Coursework Report

Christopher Johnson
40275286@live.napier.ac.uk
Edinburgh Napier University - Module Title (SET08101)

1 Introduction

The aim of this coursework was to design a simple blog platform using HTML, CSS and Javascript supplemented with additional libraries, templates or frameworks. The blog platform must allow the user to Create a blog entry, Read existing blog entries, Update existing blog entries and Delete existing blog entries using a (CRUD) API.

My blog site consists of a simple new blog page and a view all blogs page with update and delete functions.

I did lots of online reading for this project, using documentation to aid in debugging. I also read Beginning Node.js written by Basarat Ali Syed as suggested on the module page.

2 Software Design

I used Express as the framework for my Node.js web application. Express offers a choice of templating engines. I chose to use the Handlebar templating system. This decision was made because it is capable of parsing HTML without converting to another language. With this method I can use Bootstrap templates without needing to convert them, simplifying templates.

SCSS was enabled as this allowed me to locally host and compile Bootstrap and FontAwesome frameworks. I used Bootstrap because it can be used to easily create a website that responds to the device it's viewed on and is compatible with all browsers. FontAwesome was used as I wanted good looking icons on my blog update and blog delete buttons.

For data storage I used MongoDB. I was able to easily create database using mlab.com. The database has three collections:

- pages: for storing the contents of each page. When a page is rendered the page data is read by the app and the page content is displayed by the template.

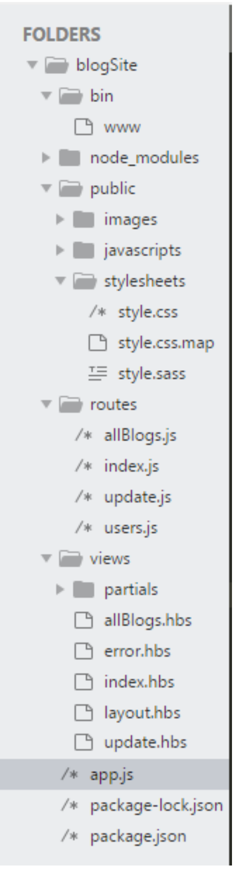
- blog-entries: for storing the contents of each blog entry.

- site-settings: for storing the site settings which are used by the app to allow the template to access the saved site settings.

Creating the pages this way means there is the potential to add new pages in the future by adding an entry to the database. The navbar element of the template reads these page values and can auto update to accommodate new pages.

The Javascript files for Bootstrap, FontAwesome, jquery, MongoDB and popper.js were included and have not been altered are unlikely to ever be.

This is was my file structure after using these tools:

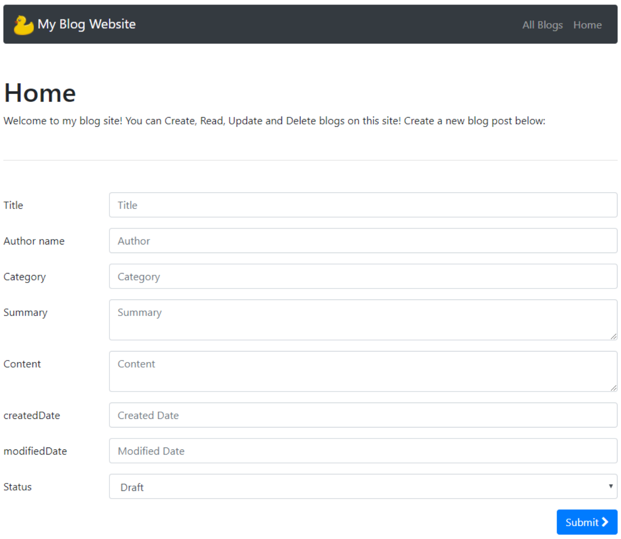


```
FOLDERS
▾ blogSite
  ▾ bin
      www
  ▸ node_modules
  ▾ public
    ▸ images
    ▸ javascripts
    ▾ stylesheets
        style.css
        style.css.map
        style.sass
  ▾ routes
      allBlogs.js
      index.js
      update.js
      users.js
  ▾ views
    ▸ partials
      allBlogs.hbs
      error.hbs
      index.hbs
      layout.hbs
      update.hbs
    app.js
    package-lock.json
    package.json
```

3 Implementation

My site has two main pages, the new blog/home page and the all blogs page. Other pages can be accessed by performing certain actions (e.g. updating blog entry).

