

9, Drop down in de Grid View

In deze les gaan we bepaalde dingen herhalen en op een ander manier gebruiken. We leren hoe we drop downs in een Gridview plaatsen en we leren ook hoe we een relatie kunnen gebruiken in de gridview.

In de gridview van de *Bestellingen* pas de kolom **status** aan:

```
[  
    'attribute'=>'status',  
    'filter'=>array('besteld'=>'is besteld', 'klaar'=>'is klaar', 'betaald'=>'is betaald'),  
],
```

In de database is de waarde van deze kolom *besteld*, *klaar* of *betaald*. Om duidelijk het verschil tussen de waarde van de kolom in de database en de getoonde waarde te laten zien zijn de getoonde waardes 'is besteld', 'is klaar' en 'is betaald'.

Let op dat

```
array('besteld'=>'is besteld', 'klaar'=>'is klaar', 'betaald'=>'is betaald')
```

hetzelfde is als

```
['besteld'=>'is besteld', 'klaar'=>'is klaar', 'betaald'=>'is betaald']
```

Als we nu een drop down willen bij de medewerkers dan moeten we dit hetzelfde aanpakken als we in de vorige les hebben gedaan:

Bovenaan in de view zetten we:

```
$medewerkerList=['1'=>'test1','2'=>'test2','3'=>'test3'];
```

Test dit uit. Als je test1 selecteert dan selecteer je dus de medewerkers met id 1.

Vanuit de controller maken we een object dat alle medewerkers bevat. Dit object geven we door aan de grid view (index). Daar maken we van het object een list en de list plaatsen we in de kolom als value van de key 'filter'.

Dus we veranderen de kolom medewerker:

```
[
    'attribute' => 'medewerker_id',
    'filter'    => $medewerkerList,
],
```

Opdracht 1

Zorg er nu voor dat je vanuit de controller de medewerkers opvraagt en deze verstuurd naar de index view.

Gebruik de

[ArrayHelper::map\(\)](#)

functie om je object om te zetten in een list.

En gebruik de list dan in de gridview.

In de kolom medewerker zien we nu de id's van de medewerker. Als we de naam willen afdrukken moeten we de relatie maken. Dit hebben we eerder gedaan, zet in het model van Bestelling de volgende code:

```
public function getMedewerkers()
{
    return $this->hasOne(Medewerker::className(), ['id' => 'medewerker_id']);
}
```

Plaats nu in de gridview in de index view van de bestelling:

```
[
    'attribute' => 'medewerker_id',
    'label'     => 'Medewerker',
    'filter'    => $medewerkerList,
    'value'     => 'medewerkers.naam'
],
```

Hiermee zetten we de waarde van deze kolom op *naam* van de relatie *medewerkers*.

Opdracht 2

Pas tenslotte het label aan en zorg ervoor dat de kolom waarin de medewerker wordt getoond in de gridview (van de index view van bestelling) er als volgt uit ziet:

[Image-1616622153704.png](#)

Als er een naam wordt geselecteerd dan worden alle bestellingen die door deze medewerker zijn uitgevoerd geselecteerd.


Hiermee zetten we de waarde van deze kolom op *naam* van de relatie *medewerkers*.

Opdracht 3

In deze opdracht gaan we alle stappen die we hebben uitgevoerd om de kolom *Medewerker* aan te passen nog een keer uitvoeren maar nu voor de kolom *menu_id*.

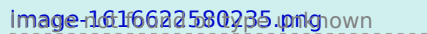
Pas de *menu_id* kolom aan in het bestellingenoverzicht (*index view* van *Bestelling*).

Zorg ervoor dat je met een drop down alle koffie soorten (*menu*) kunt selecteren. Je kunt dan dus alle bestellingen 'Americano' opzoeken.

Image-1616622602808.png

Zorg ervoor dat je de juiste relatie legt tussen de *Bestelling* en *Menu*.

Pas dan de grid view aan zodat je in de bestelling kolom geen id's meer ziet, maar dat je de naam van de bestellingen ziet.

Image-1616622580235.png

Let ook op het label. Dit is veranderd van *menu_id* naar *Bestelling*.

Succes!

--

Revision #13

Created 24 March 2021 18:53:04 by Max

Updated 25 November 2021 19:43:31 by Max