|  |  |  |
| --- | --- | --- |
| WPFW Inhaalweek Opdrachten | | |
| Opdracht | **Opdracht tekst voor studenten** | **Opmerkingen docenten** |
| Casus | Gegeven is een werkend ‘skeleton’-programma voor HHSMDB (Haagse Hogeschool Movie Database). HHSMDB is de start voor een applicatie die gebruikt wordt door studenten en docenten van de HHS om gegevens over films in vast te leggen. Op termijn moet dit uitgroeien tot de basis voor een community met filmliefhebbers aan de Haagse Hogeschool.  Tot nu toe is slechts een klein deel van de functionaliteit gebouwd. De functionaliteit die nu aanwezig is bevat een mogelijkheid om films toe te voegen, updaten, verwijderen of op te vragen. De films worden vanuit de backend weergegeven in de frontend door middel van een simpele tabel. | De studenten mogen allerlei functionele details zelf verzinnen/ aanvullen. Het gaat met name om zaken rond implementatie/ontwerp. Zolang de casus niet te veel vereenvoudigd wordt is er veel vrijheid mogelijk. |
| 1.1 (Dag 1) | Breid de gegeven (backend) code uit, zodanig dat ook gegevens kunnen worden opgeslagen van regisseurs. Regisseurs horen bij films. Films kunnen vanaf nu ook reviews krijgen. Deze bestaan uit een score tussen 1 en 5, een beschrijving, een aanmaakdatum en een naam van de gebruiker die de review schrijft. Let op, reviews worden alleen in de database opgeslagen als hun beschrijving minimaal 50 karakters bevat.  Om dit te realiseren moet het datamodel worden uitgebreid en er moeten meerdere CRUD-functies worden geïmplementeerd die de data bewerken. Dit wordt mogelijk gemaakt door het toevoegen van endpoints.  Je breidt de gegeven backend verder uit en maakt ook een endpoint die alleen alle films met een review uit de database ophaalt. Ook voeg je een endpoint toe die reviews ophaalt die maximaal 150 characters lang zijn en minder dan 1 jaar geleden zijn geplaatst (je gebruikt hiervoor LINQ).  Je maakt voor elke endpoint een DTO (Data Transfer Object) aan.  Om te testen of je gemaakte endpoints werken gebruik je Postman.  Hier horen de volgende portfolio vragen bij:   * Beschrijf wat het belang is van een DTO. | Een DTO wordt gebruikt om er te voor zorgen dat er alleen informatie wordt opgehaald die de eigenaar van de software wil terugleveren, vaak worden hier dus niet het ID of andere geheime informatie aan toegevoegd, maar wel de belangrijke informatie die nodig is. Bijvoorbeeld een kind mag niet de informatie zien die een docent heeft neergezet als opmerking over het kind zijn pakketkeuzes. Voor iedereen permissie daartoe heeft wordt een speciaal DTO gemaakt die niet de informatie van het veld DirectorNote. Voor een decaan wordt dan een DTO gemaakt die wel dat veld mag opvragen.  In combinatie met Authorization kunnen DTO’s zeer krachtig zijn, namelijk bepaalde endpoints kunen een bepaalde DTO request hebben, maar wanneer een user niet de juiste authorization heeft kan die ook geen aanvragen met dat DTO.  Verder DTO’s zorgen ervoor dat een gebruiker niet de hele database in een keer ophaalt, doordat alles naar elkaar verwijst. Door DTO’s te gebruiken kan de wijdte van het bereik afgeketend worden voor gebruikers. Door bepaalde classes niet toe te voegen.  Daarnaast is het ook preventief om te laten zien aan gebruikers wat voor velden er onder bepaalde tabellen hangen, aangezien er alleen maar bepaalde velden waar de gebruiker bij mag worden opgevraagd. |

**Film**

* **Voeg regisseur toe (geen context of class of parameter) V**

**Review**

* **Score, op controleren tussen 1 en 5 V**
* **Beschrijving, minimaal 50 tekens V**
* **Aanmaakdatum V**
* **Naam gebruiker V**

**EndPoints toevoegen**

* **Alleen films met review V**
* **Reviews met maximaal 150 characters en minder dan 1 jaar oud (LINQ) V**
* **Regisseur aanmaken**
* **Reviews aanmaken**

**DTO’s**

* **Regisseur V**
* **Film V**
* **Review V**