# COMP2011

NUR MOHAMAD AZHAR

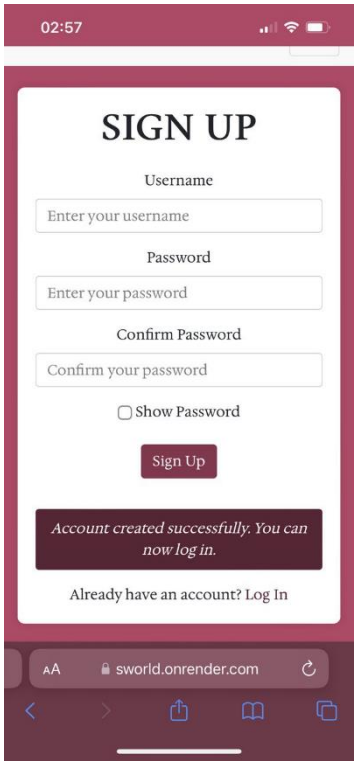
## My Website

The website that has been developed is a social networking platform named Sworld. This website enables users to generate postings and interact with other users via actions such as liking their posts and choosing to follow or unfollow them.

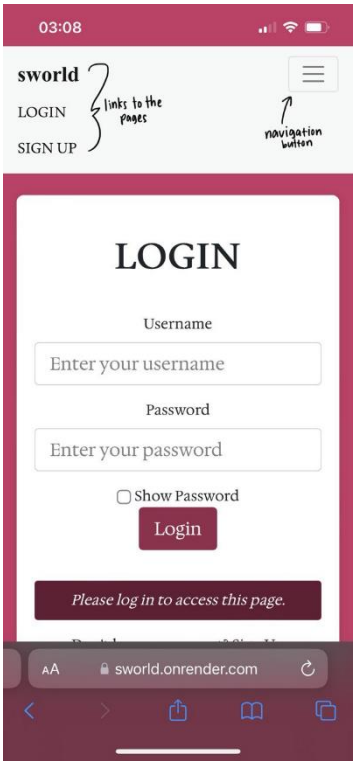
### User Path

The steps on how a user would use Sworld are as follows (assuming that this is the user's first time using Sworld):

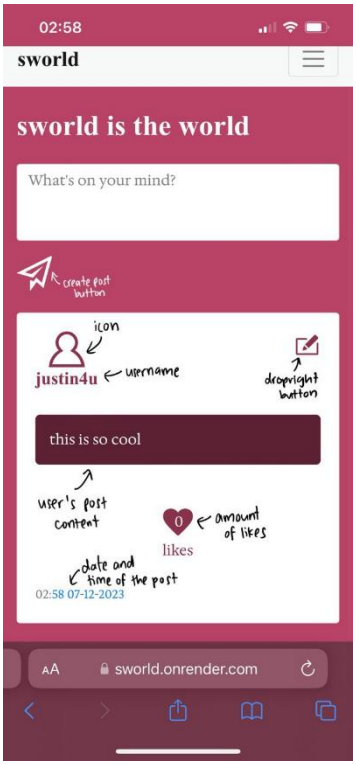
1. User creates an account on the sign-up page.
2. User will log in to their new account via the login page.
3. Sworld redirects user to the homepage, where they can create a post, see their posts, and see the posts of the people they follow.
4. User can view their profile page by clicking the navigation button or their username.
5. If the user wants to search for another user's profile, they can go to the explore page.
6. User can go to the settings page to change their password and font size.



