|  |  |  |
| --- | --- | --- |
|  | Pages.: | lecturers: A. Vranken  Course (OPO/OLA):Front-End Development  code OPO/OLA: MBI01J  **The standard duration for this exam is 2 h 20 min, but each student will be given 30% extra time, i.e. a total of 3 hours.** |
| Date: **19/1/2024, 13h**  Group: **BCS** | | Last name:  First name:  student number:  time of submission: ………………………..hours………………….min. |
| **Resources**: own laptop incl. all stored documentation, except sound. | | points      / |

* Connect your laptop to the restricted wifi network 'xtoledo'. The password is 'exam001'.
* No smartphone, nor smartwatch.
* At xtoledo.ucll.be you will find this assignment as a PDF, start code in a Zip file and videos.
* You deliver your HTML, CSS and JS code in one Zip file named "FamilyNameFirstName.zip". Make sure this zip contains only your HTML, CSS and JS, not the movies or the back-end.
* Deposit it by uploading this zip to the xToledo assignment.
* Once everything is uploaded, check again in the assignment to make sure your Zip is there. Download your submission to another folder, unzip, and check that you have submitted the correct one.
* Hand off this assignment bundle and sign the attendance list.

Question 1. Create a form to add ‘to do’ items (5 points)

This question is purely about HTML and CSS. You don't need to use javascript. The form will not work. This question is separate from questions 2, 3 and 4.

**Given**: question1.html, question1.css, video question1.mp4

In 'question-1.html' create a form with the fields

- Name

- Description: short text

- Duration: the number of minutes you think you will need to complete the task

- Deadline: the date by which the task must be completed

All fields are required. Provide HTML5 validation.

Write the necessary CSS (not SCSS) in question-1.css.

Some browsers display the date field differently than you see in the screenshot. You don't need to change that.

The layout of the form depends on the width of the screen. For wide screen, the input fields are organized in two columns:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

For narrow screens, all fields are under each other:



Requirements:

* question-1.html: HTML for the requested form
* question-1.css: CSS (not SCSS) that creates the requested layout

Question 2: Show ‘to do’ items and change status (3 points)

**Given**: question2-3.html, question2-3.js, video question2-3.mp4

In 'question-2-3.js' you will find an array of 'to do' items. A 'to do' item has a name (field name, text), a status (field isDone, boolean), and a duration (field duration, number).

In 'question-2-3.html' you will find an HTML file with the element #todos.

Write the necessary javascript code so that all 'to do' items from the array are displayed on screen as an unordered list (ul), in the element #todosAfbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving The color of a 'to do' item depends on its status.

* Done 'to do' items: green
* Not Done 'to do' items: red

The video "question-2-3.mp4" showed the expected outcome.

Make it possible to change the status of an item by clicking on the item. The color of the item should change simultaneously with the status.

Question 3: Filter ‘to do’ items based on duration (3,5 punten)

Requirements:

* question-2-3.html: shows the requested to-do items as a list and allows for status changes.
* question-2-3.js: contains the necessary javascript code.
* If an item is added tot he list, the rest of the code does not need to be changed

**Werk verder met**: question2-3.html, question2-3.js, video question2-3.mp4

Make it possible to filter the ‘to do’ list by duration.

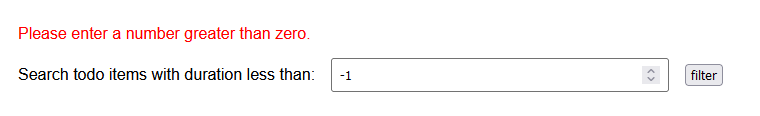
Add the necessary HTML to the 'question-2-3.html' file so that the screen has an input field and corresponding button as shown below, above the #todo-list.

When the user clicks the button, the 'to-do' items on the page are filtered according to their duration. The screen shows only the 'to-do' items that are shorter or as short (≤) as the duration entered by the user. The filtering does not occur until the user clicks the button.

Provide input validation and error message in following cases:

* If the user enters a number less than or equal to 0: "Please enter a number greater than 0".
* If the user enters a number greater than or equal to 100: "The number must be less than 100".

If the user first filters, and then changes the status of a 'to do' item (previous question), the filter must be retained.



Requirements:

* question-2-3.html: Add HTML to show the requested input field and button to filter.
* question-2-3.js: Add javascript code to make the filter form work.

Question 4: Fetch ‘to do’ items from the back-end

This question is completely separate from question 1, 2 and 3.

For this question, you need to start the back-end:

* Download and unzip the back-end from xToledo (file 'todo-backend.zip').
* Navigate to this folder and start the server with npm start (in a terminal, e.g. in VS Code itself). The server started up correctly when you see the following message "Server is running on port 3000".
* Test this by navigating in the browser to <http://localhost:3000/>. If everything is running correctly, you will see in the window the text "To-do back-end app is running...".

You work for this question in the following two files:

* "question-4.html": contains HTML and CSS. You should not edit these files.
* "question-4.js": write the necessary javascript code here.

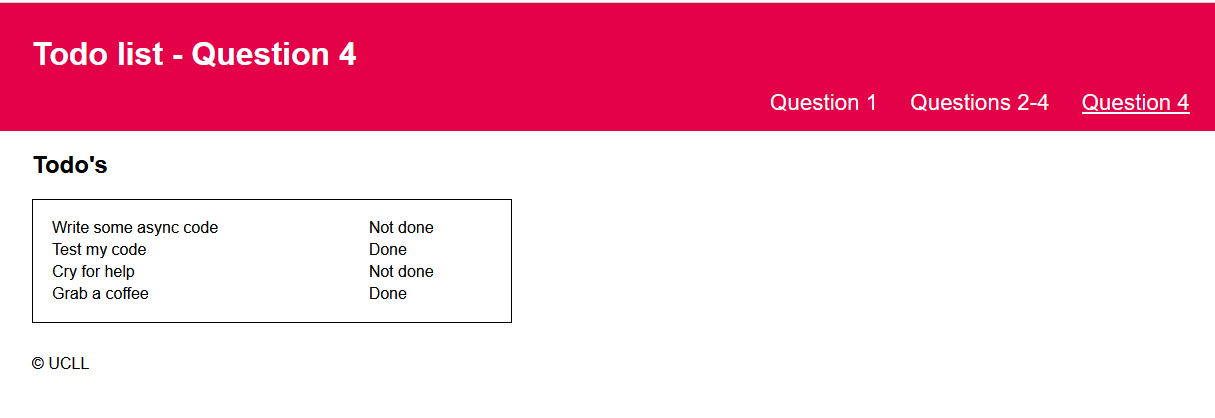
The video "question-4.mp4" shows the desired end result of question 4.1 and 4.2.

Question 4.1: Show all ‘to do’ items (1 points)

The page "question-4.html" already has a table. Write javascript that fetches all to-do items from the back-end. For each item, create a row (tr) with 2 cells (td): name and status. You can see an example in the screenshot.

Use the endpoint http://localhost:3000/todos to fetch all to-do items.

The to-do items will be displayed as soon as the page loads.



Question 4.2: Toggle status (2.5 punten)

Add a third cell with a checkbox in each row: <input type="checkbox" checked>. The attribute "checked" is optional and creates a checkmark in the checkbox.

Make sure that the attribute "checked" matches the isDone status or to-do item: the checkmark is on if the status matches "done."

Make it possible to change the status of a to-do item by enabling or disabling the checkbox: if a "change" event happens on the checkbox, the client sends a POST request to the endpoint http://localhost:3000/todos/toggle/<id> .

Make sure that, after the click, the checkbox checkmark and the status column always display the most current status of the to-do items without reloading the page.

