Todo List Project making

Reference for the project :
https://github.com/coding-blocks-archives/Project-TodoList-Bootstrap-jQuery

(i) Building the basic HTML layout :

- First step is to include the bootstrap libraries in the index.html file by copying the CDN path as follows :

```
    <!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIa
```

- Next we try including the input text box for entering the task name by going into the forms section and picking up the Overview part of code as. The code has been included in a separate row as follows :

```
<div class="container">
    <h1 align="center">Todo List</h1>

    <!--Input box-->
    <div class="row">
        <div class="col">
         <div class="mb-3">
            <label for="newTask" class="form-label">New Task</label>
            <input type="email" class="form-control" id="newTask" aria-describedby="newTaskHelp" placeholder="Enter the name of the task">
            <div id="newTaskHelp" class="form-text">Short one liner description of the task</div>
          </div>
        </div>
    </div>
```

- Next we try adding the buttons “ADD” and “CLEAR” button and adding it in the next row. Also the use of button classes “btn btn-success” and “btn btn-danger” has been included. Also padding and margin has been included in the class name using “p-2” and “m-2” resp.

```
<div class="row px-4">
    <button class="btn btn-success col m-2">ADD</button>
    <button class="btn btn-danger col m-2">CLEAR</button>
</div>
```

- Next we are including the Task list group below the row of buttons in a separate row as follows :

```
<!-- Task List-->
<div class="row m-2">
 <ul class="list-group">
    <li class="list-group-item">An item</li>
    <li class="list-group-item">A second item</li>
    <li class="list-group-item">A third item</li>
    <li class="list-group-item">A fourth item</li>
    <li class="list-group-item">And a fifth one</li>
  </ul>
</div>
```

(ii) Handle the clicks

- Next we are trying to add functionality into the add button. For the purpose of testing we have console logged the name of the task on to the console window upon clicking it. The code is as follows :

```
let ulTasks = $('#ulTasks')
let btnAdd = $('#btnAdd')
let btnClear = $('#btnClear')
let inpNewTask = $('#inpNewTask')

btnAdd.click(() => {
    console.log(inpNewTask.val())
    inpNewTask.val("")
})
```

(iii) Appending the items in the list

- We are adding the functionality into the ADD button by creating a new item onclick and then adding it to the list group as follows :

```
btnAdd.click(() => {

    //creating a new element
    let listItem = $('<li>', {
        'class' : 'list-group-item',
        text : inpNewTask.val()

    })

    ulTasks.append(listItem)
    console.log(inpNewTask.val())
    inpNewTask.val('')
})
```

- Next we are adding functionality to the clear button by adding the inpNewTask.val(‘’) as follows :

```
btnClear.click(() =>  inpNewTask.val('') )
```

(iv) Toggle done/not-done items

- Now our first task is to strike through the item which we have clicked on and also apply the diabled class on it as follows :
- The disabling feature can be added just by adding disabled besides the class name as follows :

```
<!-- Task List-->
<div class="row m-2" id="ulTasks">
 <ul class="list-group">
    <li class="list-group-item disabled">An item</li>
    <li class="list-group-item">A second item</li>
```

- Now in order to get that functionality of getting the list item striked off upon clicking the following piece of code is displayed as follows :

```
listItem.click((event) =>{
    //console.log('clicked', $(event.target))
    $(event.target).toggleClass('disabled') //toggle with the
 })
```

- Now in order to get the toggle between the done and not done features we need to do the following :

  In the style.css we are defining a new class called “done” as follows :

```
#ulTasks li.done{
    color : #6c757d;
    text-decoration: line-through;
}
```

Also in the script.js instead of toggling between to the disabled class we include the done class as follows :

```
listItem.click((event) =>{
    //console.log('clicked', $(event.target))
    $(event.target).toggleClass('done') //toggle wi
 })

ulTasks.append(listItem)
console.log(inpNewTask.val())
inpNewTask.val('')
```

(v) Handling the “Enter” button

- Here we need to add the key press functionality with the input box ID as follows :

```
//adding the keypress event as follows :
inpNewTask.keypress((e) =>{
    console.log(e.which)
})
```

- On the console the following output is displayed :

```
49
50
13
>
```

Where 13 corresponds to the pressing of “Enter” button.

