==

Group Meetings

==

On average, our group met 3 hours / week totaling in 34 hours of meeting time. During these times we discussed issues we were having, such as Unit Testing or little bugs in the project.

==

User Experience

==

User can have a role out of 6 roles:

- Private User
- Private User No Comment
- Sensitive User
- Sensitive User No Comment
- Admin
- Creator

The roles have permissions assigned to them, they are:

- Can Comment
- Can Create Blogs
- Can View Private Blogs
- Can View Sensitive Blogs
- Can Delete Blogs
- Can Edit Users
- Can Edit Blogs

A Private User has the following permissions:

- Can Comment
- Can View Private Blogs

A Private User No Comment has the following permissions:

- Can View Private Blogs

A Sensitive User has the following permissions:

- Can Comment
- Can View Private Blogs
- Can View Sensitive Blogs

A Sensitive User No Comment has the following permissions:

- Can View Private Blogs
- Can View Sensitive Blogs

An Admin has the following permissions:

- Can Comment

- Can Create Blogs
- Can View Private Blogs
- Can View Sensitive Blogs
- Can Delete Blogs
- Can Edit Users
- Can Edit Blogs

A Creator has the following permissions:

- Can Comment
- Can Create Blogs
- Can View Private Blogs
- Can View Sensitive Blogs

The blog has a main page where a basic user is able to:

- View all public blogs
- Click on a blog and view the full blog page.
- The user can see the comments but cannot comment.
- The user can click login and enter username and password or register for a new account.

When the user logs in and enters the wrong username or password the page will be reloaded and they will need to enter the correct username and password.

When the user registers and tries to create a username that already exists, the page will reload and they will need to enter the information again.

If the user successfully registers, they will be taken to the main page.

Initially when the user registers they will only be able to read blogs and comment.

An admin will have to go into the admin panel and edit the permissions of the users.

If the user has post blog permissions they will be able to click on the Post Blog button and it'll take them to a Create Blog page where they will add a title, select the tags and permissions, and write the text for their blog and submit.

If the user has Edit User permission they will be able to click on the Admin button, which will allow them to edit any user that is signed up for the account and change the role that they are assigned.

A navigation bar is present for easy navigation through the website. Depending on the permissions of the user the navigation bar will have some links displayed and some not.

How to build and run

- Pull the project from Github
- Navigate to the root directory in the command line or terminal
- Run the command “docker compose-up” (make sure there aren’t any other docker images up or this might not work)

Security

The Spring Security restricts where a user can access on the website due to the permissions the user is given. It keeps track of the user and their permissions by creating a JWT token. This token provides the application with the users information when trying to access certain pages, and if the user doesn’t have the required permission, they will be redirected to a different page.

Common Problems

None, our application is perfect. We don’t make mistakes.

Our application runs into a slight problem when creating a blog that has the exact same title and content. This is due to needing to run a sql statement to get the id (we have sql auto incrementing the id’s), and the only information we have to compare is the title and content . It will return 2 blogs instead of the blog that we are intending to receive.

The logout button doesn’t work from every page. The only page it will actually log the user out is the main page. If the user is reading or creating a blog, the logout button will not work. We could not figure out what was causing this to happen.

REST API

Viewing Blog Post by ID:

getBlogById?blogId=ID

ID: can be any integer that has a blog post tied to it

Deleting Blog Post by ID:

deleteBlogById?blogId=ID

ID: can be any integer that has a blog post tied to it

Creating Blog Post:

createBlog?blogTitle=TITLE&blogBody=BODY&AuthorID=AUTHOR&tag=TAG&permission=PERMISSION

TITLE: Name of the blog

BODY: The content of the blog post

AUTHOR: The id of the user the post will be created by

TAG: The tag, if any, that the post will be associated with.

Possible tags include: Sports, Technology, Politics, Music and Finance

Can be left empty, if so it will look like this “tag=&permission”

Permission: The permission, if any, that the post requires to view

Possible permissions: Sensitive and Private

Can be left empty, if so it will look like this “permission=”

Update Blog Post:

updateBlog?blogId=BLOGID&blogTitle=TITLE&blogBody=BODY&AuthorID=AUTHORID

BLOGID: id of the blog that is going to be changed

TITLE: New Title of the blog

BODY= New content of the blog

AUTHORID: The id of the user the post will be created by