1DA1611:A
Advanced Internet Programming

Project nr. 8 Report: 2023-04-13

Illia Priadko (309062)

Plan:

- Analyse given materials;
- Implement populating data into the header of the page;
- Implement filtering by date;
- Testing and troubleshooting

Given task:

4. Creating a menu as nested unordered lists from JSON data - **Priadko** Illia
http://codereview.stackexchange.com/questions/88433/creating-a-menu-as-nested-unordered-lists-from-json-data
Datapicker - **Priadko** Illia
https://jqueryui.com/datepicker/

1. Material analysis

I have decided to take the previous projects as a base to build project 8 upon. In that webpage. The given task involved implementing json to be used as input for the menu population script, so in my case it was a matter of adapting it to make buttons instead of lists. The script will automatically fetch key:value pairs from said json and make a menu with that.

As for the datepicker, it required some changes in the page, namely now each project has its own date attribute. It was decided that the datepicker would be used for filtering projects based on its respective date.

2. Menu creation

As having a paginator at both sides is now not a priority, I decided to get rid of the footer buttons, as the subpages would not have to be scrolled that far to the bottom to make its absence an inconvenience. In turn, it makes content more readable and the page look cleaner.

Instead of buttons defined in an HTML, it was decided that there would only be a placeholder element that will be populated with fetched buttons.

Another element in the header is the date picker, which will be used later.

```
    <header>
        <div class="navbar navbar-dark bg-dark shadow-sm">
            <div class="container">
                <div class="btn-group" role="group" id="btn-group-header"></div>
```

```
                <p style="color: white">Date: <input type="text" id="datepicker"></p>
            </div>
        </div>
    </header>
```

The script is as follows:

```
        $(document).ready(function () {
            // Get the button group element
            function populatemenu(getid) {
                var btnGroup = document.getElementById(getid);

                // Loop through the buttons data and create the HTML for each button
                buttons.forEach(function (button) {
                    var btn = document.createElement("button");
                    btn.type = "button";
                    btn.classList.add("btn", "btn-primary", "my-btn", "my-btn-
header");
                    btn.textContent = button.label + ': ' + button.date;
                    btn.onclick = function () {
                        loadDoc(button.id);
                    };
                    btnGroup.appendChild(btn);
                });
            }
            // iterate through all button groups to call populatemenu(id) function
            $('.btn-group').each(function () {
                populatemenu(this.id)
            });
```

The “button” var here is actually the JSON that is declared beforehand:

```
        var buttons = [
            { id: 1, label: "Project 1", date: "02/23/2023" },
            { id: 2, label: "Project 2", date: "03/02/2023" },
            { id: 3, label: "Project 3", date: "03/09/2023" },
            { id: 4, label: "Project 4", date: "03/16/2023" }
        ];
```

Essentially, this script operates with classes and ids of elements to assign and read properties respectively.

Please note the presence of populatemenu(id) function being called for each button group. This means that the footer menu can actually be brought back quite easily, as long as it maintains the class “.btn-group”.

3. Filtering by date

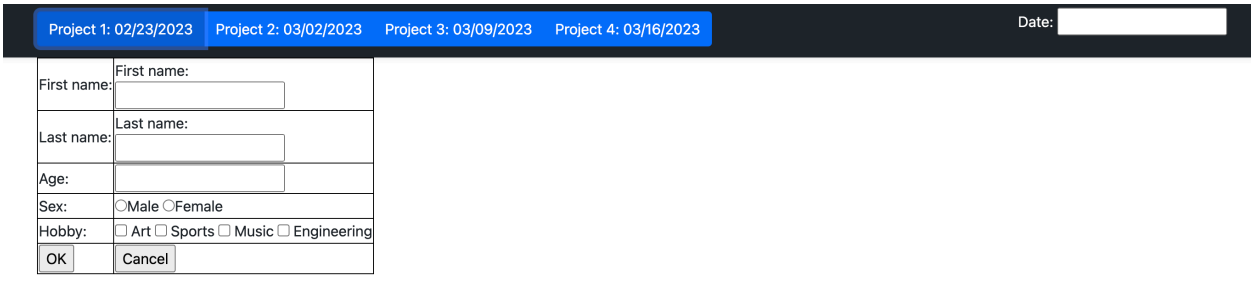
As it was also required to have a jQuery datepicker on the page, it was used to implement a filter by date. Now, each button has a date appended to it, which is an attribute upon which the string of the button title is split and read to match the input date. Alternatively, one may also use the input JSON file to filter through it.

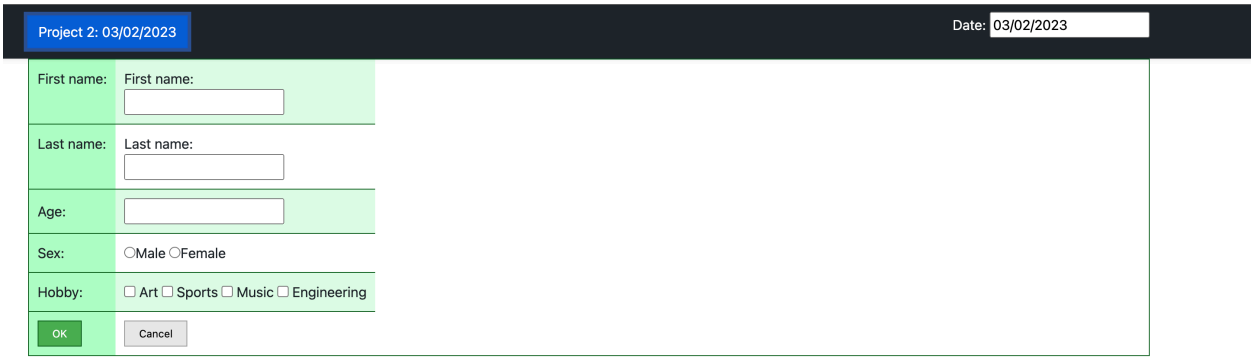
```
            $("#datepicker").datepicker({
                onSelect: function (dateText, inst) {
                    $(".my-btn-header").each(function () {
                        var buttonDate = $(this).text().split(": ")[1];
                        if (buttonDate === dateText) {
                            $(this).show();
                        } else {
                            $(this).hide();
                        }
                    });
                }
            });
```

4. Testing and troubleshooting

The page was tested to work on github pages, as well as the WE UNIX server.

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





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. I was also introduced to how JSON is used to handle and pass data through to the script, as well as implement date filtering using some string manipulation.