Edinburgh Napier UNIVERSITY

Coursework Report

Sonnan Naeem
40341713@napier.ac.uk
Edinburgh Napier University - Web Technologies (SET08101)

1 Introduction

1.1 Goal

The goal of my coursework assignment was to design and implement a basic functioning blog site using HTML, CSS, and Javascript for the client side of the platform and the Node.JS runtime environment for the server side of the platform. The client side of my blog platform would allow multiple users to be able to create an account in order to create, update, and delete their own blog posts and also read posts created by other users. I wanted to make the design as clean as possible for a smooth user experience and to also create a comment system to increase a sense of community and connectivity between users. The server side of the site would store the data related to the blog and also supply a (CRUD) API that the client side of the platform would use in order to provide functionality to the blog's features.

1.2 Overview

In developing for the site, I decided to use the libraries included in the MERN stack. I had started my development with a MERN tutorial that gave me an understanding on how to setup a basic CRUD API through Express so that I could learn and build off of that for the full blog [1]. With React being able to be used in both sides of the platform, Express simplifying server code, and MongoDB already being Javascript based, I had found that this stack would allow me to program the site in an efficient amount of time without much compromise and would allow me to execute the site as a reliable platform with robust features. In practicality, I had faced some challenges because I had spent more time trying to understand the React theory rather than implementing the website but in the end I was still able to accomplish the core features of what I originally had intended and implemented a fully working site.

2 Software Design

2.1 Sketches

Before I started doing anything else, I decided to draw sketches of some of the necessary screens that would be used in my interface. Doing this allowed me to visually see what it was that I needed to implement on the front-end and what the front-end might need in order to give a user-friendly experience. These sketches gave me a map to work off of in order to properly plan the development process I was about to undergo (Figures 1-4).

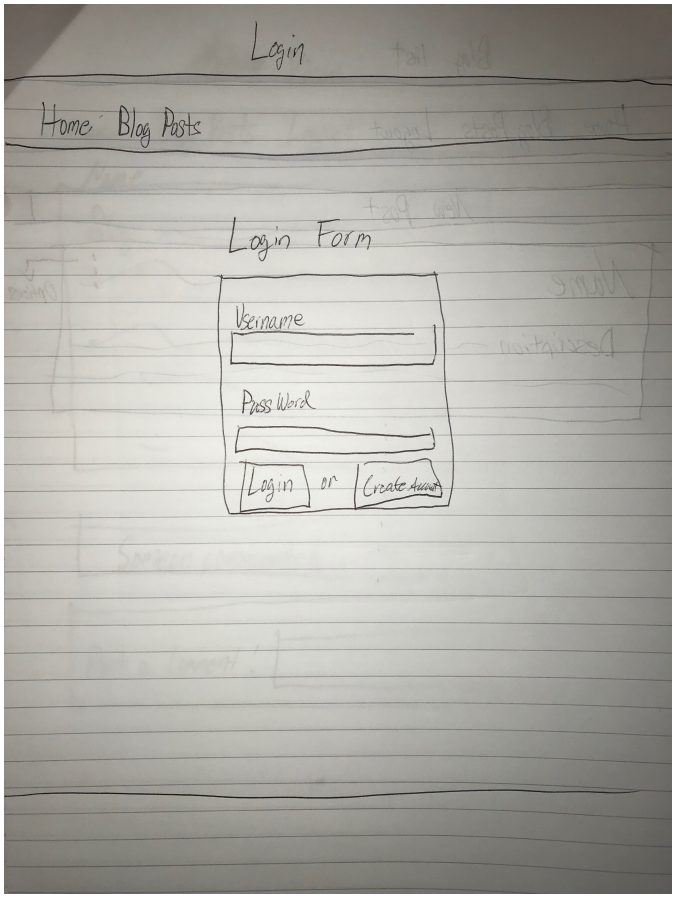

Figure 1: **Sketch 1**

2.2 Stack

Comparisons Next was researching what stack I would want to use in my implementation. Looking at the options, the two most popular stacks was the MEAN and the MERN stack. The first stack I had done research on was MEAN. I had read that one of the major benefits was the fact that with the Angular Framework, there could be immediate user feedback whenever input would be involved rather than having the user click a "submit" button to load the input [2]. Though the very same source that influenced my understanding of the MEAN stack also mentioned that with MERN, the React library allows developers to get up to speed more quickly [2].

