

# Yii 3

## Koffie opdracht

### De opdracht

Maak de *coffee* database met de drie tabellen, *menu* , *medewerker* en *bestelling* .  
Maak voor elke tabel een CRUD.

Volg de stappen hieronder om dit uit te voeren.

In stap 1 maak je een nieuwe database

In stap 2-6 maak je een nieuw Yii project aan. De uitgebreide instructie met voorbeelden kan je vinden in les 1.

In stap 7-12 maken we de tabellen en de CRUD pagina's.

**Voer stap 1 tot en met 13 uit.**

Stap 1, maak een database *coffee*.

Stap 2, maak een nieuw Yii project *coffee*. Gebruik de onderstaande regel voor in de Terminal.

Let op dat je `<naam van de map hier!!>` vervangt voor de naam van dit project

```
composer create-project --prefer-dist yiisoft/yii2-app-basic <naam van de map hier!!>
```

Stap 3, pas de database configuratie file in de Yii aan en laat deze verwijzen naar de *coffee* database.

Stap 4, pas de web.php in de Yii config aan en [zet smart routing aan](#) (uncomment het gedeelte bij 'urlManager').

Stap 5, maak de file functions.php met daarin de functies dd() en d(). Dit is beschreven in [les 1](#) .

Stap 6, roep functions.php aan vanuit de web.php configuratie file. Ook dit staat in les 1 beschreven.

Stap 7, we gaan nu de tabellen, models en CRUD's maken voor medewerker, menu en bestelling. Hieronder vind je terug hoe je de tabellen moet maken.

## medewerker

Maak de tabel in de database.

Gebruik hiervoor localhost/phpmyadmin (hiervoor moet XAMPP draaien)







#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	naam	varchar(200)	utf8mb4_general_ci		No	None		

Genereer het **model** en de **CRUD** voor *medewerkers* in Yii en zet de volgende namen in de database.utmelding meer krijgt.

## Medewerkers

Create Medewerker

Showing 1-5 of 5 items.

#	ID	Naam	
	<input type="text"/>	<input type="text"/>	
1	1	Ayoub	  
2	2	Brahim	  
3	3	Carla	  
4	4	Diego	  
5	5	Eisa	  

Houd rekening met [Yii-fout die we eerder tegenkwamen](#). Pas de gegeneerde code zodat je geen foutmelding meer ziet!

## Menu

Maak nu de tabel menu in de database

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	naam	varchar(200)	utf8mb4_general_ci		No	None		
3	prijs	int(11)			No	None		

Stap 10, Genereer het model en de CRUD voor *menu* en zet de volgende namen en prijzen in de database.















De prijzen zetten we in de database als centen. Dus E 1.95 wordt 195. Dit maakt het coderen eenvoudiger omdat we op deze manier met integers (gehele getallen kunnen werken).

Zet de testdata in de database zoals je hieronder kan zien.

## Menus

Create Menu

Showing 1-8 of 8 items.

#	ID	Naam	Prijs	
	<input type="text"/>	<input type="text"/>	<input type="text"/>	
1	1	Espresso	250	  
2	2	Latte Macchiato	350	  
3	3	Caramel Macchiato	360	  
4	4	Flat White	350	  
5	5	Cappuccino	350	  
6	6	Irisch Coffee	350	  
7	7	Americano	300	  
8	8	Cafe Latte	350	  

## Bestelling

Maak nu de tabel bestelling in de database.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	medewerker_id	int(11)			No	None		
3	naam	varchar(200)	utf8mb4_general_ci		No	None		
4	menu_id	int(11)			No	None		
5	status	enum('besteld', 'klaar', 'geleverd', '')	utf8mb4_general_ci		No	None		
6	timestamp	timestamp			No	current_timestamp()		

Stap 12, Genereer het model en de CRUD voor *bestelling*. (er hoeft geen data in de database gezet te worden voor *Bestelling*)

We hebben nu een tweede standaard Yii applicatie en we gaan nu aanpassingen maken in het invoerformulier 'Create Bestelling'.

