|  |  |
| --- | --- |
|  | Naam: |
| Groep: 1NMCT |

Beste studenten,

Hieronder kan je het examen van academiejaar 2018-2019 vinden.

Lees deze goed door, je zou alles moeten kunnen wat gevraagd is.

**Belangrijk is dat er in academiejaar 2019-2020 NIET gevraagd zal worden om een backend te maken of aan te passen. In dit examen zit geen socket.io, die moet je echter wel kunnen.**

Veel succes tijdens de uitvoering.

Device Programming I: Labo-examen

1. Haal de nodige bestanden af van Leho > Device Programming I > Modules > 12 examen.
2. Hernoem de afgehaalde map naar: **groep-voornaam.naam-DP1-Herex**.
3. Maak beide delen van het examen (backend en frontend).
4. Maak van de map een zip/rar-bestand met dezelfde naam als de map.
5. Dien in via Leho > Device Programming I > Opdrachten > examen.
6. Dien dit blad in en wis - na indienen- het zip/rar-bestand en directory van je harde schijf.

# Belangrijk

* Je krijgt voor deze oefening maximum 120 minuten tijd, je verdeelt zelf de tijd tussen het maken van de backend en het uitwerken van de frontend.
* Tijdens het examen is elke vorm van (online) communicatie of verbinding verboden.
* Je mag het materiaal gebruiken dat op je laptop staat.
* Je maakt geen nieuwe bestanden aan, alle bestanden zijn voorzien.

# Examen

## Voorbereiding: Aanmaken van de DB

* Open MySQLWorkbench en maak een connectie met je lokale MySQLServer.
* Open *mendeljev.sql* in MySQLWorkbench en voer uit. (NIET IMPORTEREN!!)
  1. In de SQL wordt een schema ‘mendeljev aangemaakt met een tabel *elements* en *types*.
* Indien je problemen hebt hiermee, geef een seintje aan de toezichthoudende docent.

## Deel1: Backend (Postman)

* Open in Visual Studio de map 'backend' en het bestand *app.py* via de Flask-server. De connectie met de database is reeds gemaakt en zou bij een standaardconfiguratie reeds moeten werken. In het bestand zijn reeds een aantal routes aangemaakt voor deel 2. Wijzig deze code niet.
* Voeg code toe onder regel 50.
* Importeer in Postman het bestand uit de map 'postman'.   
  Dit zijn een aantal requests die moeten aangemaakt worden in de backend.   
  De output ervan kan je vinden in een tekstbestand in de map postman.
  1. Types die beginnen met…  
     **GET types die beginnen met…**: Haal het type, typeCode en typeID op van alle types die beginnen met een zoekstring. Deze zoekstring moet variabel zijn.
  2. Ontdekkingsjaren  
     **GET unieke, gesorteerde ontdekkingsjaren**:   
     Haal een lijst op van alle unieke ontdekkingsjaren na 1800.
  3. Nieuw type  
     **POST een nieuw keyword**: Voeg het JSON-object {""typeCode": "MCTgrondstof", "type": "The best of the best"} toe aan de database via deze route.
  4. Wijzig type
     1. **PUT wijzig een type met een bestaand ID**: Wijzig de *typeCode* van een bestaand *typeID*.
     2. **PUT wijzig een type met een niet bestaand ID**: Breid de route van 4A uit zodat je een foutboodschap krijgt indien het id niet bestaat.

## Deel2: Frontend

* Zorg dat je het backendproject, dat je gekregen hebt bij het bronmateriaal, runt zodat je een API hebt waar je de data kan ophalen voor je frontend.
* Open in Visual Studio Code de map 'front'.   
  Deze map bevat één HTML-bestand, één CSS-bestand, twee JS-bestanden.
* In het HTML-bestand is reeds wat code gemaakt die je zal helpen om de oefening te begrijpen.
* **mendel.html – app\_index.js**
  1. Voorzie in het HTML-element met klasse *js-filter* een dynamisch opgebouwde lijst van de verschillende types van chemische elementen. De code die je zal moeten genereren via Javascript kan je terugvinden in het HTML-bestand.
  2. Voorzie in het HTML-element met klasse *js-mendeljev* een dynamisch opgebouwde lijst van de verschillende chemische elementen. De code die je zal moeten genereren via Javascript kan je terugvinden in het HTML-bestand.

Tip: Op het element op de juist plaats te krijgen voeg je de juiste klasse toe aan het element.  
dit werkt met rijen en kolommen:

Indien je *o-gridrow-1 o-gridcol-8* toevoegt komt het element in rij 1, kolom 8

Indien je *o-gridrow-12 o-gridcol-3* toevoegt komt het element in rij 12, kolom 3

De getallen van de rij en kolom kan je uit de response halen van de API.

* 1. Bij het klikken op een filter haal je enkel de elementen van één bepaald type uit de API en overschrijft je de volledige tabel met deze nieuwe gegevens.
  2. Bij het ‘hoveren’ over een element worden onderaan in de *js-info* klasse de details van het element getoond (symbool: afkorting: volledige naam: ontdekker) uit een API-call.
  3. Er is geen knop voorzien om alle elementen opnieuw te tonen na het filteren. Hiervoor moet je in deze oefening de pagina herladen.

**A screenshot of a cell phone

Description automatically generated**

Veel succes!