MERN This is when I started delving more into what it would take to use MERN for my blog. I learned that React would be easier to learn if a developer already knows how to program in HTML and Javascript whereas Angular is considered to be something that has more of a learning curve

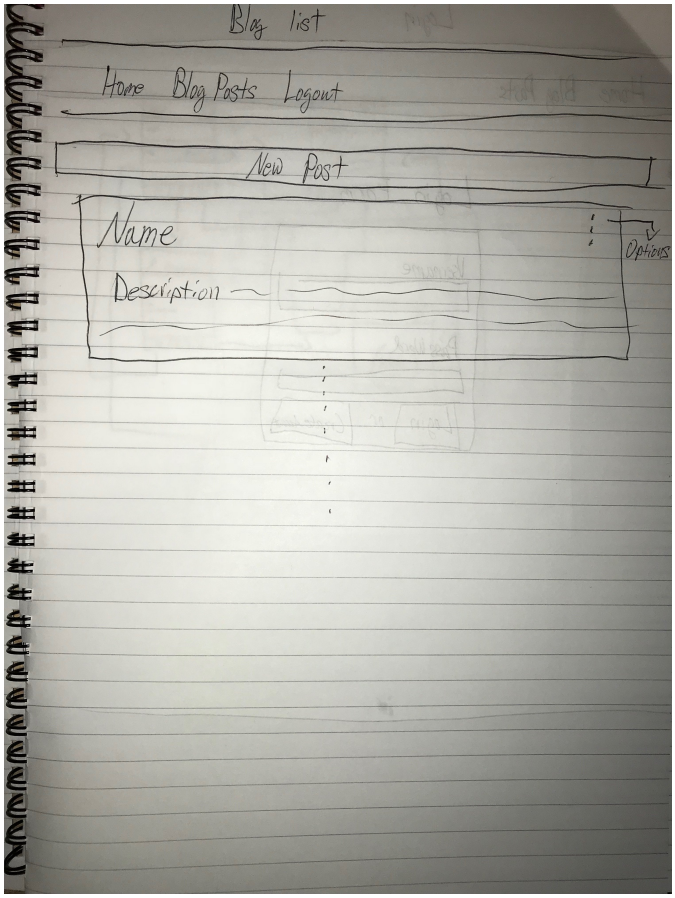

Figure 2: **Sketch 2**

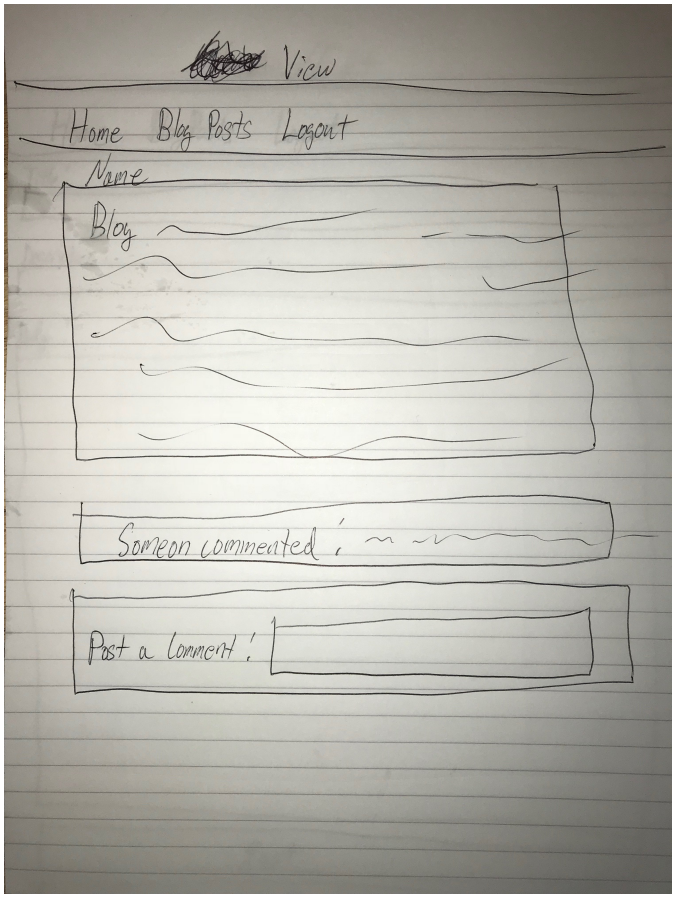

Figure 3: **Sketch 3**

[3]. Therefore, since I had to take into account my time constraint for completing this project and knew that Angular might be more powerful, I decided to use React since it would be quicker for me to pick up and use.

2.3 Inspiration

To then get a sense of how my site's design was going to be implemented, I decided to look for basic tutorials online to get a sense of foundation for the implementation. When looking around I had stumbled upon the site called Medium [4]. The minimalist style instantly captured my attention and made it easy for me to draw my eyes to important parts of the page. I wanted to use it as inspiration for the design of my site. This in turn led me to finding an article on Medium that gave a basic boilerplate tutorial to setting up the MERN stack. It worked out great since the research for design also led me to the first tutorial I read through when starting my implementation.

2.4 Use Case Diagram

Lastly, I decided to make a use case diagram before starting my implementation. The diagram allowed me to have a clear understanding on how the site would interact with the user, and the process of making it helped me keep track of the user experience (Figure 5). Therefore, I referred to the diagram, not only in my planning process, but also throughout my development in order keep tabs on my progress.

3 Implementation

3.1 Database

The first thing I did was implement a RESTful API by using Express and MogoDB just for simplicity. When it came to setting up the database I used MLab in order to set up a working server without having to run it locally on my machine.

3.2 Architecture

In the next part of my development, I decided to use a single page application architecture for speed and improved user experience. This is achieved through only reloading parts of the page that change rather than reloading the entire website when navigating through the page. To achieve this, I also leveraged component architecture in order to load and unload specific components on the page.

3.3 Navigation Bar

I first created a base component that holds all the global variables as well as the navigation bar which is always visible. Having this base component, allowed me to store global data that needs to be persisted from page to page. Additionally, it also allows me to load the navigation bar once instead of reloading it every time the user navigates to different parts of the site. It also acts as the entry point for React-Router which swaps out the components depending on the URL.