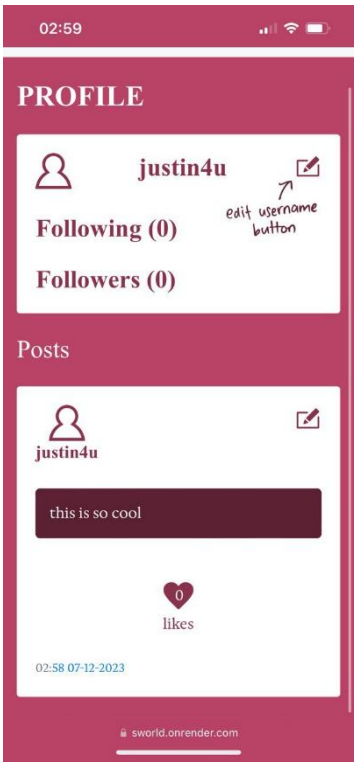

*Step 1*




*Step 2*

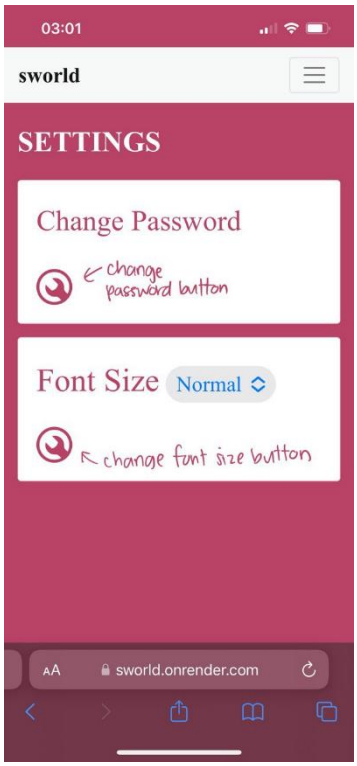



*Step 3*



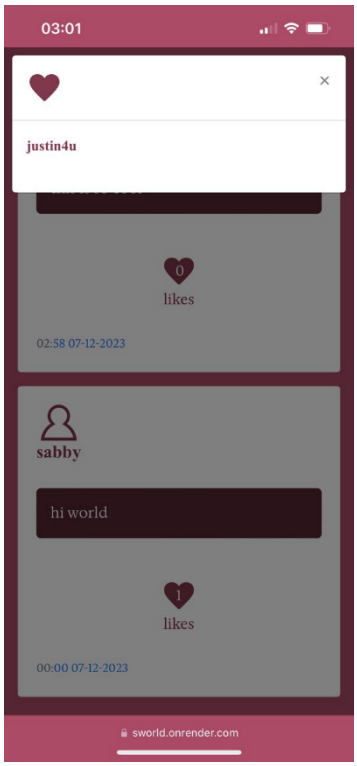

*Step 4*


*Step 5*




*Step 6*

# Design



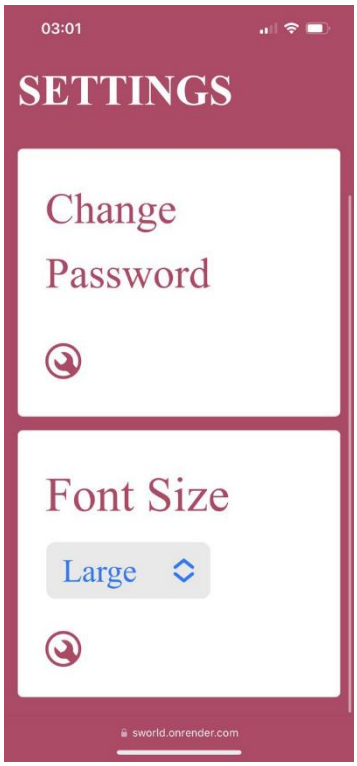

The website has a pink colour scheme, with white used for text and background components, providing contrast and readability. The typeface has a refined and graceful aesthetic, characterised by uniform forms and sizes. The navigation bar is meticulously arranged, with buttons for home, profile, explore, settings, login, and register.

The posts are in a card layout with integrated buttons for liking and disliking. Icons used to communicate activities such as confirming changes, following, posting, and exploring instead of using regular buttons. The website is responsive that adapts to various screen sizes and devices. The forms and user interaction characterised by user-friendliness, enhanced by interactive features such as buttons and dropdowns, thus intensifying user experience. Flash messages used to provide alerts and offer feedback about activities such as publishing a post.

The challenges encountered in creating the website were implementing the "like" button, management of the "follow" button, distinct and easily identifiable icons for various activities, and modals to provide further information without overcrowding the primary content section.

The screenshot above shows the modal used to provide supplementary information which showcases the users who have expressed their liking for a post, while ensuring that the primary content area remains free from clutter. This aspect of the design is one of the components that turned out well.

## Accessibility




*In this screenshot, the font size is large which fills the whole page.*

The accessibility of Sworld varies across individuals depending on the application of the font size function on the settings page. The form provides customers with a selection menu that includes choices for small, regular, and large font sizes.

The user's current font size preference dynamically adjusts the chosen attribute for the relevant option, ensuring that the dropdown accurately represents the user's decision. The form submission initiated using a submit button, enabling smooth changes to the user's font size option on the server side.