Als we nu een nieuwe bestelling maken dan zien we het volgende scherm:

## Create Bestelling

Medewerker ID

Naam

Menu ID

Status

Timestamp

Save

Stap 13 -, *timestamp* halen we weg, want die wordt automatisch ingevoerd (default). Die hoeven we dus **niet** in te voeren.

Open de `_form.php` in de view van bestelling en haal de *timestamp* regel er uit.

Controleer in alle drie de CRUD's of ID in het invoerveld staat. Controleer hiervoor de drie `_form.php` bestanden in de views van medewerker, bestelling en menu.

Haal de ID's uit het invoer forms en test of je gegevens kan invoeren.

## Inleveren

1. Schermafdruk van de browser met je `form.php` van de view *bestelling*. Noem het bestand *yii-07-jouwnaam.png*
2. het bestand `_form.php` van de view bestelling (dus `views/bestelling_form.php`). Zet jouw naam in commentaar in dit bestand.

Schermafdruck ziet er zo ongeveer uit (je moet wel je hele browserscherm inleveren!).

Screenshot 2022-11-30 160819.png

--

# Forms en Dropdown

## Inleiding

We gaan onze koffie applicatie uitbreiden.

Om het invoeren van gegevens eenvoudiger te maken gaan we drop down menu's toevoegen.

We gaan naar twee soorten drop-downs kijken; die met vaste waarden en die met dynamische waarden (uit de database).

Bij het invoeren van een bestelling gaan we allereerst de status via een drop down invoeren.

Daarna gaan we waarde voor de medewerker ook uit een drop down halen. Deze waarde moet uit de database komen.

We gaan eerst de insert aanpassen. Dus aanmaken van een nieuwe bestelling. De update doen we later.

## Wat gaan we leren?

- hoe je in Yii in een formulier een drop-down kan maken.
- hoe je in Yii een query in de controller kan toevoegen en de resultaten kan doorsturen naar de view.

## Drop down in Bestelling

We gaan kijken naar de bestelling tabel en CRUD.

Als we de status in de database als een enum hebben aangemaakt dan weet Yii dat de status uit beperkt aantal waarden kan hebben en als het goed is maakt Yii dan vanzelf een drop down in het form.

Klopt dat heb jij een drop down?

Indien niet, dan kun je de code aanpassen in het form aanpassen.

In de view `_form.php` van Bestellingen moet de volgende regel staan:

```
<?= $form->field($model, 'status')->dropDownList([ 'besteld' => 'Besteld', 'klaar' => 'Klaar', 'betaald' => 'Betaald', '' => '', ], ['prompt' => '']) ?>
```

Deze regel laat een drop down menu in het formulier zien. Kijk eens goed naar de parameter van `dropDownList`.

```
[ 'besteld' => 'Besteld', 'klaar' => 'Klaar', 'betaald' => 'Betaald', '' => '', ], ['prompt' => '']
```

De eerste parameter is een associative array waarin elk element bestaat uit een key en een value.

De key is de waarde die het element uit het form krijgt (de value) en deze waarde wordt door Yii in de database gezet. De key komt dus overeen met de waarde in de database. De value van het associatieve array is wat de gebruiker op het scherm ziet.

Deze waarde kan je dus veranderen. Je kunt dus bijvoorbeeld 'betaald' veranderen in 'afgerekend'. Het enige dat dan gebeurt is dat je iets ander op het scherm ziet.

In dit voorbeeld is de key en de value gelijk (de waarde op het scherm is hetzelfde als de waarde in de database).

Dus samengevat, de key is de waarde in de database en de value is wat de gebruiker op het scherm ziet.

Stel we willen in het form van bestelling de medewerker veranderen. Dan moeten we dus de *foreign key* die naar medewerker verwijst veranderen. Het juiste id van de medewerker moet in de tabel bestelling worden ingevuld.

De waarde wordt dus het id, maar dat is niet wat je de gebruiker wilt laten zien.

Dus als we voor de medewerkers een drop down willen maken dan hebben we een associative array nodig dat er zo uit ziet:

```
[ '1'=> 'Ayoub', '2'=> 'Brahim', '3'=> 'Carla', '4'=> 'Diego', '5'=> 'Eisa' ]
```

De keys zijn de id's die als foreign keys in de bestelling tabel staan en de namen zijn de namen van de medewerkers.

