Yii

- Yii 1 - Intro
- Yii 2
- Yii 3
- Yii Challange

Yii 1 - Intro

01 Wat is Yii?

In deze les gaan we leren wat een framework is en waarom we Yii gaan leren.
We leren verder wat MVC en CRUD is en wat heeft dat te maken heeft met een framework.

Voor het examen moet je een webapplicatie bouwen.

Je moet zelf weten hoe je dat doet. De meeste studenten kiezen voor PHP. Waarom? Het is relatief eenvoudig en de meeste web applicaties zijn ontwikkeld met PHP (bron (Koppelingen naar een externe site.)).

Veel websites lijken op elkaar en doen vaak dezelfde dingen. Ze maken een verbinding naar de database, laten gegevens zien, je kunt aanloggen, je kunt soms gegevens veranderen en ga zo maar door.

Om niet telkens al deze 'standaard' zaken van 'scratch' af aan te moeten bouwen zijn er zogenaamde frameworks ontwikkeld. Elk framework is eigenlijk een verzamelingen bouwblokken die je telkens op nieuw kunt gebruiken voor jouw website. Alle frameworks zijn net een beetje anders, maar er zijn ook veel overeenkomsten.

Wat zijn belangrijke zaken als je voor een framework kiest:

1. Wordt het framework actief onderhouden; bestaat het al wat langer? Je wilt niet in iets gaan investeren dat volgend jaar misschien niet meer bestaat.
2. Hoe moeilijk is het om een framework te leren; wat is de complexiteit?
3. Hoe snel kun je iets ontwikkelen met het framework?
4. Hoe flexibel is een framework; kun je er echt alles mee maken?

Bij punt 3 en 4 is het vaak het een of het ander. Of je kunt heel snel iets bouwen en je bent niet zo flexibel, of je kunt er eigenlijk alles mee ontwikkelen maar het kost veel tijd om dat te doen.

Bijvoorbeeld: PHP is super flexibel je kunt er echt alles mee ontwikkelen, het Laravel framework is ook redelijk flexibel, het Yii framework is iets minder flexibel en als je naar bijvoorbeeld WordPress kijkt (wat je ook als een soort framework zou kunnen zien), dan is dat het minst flexibel. Aan de andere kant kun je met WordPress heel snel een site maken, je bent alleen een beetje beperkt in de functionaliteiten.

Waarom Yii?

Laravel is het meest gebruikte PHP-framework. Yii staat in de meeste lijstjes ongeveer op plaats 7. Dat komt, omdat je weliswaar "snel even wat kan bouwen", maar je kunt het vrij lastig alles precies zo maken als jij het wilt. Het is is minder flexibel en om het precies zo te krijgen wilt dan moet je meer moeite doen. Als je straks een project maakt dan kun je waarschijnlijk ongeveer 70%-80% van je code maken met Yii, daarvoor hoef je heel weinig te programmeren en heb je tijd om je te storten op de overige 20%-30%. De projecten en examens die jullie gaan doen lenen zich goed om met Yii te maken. Je bent namelijk redelijk vrij in het ontwerp en kunt dus de 'Yii-manier' aanhouden.

Maar wat dan als ik later verder wil met (bijvoorbeeld) Laravel? Dat kan en Yii en Laravel hebben dezelfde opzet, MVC

Links to an external site.. Je zult in Laravel dus veel dingen terugvinden die je in Yii hebt geleerd.

MVC?

MVC, wat is dat? MVC staat voor Model, View en Controller. Deze drie-deling wordt gemaakt in je code, zodat je eenvoudiger code kunt terugvinden.

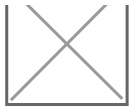
Model	Hierin staat waar de verbinding tussen jouw code en de database beschreven
View	Dit is de front-end en is vooral HTML/CSS (bootstrap) met hier een daar een 'vleugje' PHP
Control	Dit is waar je bepaald wat er gebeurt, als je op een knop drukt ergens dan wordt de code uit de bijbehorende control uitgevoerd.