- Now in order to implement the functionality we relocate/shift the contents mentioned inside into a new function called addItem as follows :

```
//defining a seperate function for adding the item
function addItem(){
    //creating a new element
    let listItem = $('<li>', {
        'class' : 'list-group-item',
        text : inpNewTask.val()

    })

    listItem.click((event) =>{
        //console.log('clicked', $(event.target))
        $(event.target).toggleClass('done') //toggle with the diable
     })

    ulTasks.append(listItem)
    console.log(inpNewTask.val())
    inpNewTask.val('')
}
```

- Next we are calling the function addItem while clicking and also with the condition that the keypress is 13 :

```
//adding the keypress event as follows :
inpNewTask.keypress((e) =>{
    if(e.which == 13){
        addItem();
    }
})

btnAdd.click(addItem) //calling the addItem function on clicking the
```

(vi) Add other buttons and functionalities

- First we are aiming to make the button look more attractive and with adding icons to it. This can be done by including the CDN of font awesome website and the inclusion in the button is done as follows :

```
<!--Buttons-->
<div class="row px-4">
    <button class="btn btn-success col m-2" id="btnAdd">
     <i class="fa fa-plus"></i>
     ADD
     </button>
    <button class="btn btn-danger col m-2" id="btnClear">
     <i class="fas fa-backspace"></i>
     CLEAR</button>
</div>
```

- Next we are going to add more buttons such as CLEANUP and SORT with making it responsive and adding items from the font-awesome website as follows :

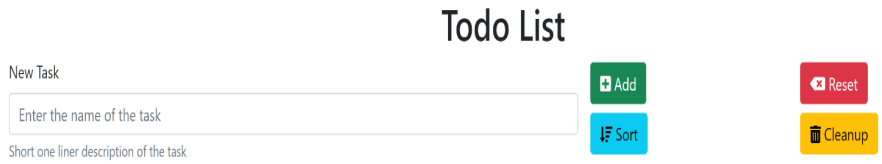
```
<div class="row">
    <!--Input box-->
   <div class="col-md-8">
    <div class="mb-3">
       <label for="inpNewTask" class="form-label">New Task</label>
       <input type="email" class="form-control" id="inpNewTask" aria-describedby="newTaskHelp" placeholder="Enter the name of the task">
       <div id="newTaskHelp" class="form-text">Short one liner description of the task</div>
     </div>
    </div>
     <div class="col-md-4">
       <!--Buttons-->
       <div class="row">
           <button class="btn btn-success col-sm m-2" id="btnAdd">
               <i class="fa fa-plus"></i>
               ADD
               </button>
              <button class="btn btn-danger col-sm m-2" id="btnClear">
               <i class="fas fa-backspace"></i>
               RESET</button>
         </div>
         <div class="row">
           <button class="btn btn-info col-sm m-2" id="btnAdd">
               <i class="fa fa-sort-amount-down"></i>
               SORT
               </button>
              <button class="btn btn-warning col-sm m-2" id="btnClear">
               <i class="fas fa-trash-alt"></i>
               CLEANUP</button>
         </div>


     </div>
   </div>
```

(vii) Adding responsiveness to the page :

