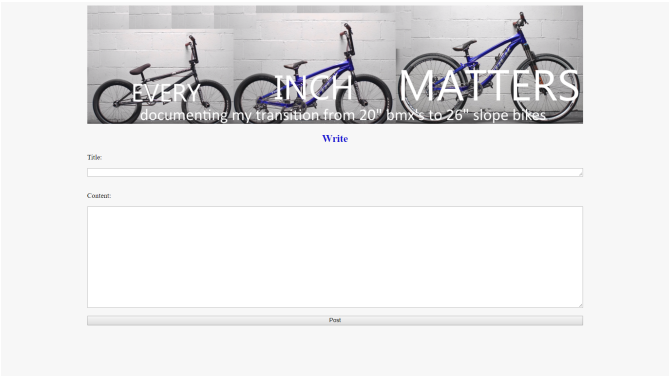

Figure 3: **Main Page** - The writing page of my blog)

3 Implementation

The implementation of this site is very simplistic. Blog posts can be added, removed and edited in a database. Each of these functions is hidden behind a password which is hard coded in the JavaScript of the site. There is no way to format the text like you would in latex. That isn't to say that the site is not well designed and pleasant to use as I have spent a lot of time perfecting the physical design and CSS of the site. I have used a variety of methods such as hover tags, dynamic sizing and colour matching to make my site look professional and pleasing to the eye. I feel that this is my sites greatest strength. Even though it looks simplistic, each design decision was thought out after lengthy research of other blogs, user testing and simple interviews with potential users to see what styles they prefer in a website/ text editor. Eventually the Word-esque style was chosen with subtle hints of blue added to compliment the bikes in the banner. There are many improvements that can be made to the blog which I will discuss bellow however as a beta product I feel that it meets the user's needs and is pleasing to use/view.

Figure 4: **Menu Icon** - The Menu Icon in my site

4 Implementation Evaluation

4.1 Does It Meet The Spec?

The User is able to post, edit and delete blog posts. These are persisted in a realm database and each function is hidden behind a password. There is only support for one user as I feel this makes the design more focused on one topic allowing me to design the site with better continuity.

The posts can be viewed easily on the site as it works in a similar way to facebook, instagram and twitter which are proving the most popular ways to consume media. This has the same seemingly endless scrolling filtering content in reverse chronological order. Because the site has been designed for only one user there is no way to filter posts or view them in another order however if I were to change the focus of the site to something that would allow for multiple writers a filtering system would be easy to implement as Realm has an

inbuilt filter method and the posts would be displayed in the same way as before but will only show x's posts. (let xsPosts = realm.objects('Post').filtered('writer == x');)

The writing portion of the blog has similar styling to Microsoft word with the white text area over the slightly off-white background. It contains the same banner as the blog for design continuity and brand recognition. through research I have found that users respond better to web pages keeping the same design with only subtle changes. As far as advertising and brand recognition goes, a well designed, subtle, ever present logo or banner has proved one of the most effective ways to drill your brand, site or application to your user to keep them coming back. That is why I have implemented similar ideology in the blogging site.

Deleting a post is simple, all the user needs to do is type the title of the post that they would like to delete and click delete. The only things that I would consider adding to this is a review page which will show the user the post and ask if they are sure. Other than that I feel my method is as easy and user friendly as possible.

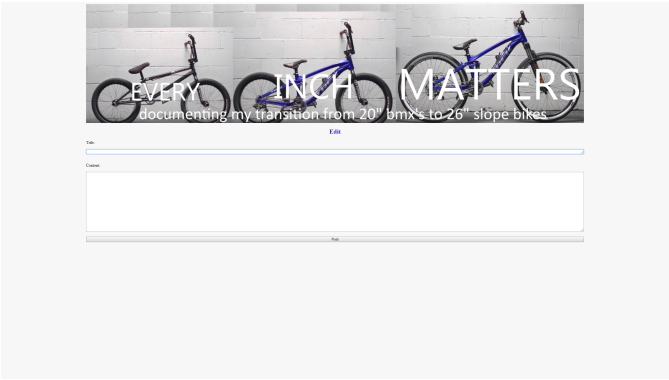

Figure 5: **Edit Page** - The edit method of my blog)

The physical design of the blog is the highlight of the blog. I have created this with the use of hover tags, dynamic resizing, well designed banners and logos and extensive testing to create a pleasing colour palette that compliments these. The design is well thought out and pleasing to use. The only negative feedback that I have had from users is the lack of formatted text however as that is a problem with the Realm database used for persistence, there is nothing I can do about that without completely redesigning the server side of the site.