In een plaatje ziet dat er als volgt uit. De gebruiker doet wat en stuurt daarmee een verzoek (request) naar de juiste controller. De controller 'overlegt' met het model als er iets met data moet gebeuren en het resultaat wordt via de view aan de gebruiker getoond.

image-1594413597727.png

In de introductie van deze module (workshop) komen deze begrippen terug. Als het nu allemaal nog wat onduidelijk is, geen probleem we gaan er mee aan de slag en zullen snel een gevoel krijgen wat MVC precies is en hoe het werkt.

Uitleg MVC YouTube (EN)



CRUD?

Nog even....waar staat CRUD ook alweer voor? CRUD staat voor **C**reate, **R**ead, **U**pdate en **D**eleten. In een CRUD-applicatie kun je dus gegevens aanmaken (create), lezen (read), bewerken (update) en verwijderen (delete).

Voor jouw examen moet je minimaal een CRUD kunnen maken.

Yii heeft een CRUD-generator en je kan zonder te coderen een standaard CRUD-applicatie maken. Super handig dus voor het examen. In de volgende les gaan we hier gelijk mee aan de slag.

Quiz

We gaan even een korte kennis quiz doen en gaan je bevragen over MVC, CRUD, en Frameworks. Als je de tekst op deze pagina goed hebt gelezen dan kun je de kennis quiz maken.

02 Installation Yii

In de komende paar opdrachten gaan we alles klaarmaken en installeren voor het werken met Yii. Aan het eind van de opdracht 4 heb je volledig werkende Yii CRUD-applicatie.

Een CRUD-applicatie is een applicatie waarmee je records kunt aanmaken (Create), records kunt lezen (Read), records kunt aanpassen (Update) en records kunt verwijderen (Delete).

We gaan drie stappen doen om een werkende web app te maken:

1. installeren van alle software
2. maken van een eenvoudige database
3. generen van code

In drie stappen hebben we straks een *werkende* web app in PHP gemaakt.

Voordat je met deze les begint weet je hoe je met phpMyAdmin (Koppelingen naar een externe site.) (of een andere tool) een database kunt maken en hoe je een .sql file kan importeren. Ook heb je XAMPP ge nstalleerd.

Installatie

De installatie kent een aantal stappen. Hieronder zijn de 5 stappen beschreven die je moet uitvoeren om tot een succesvolle Yii installatie te komen.

1 - XAMPP

Je hebt XAMPP (Koppelingen naar een externe site.) ge nstalleerd. Voor Yii hebben dit nodig, omdat we een database gaan gebruiken.

Heb je nog geen XAMPP of werkt die niet meer dan kun je hier (Koppelingen naar een externe site.) lezen hoe je XAMPP kan installeren.

2 - Composer

Zorg er voor dat je composer (Koppelingen naar een externe site.) hebt ge nstalleerd. Composer is de installer van de WEB tools.

Uitleg, wat is Composer?
Als we een framework als Yii of Laravel gaan gebruiken dan installeren we eigenlijk een hele grote doos met allemaal bouwblokken. Deze blokken moeten allemaal samen werken en zijn vaak afhankelijk van elkaar. De blokken worden door verschillende mensen ontwikkeld en op n of andere manier moet er voor worden gezorgd dat de juisten blokken zijn ge nstalleerd. Hiervoor dient Composer. Composer is als het ware de installer van PHP tools, deze worden vaak libraries of packages genoemd. Librabries zijn dan de blokken functionaliteit die je kunt gebruiken om een programma te maken.

https://getcomposer.org/ (Koppelingen naar een externe site.)

Let op: check goed of de installatie errors/warnings geeft. Zijn er errors, lees deze dan goed en los de issues met behulp van ChatGPT op!

Tip: Zorg dat XAMPP al in ge nstalleerd voordat je composer installeert; composer heeft namelijk de locatie van php.exe nodig.

In deze korte video wordt getoond hoe je Composer kunt installeren:

https://www.youtube.com/embed/Rvb6DQ5PYHM

3 - Create new yii Project

We maken het Yii project **world** aan.

Je hoeft niet zelf een map/folder aan te maken. Dit doet composer voor jou. Zorg ervoor dat je in de terminal in de parent-folder bent.

