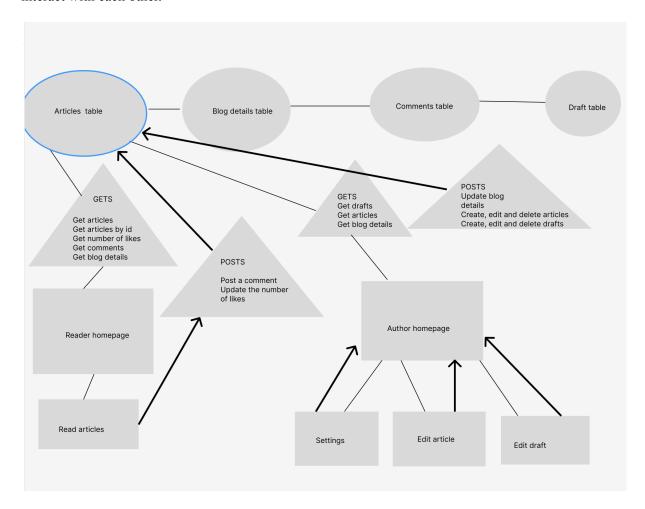
Schematic Diagram:

The image below provides a high level diagram of how the separate components in the application interact with each other.



Extension - Front End Styling and GUI:



The extension for this midterm focuses on front end styling and GUI - in particular creating a CSS file from scratch. The reason for creating the CSS from scratch – using sources such as <u>W3Schools</u> - was motivated by a desire to be creative without any reliance on CSS frameworks such as <u>Bootstrap</u>.

The CSS used for the web application can be located in the *public* folder under the subdirectory *css/styles.css*. The stylesheet uses a *div* container to structure how elements are laid out across the page (60-67, styles.css). The layout also uses muted colours such as blue and purple, so as not to be obtrusive. <u>ColorHunt</u> proved to be helpful when it came to selecting the colour palettes used throughout the web application. Furthermore, the logo image is reflected (50-51, styles.css). It is also used across all the pages – just like the footer.



Moreover, each *body* in one of the views contains its own background colour to ensure that the website is unique and aesthetically pleasing (8-36, styles.css). The buttons are gradient with the styling for these having been sourced from <u>CSS Gradient Buttons</u> with some minor modifications made (69 - 114, styles.css).

In order to make it easier for the user to locate the reader or author main pages, each of these homepages links to its respective counterpart using a top navigation bar. The code for the top navigation bar was sourced from <u>W3Schools</u>; however, modifications were made to its appearance, especially in terms of its colours (158 - 190, style.css). On the author's homepage, the drafts and articles lists use rows and columns in order to display both of these side by side (133-144, style.css).



An important aspect of the front-end design, especially when considering the numerous forms for this web application, was validating the input. Not only was it important to capture any user input that was missing, but also to ensure that date inputs were at least valid.

The *validation-script.js* located in the *public* folder under the subdirectory *scripts* handles the various validation inputs for all the forms. In order to not be repetitive, helper functions such as *validateDate* (as shown in the image below) are used (57-121, validation-script.js).



As mentioned before, the validation is applied to all of the forms used throughout the application – this helps, in some ways, to make sure that rubbish input is not entered into the application by the user.