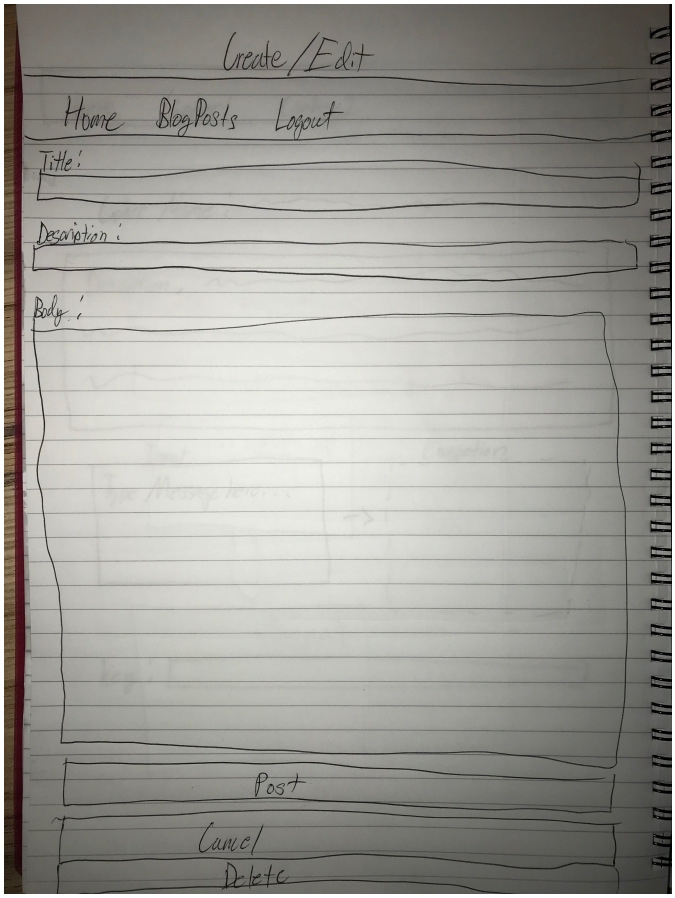

Figure 4: **Sketch 4**

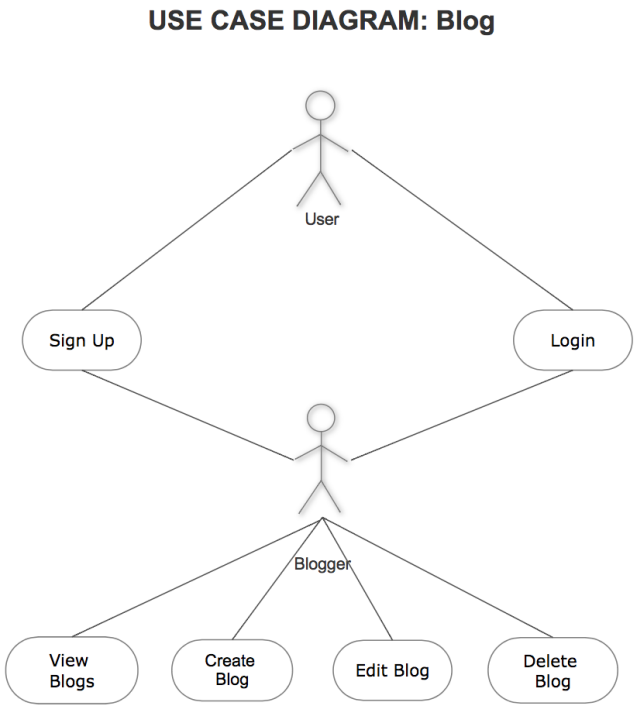

Figure 5: **Use Case Diagram**

3.4 Login Page

The next component I implemented was my login page. The page contextually displays a welcome screen or a login form, depending on whether the user is logged in or not. This form also allowed a user to create an account for the site. If the user is logged in, it will update the parent component (the application) with the user's information. This will, upon updating the application, also update all the children in the application with the user's information (Figure 6).

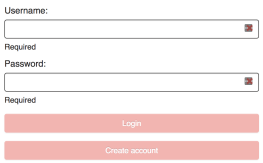

Figure 6: **Login**

3.5 Welcome Screen

Upon logging in, this component takes advantage of React's property life-cycle to then display a welcome screen if the user successfully logs into the site (Figure 7).

3.6 Blog Page

Next was the blog page. It is created in a way where the user can view any blogs that have been posted without having to be logged into an account. If the user is not logged, they are simply allowed to just view blogs, however if they are logged in, they will see more options on if they are the creator of the blog, such as edit or delete. Additionally, they would only be able to see the create-a-new-post button if they are logged in. This component also has an empty state which displays if there are no blogs and would prompt the user to create a blog (Figures 8, 9).

3.7 View Page

The view page is dedicated to displaying the blog to any user. It also gives the user edit and delete controls if the user is both logged in and also the creator of the blog post (Figure 10).

3.8 Blog Form Page

The last component is the blog form page which is one component purposed with two functions, editing and creating the blogs. This is done by querying the API if the blog ID is given and populates the appropriate fields with data received from the back-end. It also has live validation if the user inputs more characters than allowed and also live invalidation if the user goes over the character limit (Figure 11).