Vul de volgende code in de terminal:

```
composer create-project --prefer-dist yiisoft/yii2-app-basic world
```

Dit commando maakt een folder aan met de naam world. Dit is je nieuwe Yii project folder (er staan honderden bestanden in).

Errors?

image.png	Let op, check of je errors en of warnings krijgt! Zie je die, probeer die dan met behulp van ChatGPT op te lossen! Extra uitleg: bij sommige installaties van XAMPP staan bepaalde extensies niet aan. Zo staat bijvoorbeeld de ZIP extensie soms uit in de PHP config. Als dit zo is dan kan de Yii installatie fout gaan. Check daarom de output en pas de configuratie met behulp van ChatGPT aan.

Let op dat het nieuwe Yii project wordt gemaakt in de folder waarin je de cmd box opent. Stel je wilt je jouw nieuwe project op je Desktop maken dan open je een cmd box en je tikt het commando `cd Desktop` in. Voer dan het composer commando uit (zoals hierboven aangegeven) en het nieuwe Yii project zal op je Desktop worden gemaakt.

Start Development Server

yii heeft een ingebouwde php server als je die opstart dan draait jouw website op

http://localhost:8080/ (Koppelingen naar een externe site.)

Let wel dat je XAMPP ook blijft draaien voor je database.

In VSC open je jouw Yii project en daarna open je een nieuwe terminal.

image-1618344180761.png

In de terminal type je dan het commando in:

```
php yii serve
```

De Yii webserver draait nu. Ga naar je browser en controleer of op je via localhost:8080 jouw nieuwe Yii project ziet.

Wil je cmd box gebruiken of lukt dit niet. Kijk dan naar de instructiefilm (Koppelingen naar een externe site.)

https://www.youtube.com/embed/5idkjTWIL7g

4 - Configure web.php

We moeten nu nog het een en ander configureren in Yii. We willen dat Yii zoveel mogelijk voor ons regelt. Om te beginnen mag Yii de routing regelen (dit wordt uitgelegd in de volgende les (Koppelingen naar een externe site.)).

Open **config/web.php** - en zet het gedeelte dat urlmanager heet, uit commentaar (uncomment prettyURL). Het moet er als volgt uitzien (ongeveer op regel 50 van config/web.php).

```
'urlManager' => [
            'enablePrettyUrl' => true,
            'showScriptName' => false,
            'rules' => [
            ],
        ],
```

Dit is een instelling die de routing (Koppelingen naar een externe site.) (waarover later meer) eenvoudiger maakt.

5 - Database

Let er op dat je **database goed is gedefinieerd** en dat je ook de relaties heb vastgelegd. Dat kun je doen met phpMyAdmin (Koppelingen naar een externe site.) . De ingebouwde Yii CRUD-generator (Koppelingen naar een externe site.) werkt alleen goed als de database en de relaties goed zijn vastgelegd.

Maak de database world en importeer deze file: https://www.roc.ovh/attachments/22

Lukt niet of weet je niet precies hoe je een database moet importeren? In deze les (Koppelingen naar een externe site.) staat beschreven hoe je deze database kunt installeren. In deze les staat de SQL-file world.sql (Koppelingen naar een externe site.) waarmee je de database World kan maken.

We gaan Yii vertellen welke database er moet worden gebruikt.

We openen de file **config/db.php** en zetten daar het volgende in:

```
return [
    'class' => 'yii\db\Connection',
    'dsn' => 'mysql:host=127.0.0.1;dbname=world',
    'username' => 'root',
    'password' => '',
    'charset' => 'utf8',
];
```

Let op dat je geen localhost gebruikt om naar je database te verwijzen. Gebruik 127.0.0.1.

6 - Klaar?

```
Klaar? Test jouw installatie!
```

Je had al gezien dat je een leeg Yii project hebt, ga nu naar: http://localhost:8080/gii

Jouw scherm ziet er ongeveer zo uit:

Screenshot 2022-05-20 183633-1.png

Maak een screen shot van jouw *hele* scherm met browser en lever deze in.

02 Eerste CRUD

Je hebt Yii geinstalleerd en je hebt de database *world* geinstalleerd?