## Opdracht

Maak nu een statische drop down met de waarden zoals in het voorbeeld van de vorige pagina (Ayoub, Brahim, Carla, ....).

*Statisch betekent dat de waarden niet veranderen. Het tegenovergestelde is dynamische en dat betekent dat de waarden veranderen, omdat deze waarden bijvoorbeeld uit de database komen.*

De waarden worden dus (nog) niet uit de database gehaald.

## Inleveren

1. Schermafdruk yii-08a-jouw-naam met het form waarin je de drop down (opengeklapt) laat zien. Maak een schermafdruk van je **gehele** browser.

## Voorbeeld

Screenshot 2022-11-28 195157.png

## *Dynamische drop down*

## Opdracht, Dynamische drop down

In de vorige opdracht hebben we een drop down van medewerkers gemaakt, maar we willen de lijst van medewerkers natuurlijk uit de database halen.

*In deze opdracht leg ik stap-voor-stap uit wat je moet doen om dat voor elkaar te krijgen.*

*Lees alles aandachtig door en sla geen stappen over!*

Data uit de database halen doen we in de controller.

Open de BestellingController. Vanuit deze code wordt de create view aangeroepen en vanuit de create view wordt de \_form.php aangeroepen. Waarom dit in twee stappen gaat leggen we later uit.

We veranderen de code in Bestelling controller:

```
use app\models\Medewerker;

use Yii;

..
```

```

..

public function actionCreate()
{
    $model = new Bestelling();
    $medewerkers = Medewerker::find()->all();

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['view', 'id' => $model->id]);
    }

    return $this->render('create', [
        'model' => $model,
        'medewerkers' => $medewerkers
    ]);
}

```

De eerste regel `use app\models\Medewerker;` zetten we bovenaan in de *BestellingController*. Hiermee vertellen we Yii dat vanuit de *BestellingController* gebruik willen gaan maken van het model *Medewerker*.

Let op dat je op de bovenaan in je code `use Yii;` moet hebben staan.

Let ook op wat je moet aanpassen, de code die hierboven staat kan in een nieuwere Yii versie iets anders zijn. Kopieer **alleen** de regels die er voor zorgen dat de create view de `$medewerkers` krijgt.

We kunnen de medewerkers ook ophalen via een sql-query. Je **mag** regel 10 ook vervangen in de volgende twee regels:

```

// $medewerkers = Medewerker::find()->all();
// dit is hetzelfde als
$sql="select * from medewerker";
$medewerkers = Yii::$app->db->createCommand($sql)->queryAll();

```

Vanuit de controller zie je dat de `$medewerkers` (waar dus alle medewerkers in staat) wordt doorgegeven aan de *create* view (van bestelling).

## Dus wat hebben we gedaan tot nu toe?

In de function *actionCreate* die wordt aangeroepen als we een nieuwe bestelling willen maken, voegen we een object toe waarin alle medewerkers zitten. In regel 10 halen we alle medewerkers



op en in regel 18 sturen we het resultaat naar de view create.php (van *bestelling* ).

In de view create.php van bestelling controller passen we het laatste stukje code aan

```
<?= $this->render('_form', [  
    'model' => $model,  
    'medewerkers' => $medewerkers  
) ?>
```

Hiermee geven we het object \$medewerkers door aan de view *\_form* van bestelling.

We hebben dus deze volgorde afgelegd: *bestellingController.php* (controller) -> *create.php* (view) -> *\_form.php* (view)

In het view *\_form* zetten we nu bovenaan in het PHP-gedeelte.

```
_dd($medewerkers);
```

Dit is de debugfunctie en hiermee controleren we of we inderdaad alle medewerkers naar de view hebben gestuurd.

We hebben nu als het goed is een lijst van medewerkers in de *\_form* maar we moeten het ombouwen naar een associatieve array.

