|  |  |  |
| --- | --- | --- |
|  | aantal pagina’s.: | lectoren: T. Destexhe, G. Jongen, J. Pieck, L. Van Houdt, A. Vranken  opleidingsonderdeel (OPO/OLA):Front-End Development  code OPO/OLA: MBI01h  **De standaardduur voor dit examen is 2 uur 20 min, maar elke student krijgt 30% extra tijd, nl. in totaal 3 uur.** |
| datum: **19/1/2024, 13u**  groep: **1TI** | | naam student:  voornaam:  studentennummer:  tijdstip van indienen: ………………………..uur………………….min. |
| hulpmiddelen: **eigen laptop incl. alle opgeslagen documentatie, m.u.v. geluid.** | | punten      / |

* Verbind je laptop met het beperkt wifinetwerk ‘xtoledo’. Het wachtwoord is ‘examen001’.
* Geen smartphone, noch smartwatch.
* Op xtoledo.ucll.be vind je deze opgave als PDF, startcode in een Zip-bestand en filmpjes.
* Je geeft je HTML-, CSS- en JS-code af in één Zip-bestand met als naam “FamilienaamVoornaam.zip”. Zorg dat deze zip enkel je HTML, CSS en JS bevat, dus niet de filmpjes of de back-end.
* Afgeven doe je door deze zip te uploaden naar de xToledo assignment.
* Als alles geüpload is, kijk je in de assignment nog eens na of jouw Zip er wel degelijk staat. Download je indiening naar een andere folder, unzip, en controleer of je het juiste ingediend hebt.
* Geef deze opgavebundel af en teken de aanwezigheidslijst.

Vraag 1. Maak een formulier om ‘to do’ items toe te voegen (5 punten)

Deze vraag gaat louter over HTML en CSS. Je hoeft geen javascript te gebruiken. Het formulier zal niet werken*. Deze vraag staat volledig los van vraag 2, 3 en 4.*

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

Maak in ‘question-1.html’ een formulier met de velden

* Name (naam)
* Description (omschrijving): korte tekst
* Duration (duur): het aantal minuten dat je denkt nodig te hebben om de taak af te werken
* Deadline: de datum tegen wanneer de taak afgewerkt moet zijn

Alle velden zijn verplicht. Zorg voor HTML5 validatie.

Schrijf in question-1.css de nodige CSS (geen SCSS) zodat het formulier eruit ziet als in het filmpje.

*Sommige browsers tonen het datum-veld anders dan je ziet op de screenshot. Je hoeft dat niet te wijzigen.*

De lay-out van het formulier hangt af van de breedte van het scherm. Voor breed scherm zijn de input fields georganiseerd in twee kolommen:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Voor smal scherm staan alle velden onder mekaar:



Requirements:

* question-1.html: HTML die het gevraagd formulier genereert
* question-1.css: CSS (geen SCSS) die de gevraagde lay-out maakt

Vraag 2: Toon de ‘to do’ items en verander de status (3 points)

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

In ‘question-2-3.js’ vind je een array met ‘to do’ items. Een ‘to do’ item heeft een naam (veld *name*, tekst*)*, een status (veld *isDone,* boolean), en een duur (veld *duration,* number).

In ‘question-2-3.html’ vind je een HTML bestand met het element #todos.

Schrijf de nodige javascript code zodat alle ‘to do’ items uit de array op het scherm getoond worden als ongeordende lijst (ul), in het element #todosAfbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijvingDe kleur van een ‘to do’ item hangt af van de status.

* Done ‘to do’ items: groen
* Not Done ‘to do’ items: rood

De video “question-2-3.mp4” toonte de verwacht uitkomst.

Maak het mogelijk om de status van een item te veranderen door een klik op het item. De kleur van het item moet tegelijkertijd met de status veranderen.

Vraag 3: Filter ‘to do’ items op basis van duur (3,5 punten)

Requirements:

* question-2-3.html: toont de gevraagde to-do items als lijst en maakt het mogelijk om de status te veranderen
* question-2-3.js: bevat de nodige javascript code
* als een item toegevoegd wordt aan de lijst, hoeft de rest van de code niet aangepast te worden

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

Maak het mogelijk om de takenlijst te filteren op basis van duur.

Voeg aan het bestand ‘question-2-3.html’ de nodige HTML toe zodat het scherm een input veld en bijhorende knop heeft zoals in het filmpje getoond wordt, boven de #todo-lijst.

Als de gebruiker op de knop klikt, worden de ‘to-do’ items op de pagina gefiltered naargelang hun duur. Het scherm toont alleen de ‘to do’ items die korter of even kort (≤) duren als de duur die de gebruiker ingegeven heeft. *De filtering gebeurt pas als de gebruiker op de knop klikt.*

Zorg voor input validatie en foutboodschap in volgende gevallen:

* Als de gebruiker een getal kleiner dan of gelijk aan 0 ingeeft: “Please enter a number larger than zero.”
* Als de gebruiker een getal groter dan of gelijk aan 100 ingeeft: “The number must be less than 100.”