- Responsive has been added as follows :

```
<!--Buttons-->
<div class="col-md-5">
  <div class="row">
    <div class="col-6 col-sm-3 col-md-6 p-1">
      <button class="btn btn-success col"  id="btnAdd">
        <i class="fas fa-plus-square"></i> Add
      </button>
    </div>
    <div class="col-6 col-sm-3 col-md-6  p-1">
      <button class="btn btn-danger col"  id="btnReset">
        <i class="fas fa-backspace"></i> Reset
      </button>
    </div>
    <div class="col-6 col-sm-3 col-md-6  p-1">
      <button class="btn btn-info col"  id="btnSort">
        <i class="fas fa-sort-amount-down"></i> Sort
      </button>
    </div>
    <div class="col-6 col-sm-3 col-md-6  p-1">
      <button class="btn btn-warning col" id="btnCleanup">
        <i class="fas fa-trash-alt"></i> Cleanup
      </button>
    </div>
  </div>
```

Output :



(viii) Clear the done tasks

In this section we will include functionality into the cleanup button as follows :

- Define a variable referring to the ID of the cleanup button as follows :

```
let btnCleanup = $('#btnCleanup')
```

- Next we are going to define a function clearDone in order to remove the items that have been striked off/done in the list as follows :

```
function clearDone(){
    //console.log($('#ulTasks.done'))
   $('#ulTasks .done').remove()
}
```

- Next we are going to invoke it on the call of the btnCleanup as follows :

```
//calling the btnCleanup function
btnCleanup.click(clearDone)
```

(ix) Sorting the tasks

The sorting of the task is associated with the SORT button and can be carried out steps as follows :

- Defining the sort button in the script file :

```
let btnSort = $('#btnSort')
```

- Next we define the function sortTasks as follows :

```
function sortTasks() {
    $('#ulTasks .done').appendTo(ulTasks)
  }
```

- Next we call the function as follows :

```
//calling the sort function
btnSort.click(sortTasks)
```

(x) Enabling/Disabling of buttons

Here our aim is to disable the buttons and their functionalities when not needed. For example, the reset button is required only when there is some text entered inside the text box.

To include that functionality following steps were followed :

- Make the toggleResetBtn function as follows :

```
function toggleResetBtn(enabled){
    if(enabled) btnReset.prop('disabled', false)
    else btnReset.prop('disabled', true)
}
```

- The function will be called only when, there is some text entered in the textbox as follows :

```
inpNewTask.on('input', () =>{
    toggleResetBtn(inpNewTask.val() != '')
})
```

- A similar functionality can be extended to the ADD button as follows :

```
function toggleInputButtons(valIsEmpty){
    if(!valIsEmpty){
        btnReset.prop('disabled', false)
        btnAdd.prop('disabled', false)
    }
    else{
        btnReset.prop('disabled', true)
        btnAdd.prop('disabled', true)
    }
}
```

- The function is called only when there is some text entered inside the text box as follows :

```
inpNewTask.on('input', () =>{
    toggleInputButtons(inpNewTask.val() != '')
})
```

- We also need to extend the functionality while clicking the reset button. Means when the reset button is clicked, then the ADD and RESET buttons needs to get diabled as follows :

```
btnReset.click(() => {
    inpNewTask.val('')
    toggleInputButtons(true)
})
```

(xi) Making Dynamic Buttons

- Now since the functionality has to be extended to the sort and cleanup button as well we have made the following changes in the original toggleInputButtons function where earlier a boolean value was passed as an function argument, we have made it empty as follows :

```
function toggleInputButtons(){

    btnReset.prop('disabled', inpNewTask.val() == '')
    btnAdd.prop('disabled', inpNewTask.val() == '')
```

- Next we are going to add the toggle functionality for the setup and sort button. Our aim is to enable the button only when there are more than one items in the task list. This can be found by using :

```
> ulTasks.children().length
< 2
```

- This condition is incorporated in the function as follows :

```
function toggleInputButtons(){

    btnReset.prop('disabled', inpNewTask.val() == '')
    btnAdd.prop('disabled', inpNewTask.val() == '')
    btnSort.prop('disabled', ulTasks.children().length < 1)
    btnCleanup.prop('disabled', ulTasks.children().length < 1)

}
```

- We are algo going to add the toggleInputButtons at the end of the “addItem” and “clearDone” function.

FINAL OUTPUT :