Het is tijd om een CRUD te maken met Yii.

Eerste CRUD

We gaan een CRUD maken en met Yii is dat heel makkelijk.

We moeten eerst een verbiding met de database maken. Weet je nog welke letter van MVC daar ook alweer voor stond?
Yep, de M van Model. Het model beschrijft hoe jouw Yii applicatie een verbinding met de database kan maken. We gaan dus een Model maken.

Stap 1 - models maken

We gaan opnieuw naar de component builder, open http://localhost:8080/gii/model (Koppelingen naar een externe site.)

Klik links op 'Model Generator' en type de naam van de databasetabel **country** in:

image-1594285274135.png

De naam van het model wordt automatisch aangemaakt **Country** . Let op dat de 'Model Class Name' met een **hoofdletter** begint; kijk goed naar het voorbeeld. De naam van een class (oop) wordt altijd met een hoofdletter geschreven.

Druk onderaan op de pagina op *preview* , Yii laat nu zien welke file hij gaan aanmaken voor je. Druk op **generate** om de file te maken.

De file **model/country** is nu aangemaakt, we gaan later meer in op de inhoud van deze file. Wat voor nu belangrijk is om te onthouden is dat de model/country.php file de verbinding is tussen de tabel country en Yii. Elke keer als Yii informatie uit de country tabel nodig heeft dan gebruikt Yii de model/country.php file

Omdat er vanuit de country tabel wordt verwezen naar de andere twee tabellen, **City** en **Countrylanguage** maak je nog twee modellen van deze twee tabellen.

In je model directory (folder) heb je nu dus de volgende files staan:

```
City.php
ContactForm.php
Country.php
Countrylanguage.php
LoginForm.php
User.php
```

Controleer dit en als het klopt (let op de hoofdletters!) ga dan door.

Stap 2 - CRUD maken

We gaan de complete CRUD maken.

Ga terug naar de component builder, open (Koppelingen naar een externe site.)

http://localhost:8080/gii/model (Koppelingen naar een externe site.)

Klik links op 'CRUD Generator' en type het volgende in:

image-1618227641051.png

Let goed op de hoofdletters, deze zijn belangrijk. Het View path wordt leeg gelaten, daar kan Yii zelf iets voor kiezen.

Druk op preview, je ziet 8 files die klaar staan om te worden aangemaakt.

image-1618227971123.png

Druk op **generate** om de files aan te maken.

Controller

In de directory controllers staat een file CountryController.php. Dit is de controller van de country CRUD. Al het denkwerk dat nodig is om de country CRUD aan te sturen zit in de CountryController.php file.

View

In de views/country directory staan de views. De index.php is de standaard view en deze file bevat de (voornamelijk HTML code) om de pagina weer te geven.

Weten we nog waar de controller en view ook alweer vandaan komen? Het Framework is gebouwd volgens de MVC-architectuur. In stap 1 hebben we het model gemaakt en in deze 2de stap hebben we de view en controller gemaakt.

image-1612557541726.png

(deze sheet is uit de vorige les)

De controller is waar alle verzoeken van de gebruiker naar toe gaan. De view is de presentatielaag met voornamelijk HTML, CSS, en JavaScript. En in het model wordt de verbinding met de database gemaakt.

Klaar ?

Ga naar http://localhost:8080/country (Koppelingen naar een externe site.)

image-1594288457454.png

Je hebt een CRUD van Country gemaakt, compleet met search en sort-functionaliteiten. Je hebt ook een mooie pagina selector waarmee je daar de verschillende pagina's kan navigeren. Je kunt country's lezen/tonen, aanpassen, verwijderen en aanpassen. Probeer het maar! Wees niet bang om de database 'kapot' te maken, want je kunt in een paar tellen de database opnieuw importeren.

In de volgende lessen gaan we de automatisch gebouwde applicatie stap-voor-stap aanpassen.

Voor nu kan je trots zijn op je eerste (?) web CRUD applicatie.