Pas hiervoor de code aan in de view *\_form* van bestelling.

```
<?php  
  
use yii\helpers\Html;  
use yii\widgets\ActiveForm;  
use yii\helpers\ArrayHelper;  
  
// hiermee maken we een array met de id's(als key) en de name (als value)  
$medewerkerList = ArrayHelper::map($medewerkers,'id','naam');  
_dd($medewerkerList);  
?>  
  
...  
...  
...
```

Deze code laat het eerst stukje van de view *\_form* zien.

Met de functie `ArrayHelper::map` zetten we het object `$medewerkers` om in een associatieve array. Met `_dd()` laten we dat zien. Probeer maar! Als het goed is, zie je het volgende:

```
[
    1 => 'Ayoub'
    2 => 'Brahim'
    3 => 'Carla'
    4 => 'Diego'
    5 => 'Eisa'
]
```

En dit is precies wat we nodig hebben om de Drop Down te maken.

Pas de regel in de view `_form` van bestelling waarin de user het id van de medewerker moet intypen aan. Verander deze regel in:

```
<?= $form->field($model, 'medewerker_id')->dropDownList($medewerkerList, ['prompt' => ''])-
>label('Medwerker') ?>
```

Als het goed is heb je hiermee een werkend menu gekregen.

## Create Bestelling

**Medewerker ID**

Ayoub  
Brahim  
Carla  
Diego  
Eisa

**Status**

Save

Gelukt? Wordt de lijst van medewerkers in de drop down uit de database gehaald?

## Inleveren

1. de view `_form.php` van bestelling
2. schermafdruck yii-08b-jouw-naam met het form waarin je de drop down van de medewerkers (opengeklapt) laat zien. Maak een schermafdruck van je gehele browser

# Nog een drop down

We gaan nog een keer stap-voor-stap alle stappen uitvoeren om nog een drop down in bestelling te maken.

## Opdracht

Maak nu een drop down voor de bestelling zodat de medewerker uit een lijstje van koffiesoorten kan kiezen bij het opnemen van de bestelling.

### Create Bestelling

Medewerker ID

Naam

Menu ID

Espresso

Latte Macchiato

Caramel Macchiato

Flat White

Cappuccino

Irish Coffee

Americano

Cafe Latte

Gebruik hiervoor het stappenplan dat hieronder is beschreven.

## Het stappenplan voor een drop down

Schrijf eerst het volgende op:

In welk model komt de drop down? Dit noemen we de target.  
(in opdracht 2 hierboven is dat *Bestelling* )

Waar komt de informatie voor de Drop Down uit? Dit noemen we de source.  
(in opdracht 2 hierboven is dat *Menu* )

## Voer nu de stappen 1-6 uit

1. Zet in de target controller welke source model we willen gebruiken.

```
// in controller van target
use app\models\<SourceModelnaam>;
```

2. Haal in de target controller in de `actionCreate()` de informatie uit de source op.

```
// sourceName bedenkt jij zelf en daarin staat de informatie die in de drop down moet komen.
$sourceName = source::find()->all();
```

3. Geef vanuit de target controller in de `actionCreate()` de informatie door aan de view.

```
return $this->render('create', [
    'sourceName' => $sourceName,
    ....
]);
```

4. Open de view `create` in de target (view) en geef de informatie door aan `_form` view.

```
<?= $this->render('_form', [
    'sourceName' => $sourceName,
    ...
]);
```

5. Open de view `_form` in de target (view) en zet de informatie om in een associatieve array

```
// in dit voorbeeld worden id en naam gebruikt:
// id is het veld dat moet worden opgeslagen
// naam is het veld dat in de drop te zien is
$sourceNameList = ArrayHelper::map($sourceName,'id','naam');
```

6. Pas het invoerveld aan in de `_form` van de target (view).

```
// veld_in_de_db is het veld van de target dat in dit invoerveld wordt weggeschreven.
<?= $form->field($model, 'veld_in_de_db')->dropDownList($$sourceNameList, ['prompt' => '']) ?>
```