The combination of jQuery and Bootstrap libraries improves the visual attractiveness and interactivity of the font size selection, offering consumers a user-friendly and visually appealing way to personalise their viewing experience. In general, the implementation showcases a design that prioritises the needs and preferences of the user, resulting in a seamless and captivating experience while interacting with the website's settings.

## Database

The models.py file incorporates a many-to-many connection for followers and a similar association for post likes in order to provide a comprehensive user experience. These connections enabled by two intermediary tables: followers and post_followers. The followers table facilitates a many-to-many relationship between users, allowing each user to have numerous followers and follow multiple users. The link between followers and following established using the followers property in the User class, facilitating convenient navigation between the two.

```
followers = db.Table('followers',
    db.Column('follower_id', db.Integer, db.ForeignKey(id_user)),
    db.Column('followed_id', db.Integer, db.ForeignKey(id_user))
)

# Define User Model
class User(db.Model, UserMixin):

    # other existing code…

    # Define followers relationship
    followers = db.relationship('User', secondary='followers',
                                primaryjoin=(followers.c.follower_id == id),
                                secondaryjoin=(followers.c.followed_id == id),
                                backref=db.backref(
                                    'following', lazy='dynamic'),
                                    lazy='dynamic')
```

The post_followers database creates a many-to-many link between users and posts, allowing individuals to express their likes for numerous posts and enabling posts to get likes from multiple users. The Like model establishes a connection between users and content through the Like table.

```
# Define post_followers table
post_followers = db.Table('post_followers',
    db.Column('user_id', db.Integer, db.ForeignKey(id_user)),
    db.Column('post_id', db.Integer, db.ForeignKey('post.id'))
)

# Define Like Model
class Like(db.Model):

    user = db.relationship('User', backref=db.backref('likes', lazy=True))
    post = db.relationship('Post', backref=db.backref('likes', lazy=True))
```

The database architecture improves the user experience by allowing features such as user following and liking. Users may establish connections with one another, promoting a feeling of community and active participation. The use of many-to-many connections offers a versatile and scalable framework capable of supporting future enhancements while preserving the integrity of user interactions. In addition, backrefs streamlines query navigation, guaranteeing smooth access to associated data, which is essential for quickly presenting lists of followers and liked articles.

# Advanced Feature

The implementation of an AJAX like button is an advanced feature that significantly enhances the website's user experience by providing real-time feedback and interactivity without requiring a page refresh. This approach improves the overall responsiveness and perceived speed of the application. Below is the code snippet of how the like button is implemented using Flask, jQuery, and the existing database models:

```
$(document).ready(function() {
    // Handle like/unlike actions
    $(".like-btn").click(function(e) {
        e.preventDefault();

        var post_id = $(this).data("post-id");
        var is_liked = $(this).hasClass("btn-primary");

        // Send AJAX request to like/unlike post
        $.ajax({
            type: "POST",
            url: "/like_post/" + post_id,
            success: function(data) {
                // Update the like button and like count
                if (is_liked) {
                    $(`.like-btn[data-post-id=${post_id}]`).removeClass("btn-
primary").addClass("btn-secondary");
                } else {
                    $(`.like-btn[data-post-id=${post_id}]`).removeClass("btn-
secondary").addClass("btn-primary");
                }

                // Update the like count
                var likesCount = data.likes_count || 0;
                $(`.like-count[data-post-id=${post_id}]`).text("Likes: " +
likesCount);
            },
            error: function(error) {
                console.log("Error:", error);
            }
        });
    });
});
```


Upon activating this configuration, when a user selects the like button, an AJAX request is sent to the server, targeting the /like_post route. The server handles the request, modifies the database appropriately, and returns a response to the client. Subsequently, JavaScript revise the user interface (UI) to accurately represent the action, all while avoiding the need to refresh the whole website.

This like button improves the website by offering a smooth and quick-to-like experience that adapts well to user actions. Users may promptly express their admiration for content without disruptions, resulting in a more captivating and lively user experience. Furthermore, it minimises the strain on the server and conserves bandwidth by just refreshing the essential components of the webpage, enhancing the overall user experience.