Extra hulp met Filmpje

Niet gelukt? Kijk in dit filmpje nog een keer alle stappen door: Instructie Yii - les1 (Koppelingen naar een externe site.)

https://www.youtube.com/embed/6NRMaJgSmLw

Opdracht 1

Maak voor de tabellen:

- city
- countrylanguage

ook een (complete) CRUD met de Crud Generator. Je moet natuurlijk wel eerst de *models* hebben gemaakt in stap -1.

Controleer op http://localhost:8080/city (Koppelingen naar een externe site.) en http://localhost:8080/countrylanguage (Koppelingen naar een externe site.) of de CRUD werkt.

GIT

"*Meester, gisteren werkje het nog en ik heb niets gedaan?*"

Bij Yii loop je soms vast. Je maakt wijzigen in een door het framework gemaakte files. Soms werkt er opeens *niets* meer (echt dat gebeurt iedereen!).

Met GIT kun je ervoor zorgen dat je steeds een werkende versie opslaat. Werkt het dan niet meer, dan kun je terug naar de vorige werkende versie.

Wert GIT nog niet in jouw VCS? Hier staat hoe je GIT in VCS kunt 'aanzetten'.

Opdracht 1

Ga in de code-editor (VSC) naar de map **views** en probeer te vinden waar het overzicht Countries is gemaakt. Haal code weg zodat de kolom *Surface Area* niet meer wordt getoond.

image-1612558210976.png

Opdracht 2

Kun je de in dezelfde view de groene knop 'Create Country' verplaatsen en onderaan de pagina plaatsen?
Dus het moet er zo uit te komen zien:

image-1612558884647.png

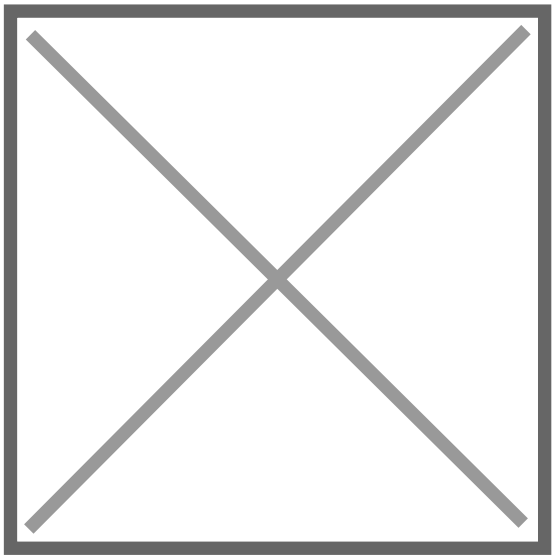
Inleveren

1. Lever een compleet screenshot in van de **country** read page (Grid View; zoals plaatsje hierboven met het rode kruis).
2. Lever een compleet screenshot in van de read page (Grid View) van ***city***
3. Lever een compleet screenshot in van de read page (Grid View) van ***countrylanguage***

4. Lever het (aangepaste) bestand *views/country/index.php* in - noem dit bestand jouw-naam-index.php

Resultaat

Jouw *country* overzicht ziet er dus zo uit:



--

03 Countries CRUD in Yii

Je hebt een CRUD gemaakt voor de country's tabel. De volgende vragen gaan over de door jouw gemaakte applicatie.

Je kunt deze vragen pas maken als je de pagina de opdrachten in een CRUD voor de world database hebt gemaakt hebt gemaakt.

Voor een paar vragen moet je de applicatie (http://localhost:8080/country) hebben draaien.

04 Routing

In deze les leer je wat routing is en hoe je binnen Yii de routing kunt instellen.

Het begrip routing komt terug in alle of veel frameworks waarmee je web applicaties kan maken; Yii, Laravel, Flask, Django,...

