1DA1611:A
Advanced Internet Programming

Project nr. 7 Report: 2023-04-06
Illia Priadko (309062)

Plan:
- Familiarize oneself with jQuery;
- Assess which part of Project 5 can benefit from jQuery;
- Implement animation for more sensible UX
- Test and troubleshoot

Given task:

Tasks

Please use the following jQuery features to make your main website clearer when it comes to linking to content

(jQuery Examples are given in this folder)

1. Selector manipulation.
2. Event handling (on mouse over/out, focus/blur etc.) - for example (a description of what you get when clicking the link)
3. Hide/Fade some text/menu options elements
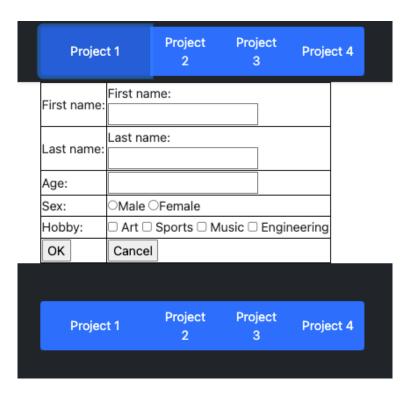4. Slide effect,
5. Animation effects

1. Brief jQuery analysis:

jQuery can be a powerful tool for simplifying and speeding up web development, particularly for tasks that involve manipulating HTML and interacting with server-side data. It is essentially a Javascript library that provides a concise and easy-to-use API which abstracts away many low-level details of JS, making it overall easier to do many tasks incl. animation, HTML document manipulation, Ajax.
Given examples, as well as tutorials readily available online, have been quite insightful about the use cases and syntax of jQuery.

2. Assess which part of Project 5 can benefit from jQuery

As the page is pretty much done and fully functional, a good way I could utilize jQuery, is to improve UX (User experience). Using built-in animation utilities, I had decided to use them to highlight which tab of the page is selected.

On the screenshot it is visible that Project 1 is currently selected, as its button is widened.

The task ended up being a bit complicated due to the fact the animation should be duplicated on both the header and footer of the page, which concluded that $(this) would not be enough.

Here is a code snipped from the <script> section of the main page, please refer to comments as explanation:

```
<script>
    // make sure page loads fully first
    $(document).ready(function () {
        // define identifier for clicked button
        var $clickedButton = null;
        // select all .my-btn class elements and define click event
        $('.my-btn').click(function () {
            // index which button is pressed to later refer to in accessing the other set
(header/footer)
            var index = $(this).index('.my-btn');
            var $buttonFooter = $('.my-btn-footer').eq(index);
            // if clicked button not used, retract to default width
            if ($clickedButton != null) {
                $clickedButton.removeClass('clicked');
                $clickedButton.animate({
                    width: '-=50px'
                }, 100);
                var indexClicked = $clickedButton.index('.my-btn');
                $('.my-btn-footer').eq(indexClicked).removeClass('clicked');
                $('.my-btn-footer').eq(indexClicked).animate({
                    width: '-=50px'
                }, 100);
            }
            // if clicked button is used, widen by 50 pixels
            if (!$(this).hasClass('clicked')) {
                $(this).addClass('clicked');
                $(this).animate({
```

```
                    width: '+=50px'
                }, 100);
                $buttonFooter.addClass('clicked');
                $buttonFooter.animate({
                    width: '+=50px'
                }, 100);
                $clickedButton = $(this);
            } else {
                $clickedButton = null;
            }
        });
    });
</script>
```

Troubleshooting the code involved fixing references to make sure both sections are synchronized, as well as checking how it would work together with the already existing implementation of Ajax on said page.

The final page is available at http://10.44.99.99:23880/~309062/L7/

Conclusions:
After having completed this project, I became familiar with jQuery and its use cases, which allow for more versatile and easy HTML manipulation from inside of JS, abstracting the programmer from lower-level details, as an easy to install library.