Als je alles correct gedaan hebt dan moet je nog 1 regel aanpassen in de code om de functionaliteit te gebruiken. Yii controleert automatisch of alle databasekolommen zijn ingevuld in het formulier. Dit controleren wordt *validatie* genoemd. Deze regels staan in het model. Omdat je geen Bestelling ID opgeeft bij het maken van een bestelling moet je deze requirement verwijderen op regel 33 van model Bestelling.

```
return [  
    ['naam', 'medewerker_id', 'menu_id', 'status'], 'required'],  
    ...  
    ...  
]
```

## Inleveren

1. schermafdruck yii-09-jouw-naam met het form waarin je de drop down van de producten (opengeklapt) laat zien. Maak een schermafdruck van je gehele browser.
2. de *BestellingController.php* en zet jouw naam bij de aanpassingen die je hebt gedaan.

## *Drop down afmaken*

## Inleiding

De insert en update lijken op elkaar en gebruiken hetzelfde form (\_form.php).

We hebben nu de insert aangepast, maar de update nog niet. Controleer maar! We moeten nu de update dus ook nog aanpassen.

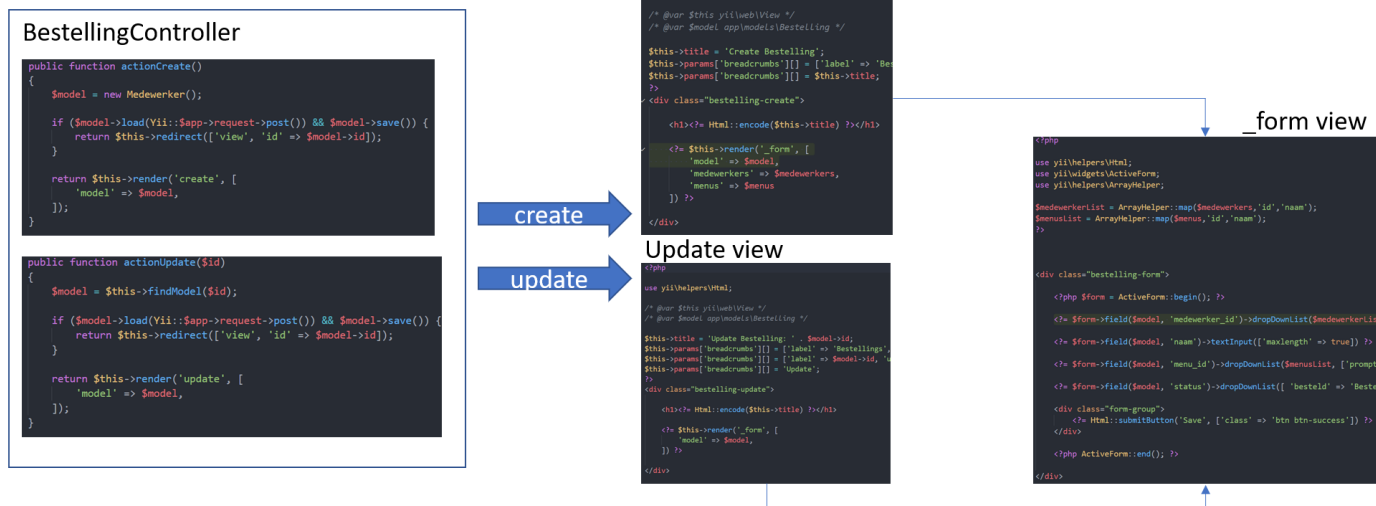
## Wat gaan we leren?

- Hoe in Yii de insert en update met elkaar zijn verweven. Dit werk ook ongeveer zo in Laravel.
- Hoe we labels aanpassen in het Yii insert- en update formulier.

## Aanpassen update view

Weet je nog dat de -update view niet rechtstreeks door de actionCreate van de BestellingController wordt aangeroepen?

Hoe zit dat nu?



De *actionCreate* en *actionUpdate* vanuit de controller gaan bieden naar hun 'eigen' view maar via deze eigen view komen beide op hetzelfde form uit. Dat komt omdat een *update* en *create* hetzelfde form gebruiken.