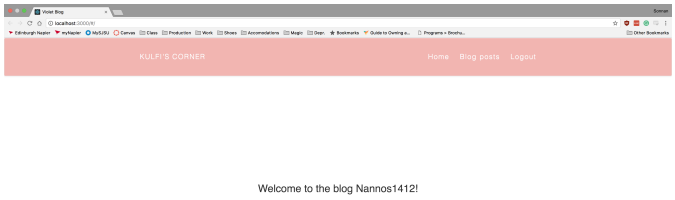

Figure 7: **Welcome Screen**

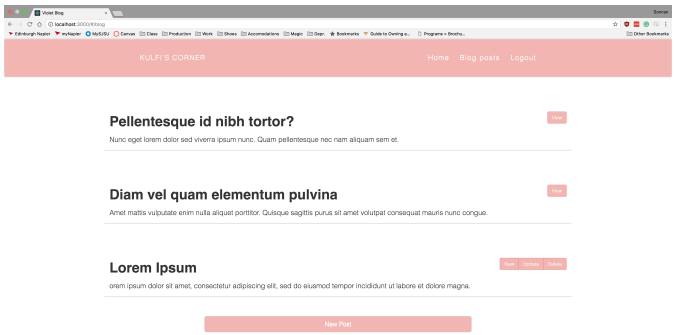

Figure 8: **Blog Page - Logged In**

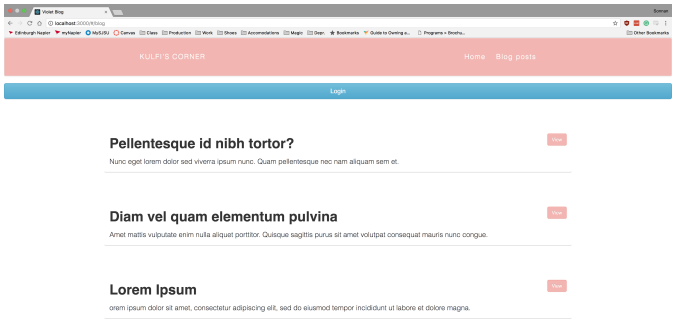

Figure 9: **Blog Page - Logged Out**

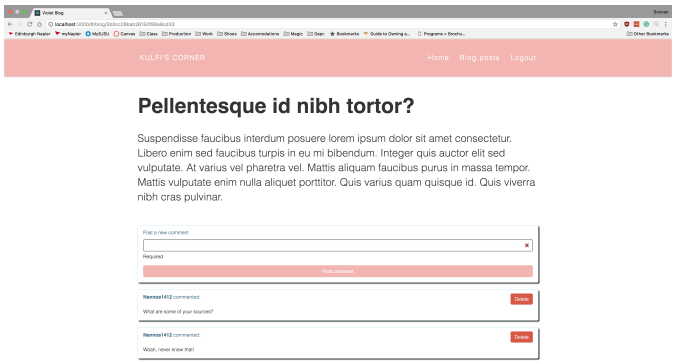

Figure 10: **View Page**

4 Implement Evaluation

4.1 Glows

In comparison to what I set out to do in this document, I completed the core functionality that was needed on the site such as the CRUD API and I allowed more than one user to create an account to access the sites features. The usability of the site is very straightforward and clear which allows users to understand how to use the site in a short amount of time. The site also does not require as much network usage than other sites due to the fact that only certain components that need to update do so on their own rather than refreshing the entire site. This in turn allowed for a smoother user experience.

4.2 Grows

Furthermore, there were some aspects of this site that I wish I could have programmed further, added, and improve upon. For example, I had initially wanted to implement more security around the multiple users feature but due to the unexpected amount of additional time spent on research, I had to focus more of my attention on the core functions of the site rather than the additional features. Something else I had a difficult time juggling was styling versus functionality. The site could have better styling for the user experience but more time was spent on the functionality instead. When looking at other sites such as Tumblr for comparison, I could have also added a search function or filters that could customize a users blog feed.