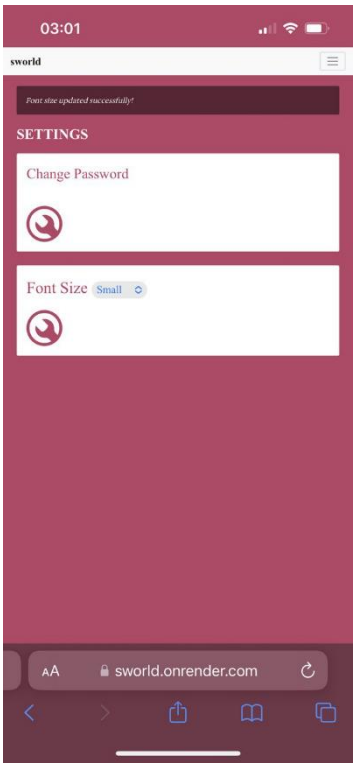
# Testing & Critical Analysis

## Testing

In each of the sections below, talk about how you tested your code – you can use screenshots or code snippets.

### *Accessibility*

The website offers a user to choose their preferred individual preferences and feature is for users with who prefer larger text. The readability and impaired users, aligning with The font size button also needs, ensuring a more for a broader audience.




experience by allowing users font size, catering to accessibility needs. This visual impairments or those font size option enhances comprehension for visually web accessibility standards. addresses diverse user inclusive and usable website

### *Functionality*

In the Flask application with SQLAlchemy models, several testing approaches were adopted, such as unit testing. Below are some part of the code used in testing the website:

```
class ModelsTestCase(unittest.TestCase):

    def setUp(self):
        self.app_context = self.app.app_context()
        self.app_context.push()
        db.create_all()

    def tearDown(self):
        db.session.remove()
        db.drop_all()
        self.app_context.pop()
```

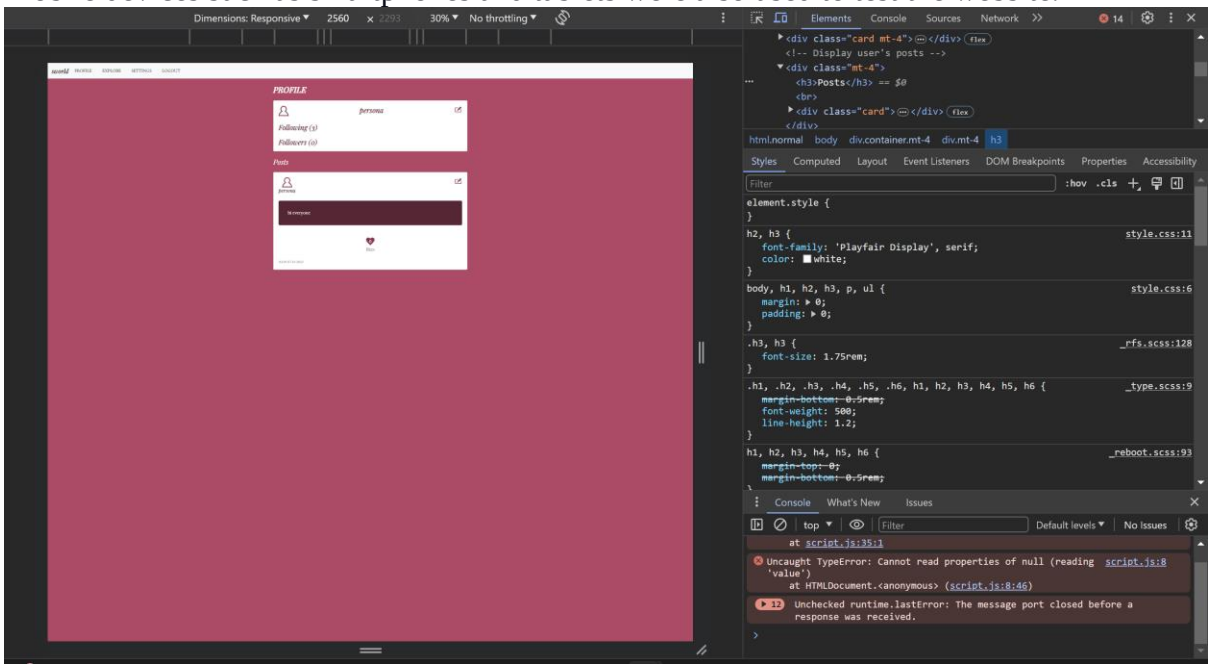
## *Layout & responsiveness*

Ensuring that a web application layout works well on different devices is typically achieved through responsive design. Responsive design aims to provide an optimal viewing experience across a range of devices, from desktop monitors to smartphones. The use of Bootstrap is evident, which is a popular front-end framework known for its responsive design features. Here are a few points that were considered in the code that contribute to responsive design:

1. **Viewport Meta Tag:** The absence of the viewport meta tag in the HTML head might affect responsiveness. Ensuring that the following meta tag is present in the <head> section is crucial.

```
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">
```

2. **Testing on Different Devices:** The browser developer tools such as Inspect used to stimulate different screen sizes during development that provide virtual device testing. Mobile devices such as smartphones and tablets were also used to test the website.