When a user visits the site they directed to the Home/New Blog page:

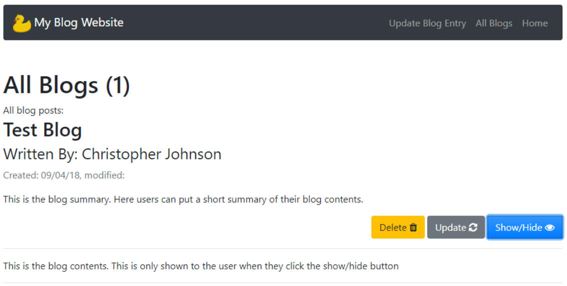


Home

Welcome to my blog site! You can Create, Read, Update and Delete blogs on this site! Create a new blog post below:

Title	Title
Author name	Author
Category	Category
Summary	Summary
Content	Content
createdDate	Created Date
modifiedDate	Modified Date
Status	Draft

Here a user can create new blog by filling in the form. And clicking submit. This will cause the app to read the form entries and save them as an item in the 'blog-entries' collection in the Database. This will cause the blog to show up on the all blogs page:



All Blogs (1)

All blog posts:

Test Blog

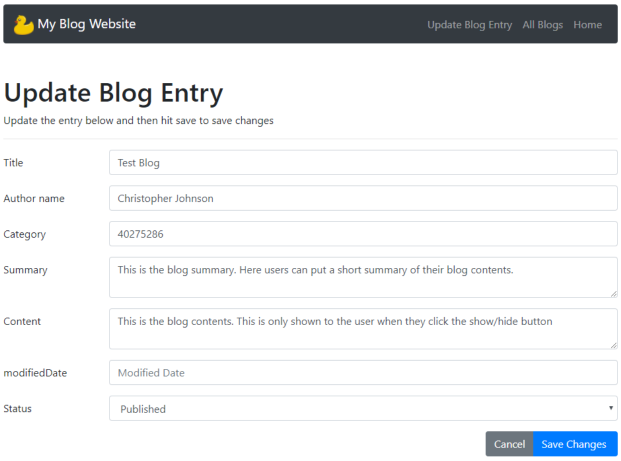
Written By: Christopher Johnson

Created: 09/04/18, modified:

This is the blog summary. Here users can put a short summary of their blog contents.

This is the blog contents. This is only shown to the user when they click the show/hide button

From here users can view all the blog entries. The summary is only displayed unless the user clicks the show/hide button which displays the blogs content. If the user clicks update they are redirected to this page:



Update Blog Entry

Update the entry below and then hit save to save changes

Title	Test Blog
Author name	Christopher Johnson
Category	40275286
Summary	This is the blog summary. Here users can put a short summary of their blog contents.
Content	This is the blog contents. This is only shown to the user when they click the show/hide button
modifiedDate	Modified Date
Status	Published

If the user clicks cancel they are redirected to the all blogs page. Clicking save changes removes the current displayed entry and then saves a new entry with the same id by reading the forms on the page. The user is then redirected to the all blogs page and the blog will be updated.

4 Critical Evaluation

The aim of the coursework was to design a simple blog platform that allows a user to create, update, read and delete blogs. Using my site a user can do all of these things however there are many things I would improve if I were designing a blog platform again.

My site supports the creating, updating, reading and deleting of blog entries. The pages respond to these changes, updating so that the most recent information is always displayed.

My site does not support different users or provide any protection against people deleting, updating or creating blog entries. This is something I would have liked to implement but did not have the time for unfortunately.

5 Personal Evaluation

This was definitely the most challenging coursework I have completed during my course at Napier. However, Web Design is one of my favourite areas of computing and so although challenging the work was interesting.

Whilst working on this coursework I learned a lot about using Node.js to create a functioning site. I especially improved my knowledge on how templates work, through the use of handlebars.

6 References

Node Modules Used (Licenses are provided in git repo):

- bootstrap : https://getbootstrap.com/
- mongoDB : https://www.mongodb.com/
- FontAwesome : https://fontawesome.com/
- popper.js : https://popper.js.org/