Als we nu dus een update uitvoeren van de bestelling (pennetje vanuit de gridview) dan krijgen we een foutmelding. Dat komt omdat de form verwacht dat er gegevens voor de drop down worden meegestuurd.

Om dit te fixen kopiëren we de code in de controller die we in de *actionCreate* hebben gemaakt dus naar de *actionUpdate*.

Deze twee functies zijn bijna hetzelfde. Bij de create wordt alleen een nieuw object gemaakt en bij de update wordt een bestaand object ingeladen.

Nu moeten we de objecten nog doorgeven van de *update* view naar de *\_form* view. Net zoals we dat ook deden bij de *create*.

## Opdracht

Zorg ervoor dat de update weer werkt en dat de objecten op de juiste manier worden doorgegeven. Lees hiervoor de uitleg die op de vorige pagina staat.

Pas de *actionUpdate* in de *BestellingController* aan.  
Pas de update view van de *BestellingController* aan.

Zorg dat de update van een Bestelling goed werkt.

## Inleveren

1. de *BestellingController.php* en zet jouw naam bij de aanpassingen die je hebt gedaan.

# Final Touch

## Opdracht

### De final touch

Laten we nog wat labels aanpassen. Dit zijn de teksten die boven de invoervelden in het form staan.

Pas de labels in het form aan zodat er Medewerker, Klantnaam, Bestelling en Status Bestelling komt te staan:

### Create Bestelling

Medewerker

Klantnaam

Bestelling

Status bestelling

Save

Zoek zelf uit hoe dit moet. Tip gebruik de zoektermen:

**How to change label text of the ActiveForm?**

## Inleveren

1. schermafdruck yii-010b-jouw-naam met het bestellingen form waarin je de aangepaste labels laat zien. Maak een schermafdruck van je gehele browser.

# Relaties en drop down

# Todo inleiding

*We gaan nog één keer 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

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* .

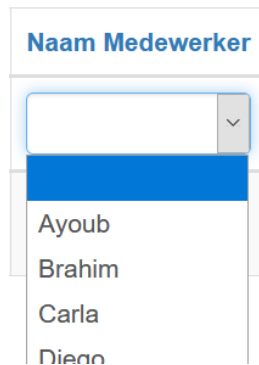
## Inleveren

1. schermafdruck yii-011a-jouw-naam met de bestellingen index view waarin je laat zien dat de relatie is gemaakt met de tabel medewerker. Maak een schermafdruck van je **gehele** browser.

# Medewerker tonen

## Opdracht

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:



The screenshot shows a web form with the title "Naam Medewerker" in blue text. Below the title is a dropdown menu. The dropdown menu is currently open, showing a list of names: Ayoub, Brahim, Carla, and Diego. The first option, Ayoub, is highlighted with a blue background. The dropdown menu has a small downward arrow icon on the right side of the input field.

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* .

## Inleveren

1. schermafdruck yii-011b-jouw-naam met de bestellingen index view waarin je laat zien dat de drop down is gevuld met de namen van de medewerkers. Maak een schermafdruck van je gehele browser.

# Menus en drop down

## Opdracht

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.

Bestelling
<div><div></div><div></div></div> <div>Espresso</div> <div>Latte Macchiato</div> <div>Caramel Macchiato</div> <div>Flat White</div> <div>Cappuccino</div> <div>Irish Coffee</div> <div>Americano</div> <div>Cafe Latte</div>

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.

Bestelling
<div><div></div><div></div></div>
Espresso
Americano

Let ook op het label. Dit is veranderd van *menu\_id* naar *Bestelling* .

**Succes!**

# Inleveren

1. Laat in de les zien dat alles werkt.

lukt dat niet maak dan een heel kort demo filmpje (10 to 30 seconden).

1. In dat filmpje laat je zien dat je een nieuwe bestelling aan kan maken en gebruik maakt van de drop downs in het form.
2. In het filmpje laat je zien dat je een bestelling kan aanpassen.
3. In het filmpje laat je zien dat de grid view werkt en dat via de twee drop downs een selectie kan maken.

--

---

Revision #4

Created 30 May 2024 17:49:27 by Max

Updated 24 October 2024 09:25:22 by Max