# Analysis

For each section below, try and come up with at least 2 things you can talk about. You should avoid discussing things outside of your control ('I did not have time to…', 'I could not work out how to…') and instead focus on specific issues you had. Again, you can include screenshots of your website or snippets of your code if you want to.

## *What Went Well*

The parts of the website that were done neatly are the modal likes and the follow/unfollow button. The modal likes feature likely involves displaying a modal (a pop-up dialog) when a user interacts with the like button on a post. This approach allows users to express their liking for a post without navigating away from the current page. The modal also includes details about users who have liked the post, potentially provide options for the current user to "unlike" the post. This feature adds a layer of interactivity to the liking functionality, making it more engaging and user-friendly. It prevents the need for a full-page refresh and provides instant feedback to the user about the like action.

The follow/unfollow button is a crucial social interaction element. Users can follow or unfollow other users to stay updated with their activities. The implementation involves dynamically updating the button appearance based on the current relationship status (following or not following). This feature utilizes AJAX (as mentioned earlier) to perform these actions without requiring a full page reload. It enhances the website's responsiveness and provides a seamless social interaction experience.

## *What Went Badly*

One significant challenge encountered was related to the like button not updating in the database initially. This issue attributed to problems with the AJAX request or the logic responsible for handling the like/unlike action on the server. The investigation focused on ensuring the AJAX request sends the required data to the server and confirms that the server processed the like/unlike action accurately, updating the database accordingly. The resolution involved a thorough debugging process, addressing issues identified in the AJAX request and server-side logic.

Another challenge pertained to the page refreshing every time a user liked or disliked a post. The root cause identified as the AJAX request not being handled correctly, triggering a full-page refresh. In addressing this, the solution involved updating the AJAX request to prevent the default form submission behaviour, allowing the like/unlike action to occur asynchronously. Additionally, client-side JavaScript implemented to dynamically update the like count and button appearance without requiring a full-page refresh, enhancing the user experience.

In terms of redirection, a challenge emerged when users clicked on their own username, leading to redirection to the other user profile page rather than their own profile. The underlying issue traced to a conditional statement not implemented correctly. The solution included updating the conditional statement to redirect to the user's own profile page when the clicked username matched the current user's

username. This adjustment ensured a more intuitive and user-friendly navigation experience for individuals interacting with their own profiles.

### *Improvements*

One key enhancement for the website project would be to enable users to upload pictures within their posts. This feature can significantly enrich the user experience, allowing individuals to share visual content with text. This addition could elevate the platform by providing users a more diverse and engaging means of expression.

To further personalize user profiles, allowing users to upload a profile picture would be a valuable addition. A profile picture serves as a visual representation of the user and fosters a sense of identity within the community. Implementing this feature would involve an image upload mechanism specific to user profiles, ensuring proper validation and handling of profile pictures. This enhancement adds a personal touch to user profiles and contributes to a more visually appealing and connected social environment.

Enhancing user experience includes providing options for personalization. Integrating a toggle between light and dark mode is a feature that caters to users' preferences and improves accessibility, especially in varying lighting conditions. Implementing this toggle involves creating alternative colour schemes for the website and incorporating a user-friendly mechanism for users to switch between modes seamlessly. This addition enhances the website's versatility, making it adaptable to users' preferences and promoting a comfortable browsing experience.

Another notable improvement would involve revisiting the website's layout to make it more user-friendly. It could encompass optimizing the navigation flow, ensuring intuitive placement of features, and refining the overall visual hierarchy. Consideration for responsive design, accommodating various screen sizes and devices, is also crucial. A user-friendly layout fosters ease of use, enhances accessibility, and contributes to a positive user experience, ultimately making the website more welcoming and engaging.

# Bibliography

References:


1. iconmonstr. 2023. iconmonstr - Free simple icons for your next project. [Online]. [Accessed 27 November 2023]. Available from: https://iconmonstr.com/
2. Freepik Company S.L. 2010-2023. Vector Icons and Stickers - PNG, SVG, EPS, PSD and CSS. [Online]. [Accessed 28 November 2023]. Available from: https://www.flaticon.com/