Als de gebruiker eerst filtert, en daarna de status van een ‘to do’ item wijzigt (vorige vraag), moet de filter behouden blijven.

Bekijk ook de video voor het verwachte eindresultaat.

Requirements:

* question-2-3.html: voeg HTML toe zodat het gevraagde input veld en de knop “filter” getoond worden
* question-2-3.js: voeg javascript code toe zodat de knop “filter” werkt

Afbeelding met tekst, schermopname, lijn, Lettertype

Automatisch gegenereerde beschrijving

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

*Deze vraag staat volledig los van vraag 1, 2 en 3.*

Voor deze vraag moet je de back-end opstarten:

* Download en unzip de back-end van xToledo( (file 'todo-backend.zip', enkel als je dat vooraf nog niet deed).
* Navigeer naar deze folder en start de server met npm start (in een terminal, e.g. in VS Code zelf). De server startte juist op als je de volgende boodschap ziet "Server is running on port 3000".
* Test dit door in de browser te navigeren naar [http://localhost:3000/](http://localhost:3000/" \t "_blank) . Als alles goed loopt, zie je in het venster de tekst "To-do back-end app is running...".

Je werkt voor deze vraag in de volgende twee bestanden:

* “question-4.html”: bevat HTML and CSS. Je mag deze bestanden niet aanpassen.
* “question-4.js”: schrijf hier de nodige javascript code.

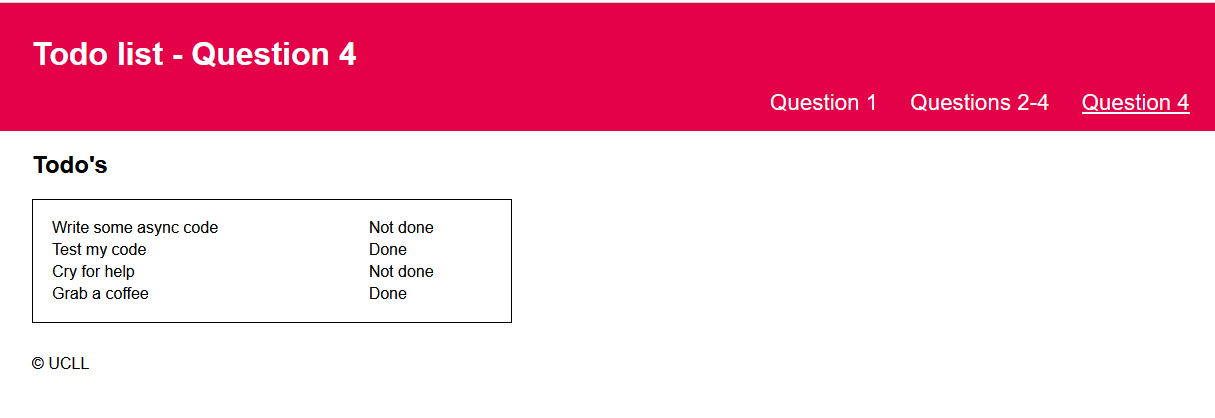
De video “question-4.mp4” toont het gewenste eindresultaat van vraag 4.1 and 4.2.

Vraag 4.1: Toon alle to-do items (1 punt)

De pagina "question-4.html" heeft reeds een table. Schrijf javascript dat alle to-do items fetcht van de back-end. Maak voor elk item een rij (tr) met 2 cellen (td): name en status. Je ziet een voorbeeld op de screenshot.

Gebruik het endpoint <http://localhost:3000/todos> om alle to-do items te fetchen.

De todo’s worden getoond zodra de pagina geladen wordt.



Zie volgende bladzijde voor vraag 4.2

Vraag 4.2: Toggle status (2.5 punten)

Voeg een derde cel met een checkbox in iedere rij: <input type=”checkbox” checked>. Het attribuut “checked” is optioneel en zorgt voor een vinkje in de checkbox.

Zorg ervoor dat het attribuut “checked” overeenkomt met de isDone status of het to-do item: het vinkje staat aan als de status overeenkomt met “done”.

Maak het mogelijk om de status van een to-do item te wijzigen door het vinkje van de checkbox aan of uit te zetten: als een “change” event gebeurt op de checkbox, stuurt de client een POST request naar het endpoint [http://localhost:3000/todos/toggle/<id](http://localhost:3000/todos/toggle/%3cid)> .

Zorg ervoor dat, na de klik, het vinkje van de checkbox en de status kolom altijd de meest actuele status van de to-do items weergeven worden zonder dat de pagina herladen moet worden.