4.3 Conclusion

All in all, I had prioritized the usability of the site and smooth user experience over spending time on additional features, such as comments. Due to the time and the resources I had, the core functionality and the focus on site usability were my priorities. Given these limits, I feel as if I was still able to develop and build an effective and serviceable website.

5 Personal Evaluation

5.1 Reflections

When reflecting on the path it took to reaching the functionality of the site, I feel well accomplished. Though, in completing the project, I felt as if my end result did not match the high ambitions I had when I was in the early stages of the development. I had expected so much out of myself in such little time that it became overwhelming at times when forcing myself to try and complete everything that I had set out to accomplish for the site. However, while writing this report, I have realized that those ambitions were the main reason in motivating me in to completing the project to the

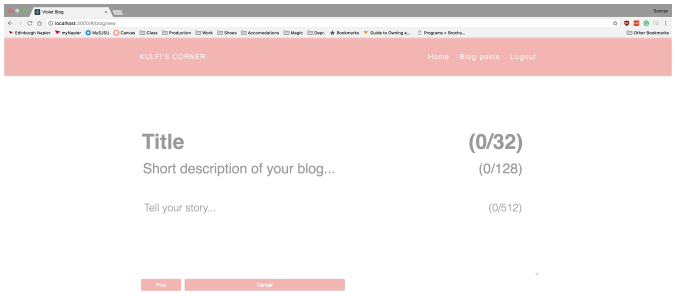

Figure 11: **Blog Form Page**

best of my ability. With the different methods and features I researched that excited me, it made me that much more invested in the project than I would have been if I was only doing it for the mark. Looking back there were a few things I wish I could have done better but there was also a lot that I had learned about myself when it came to my work-flow for a big project such as this one.

5.2 Challenges

There were many different challenges when it came to developing the blog but not all of them were technical challenges. My main challenge was anxiety when it came to tackling the project. It hit me the most after I had completed my research for my development. I felt overcome by the amount of different components that I needed to connect together and almost felt like I could not complete the site at all. However, I found that setting end goals for what I wanted completed by the end of the day for the project helped me breakdown all the components into smaller objectives that I could later put together. This was then combined with a decision of taking more time out to thoroughly read documentation rather than looking for answers in sporadic online articles. The decision would take longer than Google searches in the beginning but I found that with a little time, I was able to find answers to technical questions I was looking for a lot quicker with a deeper understanding once I knew how to properly navigate documentation.

5.3 Conclusion

Completing this project taught me essential skills on how to properly carry out a project in a short amount of time given the resources I had. My biggest take away was learning how to strategize and prioritize under pressure. I learned that it is okay to have high ambitions for a project even if the if it seema unrealistic. The important lesson I gathered was to use those ambitions in a way to propel me further instead of leading me to feel I had failed. I am proud of the fact that I was able to achieve creating a blog that is fully functional given the various pressures associated with this project. Above all, there was a sense of accomplishment in developing not only the site but also my own skills and strategies as a programmer.

References

[1] B. Gilbraith, "React getting started." https://medium.com/@bryantheastronaut/react-getting-started-the-mern-stack-tutorial-feat-e 2016. [Online; last accessed 22-May-2018].

[2] A. Morgan, "Introducing the mean and mern stacks." https://www.mongodb.com/blog/post/the-modern-application-stack-part-1-introducing-the- 2017. [Online; last accessed 22-May-2018].

[3] N. Pandit, "What is reactjs and why should we use it?." https://www.c-sharpcorner.com/article/what-and-why-reactjs/, May 2017. [Online; last accessed 22-May-2018].

[4] "Medium." https://medium.com/. [Online; last accessed 22-May-2018].

[5] "Declarative routing for react.js." https://reacttraining.com/react-router/web/api/Route/render-func. [Online; last accessed 22-May-2018].

[6] "Components." https://react-bootstrap.github.io/components/alerts/. [Online; last accessed 22-May-2018].

[7]

[8] "Mongodb driver api." http://mongodb.github.io/node-mongodb-native/3.0/api/. [Online; last accessed 22-May-2018].