The Server persists the posts within the realm database storing not only the post but the time and date of it's creation. This allows the posts to be displayed in the correct order. While researching blogs I also noticed that the better designed blogs show the date above the post itself so that is what I have done.

4.2 Possible Improvements?

I could also create a postId to act as the primary key so that the title could be edited. The reason I have not done that in this site though is that if the postId was auto incremented you would quickly lose track of which post corresponds to which id and if that Id was displayed it would make no sense to the user. A possible way to implement this is by hiding the id and having it appear when the editor logs in however I feel that entering the title of the post that you would like to edit is still more user friendly.

Due to the nature of the Realm database used, there is no way to format the text. This means that all posts are posted as one lump of text. An improvement to this would be to add support for new line characters to make the blog posts look nicer and become easier to read in the process.

Figure 6: **The Banner** - The Banner I created using the transition pictures of Drew Bezanson's bike

5 Personal Evaluation

While I am proud of what I have achieved, I would not let this site go live as there are too many features left on the shelf. The only issue that I am yet to resolve is the lack of ability to format your text. Due to the simple nature of the Realm database it cannot store new line characters making everything you input end up as one block of text. To resolve this issue I would have to change the database to mongoDB or something similar before making the site live.

Other than that I feel that I have performed well especially in the physical design of the site. The CSS and overall styling are what I am most proud of. I feel that the research and experimentation payed off and have made the site look pleasing and professional and these skills will stick with me throughout future projects.

The server side aspects of the site were completely new to me and I am pleased that I have managed to develop a site that not only generates the HTML from an embedded JavaScript document that I didn't even know was possible beforehand but that it also manages to enable you to write and edit a database with full persistence on the site. I couldn't be happier with the server side of the site.

I have faced many challenges throughout the making of this blog. The most notable of which is to do with the deleting of a post from a database. I was under the impression that this was working perfectly as when I had tested it there was only one post on the site however I later discovered that when I tried to delete one post it deleted all posts from the database. This was due to problems with filtering the posts by the title. I knew that my editing worked perfectly by selecting the post based on it's title, the primary key, and effectively writing over it. This obviously wasn't what I was hoping to achieve when deleting a post but through experimenting with filters I realized that was not going to work so I effectively edited the post before deleting it so that the correct post was to be deleted. That post was then removed from the database.

My second challenge was with dynamically shaping the banner and background. to begin with everything was fixed meaning if you have a 1080p display you would find the website pleasing to use but if the site was windowed or used on a lower resolution display, the border will extend past the screen. This makes my site almost inaccessible to anyone with different specifications to myself. As I am unaware what computers my site will run on I had to make the text's border dynamically fill the screen. This caused me a multitude of problems as initially my border would fit in the screen but would shift my text too far to one side. In order to then center my text I had to put my background on a second div two above the body so that it would not affect the layout of the text.

Placing a drop-down menu under my logo was another challenge as it would not allow me to hover over the image id as I then wouldn't be able to select an option whereas if I were to set a div, the border that was surrounding the logo it would be an awkward shape and would still not fit the entire menu.
Eventually I set a div around the logo and drop-down menu, set an on hover which would make the drop down menu appear and removed the border which actually made the site look better.

Possibly the biggest challenge has been trying to pull the post's text from the database and to populate a text area with it. This was made more complicated thanks to node not supporting document.getelementbyid. The solution has been to create an ejs file which contains the content of a select blog post in a text area. This content has been taken in very much the same way as editing a post where the title is entered and checked against the titles in the database. Fixing this issue gave me a great understanding of the workings of node and embedded JavaScript.

References

[1] Realm, "Realm javascript 2.3.0," 2018.

[2] w3schools, "The world's largest web developer site," 2018.

[3] Anonymous, "Belle de neige," 2014.

[4] Realm, "Building a blog with realm node.js and express," 2018.