Wat is routing? (Algemeen)

Routing is de manier waarop de webserver en jouw programma weet welke pagina er moet worden getoond. Kijk maar eens goed naar de URL in je browser. Je kunt de URL opdelen in 5 stukjes:

image-1594292626981.png

Onderdeel	Wat is het?	Uitleg
Scheme	protocol	hoe wordt de informatie verzonden (http, https, ftp, smb, file, ...)
domain name	domein	domein verwijst naar een server of een groep servers.
file path	path	verijst naar de file die de webserver moet 'uitvoeren'
parameters	variabelen	hiermee kun je informatie naar de webserver sturen.
anchor	verwijzing binnen de pagina	hiermee kun je naar een onderdeel van de pagina springen. Voorbeeld: https://www.roc.ovh/link/302#bkmrk-nog-een-keer-in-sche

Het schema bepaald hoe de informatie wordt opgevraagd (bij een webserver is dit meestal http en https). De domain name is de naam van de server(s). Het file path verteld de server welke informatie er moet worden getoond. De parameters zijn variabele die mee kunnen worden gegeven bijvoorbeeld om waarden die een gebruiker heeft ingevoerd naar de webserver te sturen. Tot slotte is het anchor (weinig gebruikt) een verwijzing naar een stukje binnen een pagina.

Normaal gesproken komt het file path overeen met de locatie van de bestanden op de server. Het file path is het path vanaf de document root. Stel de document root staat op jouw laptop naar:

c:\www

Je hebt XAMPP opgestart en je gaat naar:

http://localhost/ **opgave1/uitslag/index.php**

Dan wordt de file

c:\www\ **opgave1\uitslag\index.php**

door de webserver uitgevoerd. Het file path wordt dus achter de *document root* geplaatst

Hoe kom je van de URL in je browser naar de file op jouw laptop?

image-1613336806883.png

Met routing zorg je ervoor dat je ergens een vertaling wordt gemaakt tussen het URL en de file PHP die moet worden uitgevoerd.

Alles wat hierboven is beschreven is de *standaard* routing. Zo werkt het op jouw laptop als je plain PHP gebruikt (dat is PHP zonder framework).

Routing zorgt ervoor dat het path en eventueel de parameters de juiste code uitvoeren. Bij eenvoudige web applicaties is het path gewoon de locatie van de file op de harddisk, maar voor meer ingewikkelde programma's kan dat al snel onoverzichtelijk worden.

Met een framework werkt het daarom anders.

Routing in Yii

Standaard hoef je niets te doen voor routing. Er zit al heel veel voorgebakken in Yii.

Stel je typt in je browser in

localhost:8080/ country / index

Dit path bestaat uit twee delen; country en index .
country verwijst naar controllers/ Country Controller en
index verwijst naar de public function action Index in de class Country Controller.

Het path bestaat in Yii uit twee delen, de verwijzing naar de controller en de verwijzing naar de functie/method.

De vertaling van path naar controller en function/method gaat als volgt:

1. het eerste deel van het path is de controller-naam, maar
    1. de controller-naam begint altijd met een hoofdletter, en;
    2. als er een streepje (-) in het path staat dan is dat een nieuw woord en in de controller begint dit nieuwe woord met een hoofdletter. Voorbeeld: dit-is-het-eerste-deel wordt DitIsHetEersteDeel.
2. Het tweede deel van het path wordt op dezelfde manier vertaald maar er wordt ook nog eens het woord action voorgezet. Voorbeeld: dit-is-het-tweede-deel wordt actionDitIsHetTweedeDeel.

Een paar voorbeelden:

path	file	public function
/kaart/overzicht	KaartController	actionOverzicht
/klas-lokaal/overzicht	klasLokaalController	actionOverzicht
/klas-lokaal/stoel-maat	klasLokaalController	actionStoelMaat
/klas-lokaal/index	klasLokaalController	actionIndex
/klas-lokaal	klasLokaalController	actionIndex

In de laatste regel ontbreekt het tweede deel van het path. In dat geval wordt de actionIndex uitgevoerd.

Opdracht 1

We gaan in onze World-applicatie oefenen met routing.

(1) Zet in de CountryController class een nieuwe public function:

```
public function actionHello() {
    echo "Hello World!";
    exit;
}
```

Save de aangepaste CountryController en bedenk met welke url je nu deze nieuwe functie (method) moet aanroepen.

Controleer dat, zodat je in jouw browser het volgende ziet:

image-1594296881002.png

Opdracht 2

Maak een nieuwe file **controllers/ExampleController.php** en zet daar het volgende in.

```
<?php

namespace app\controllers;

use Yii;
use yii\web\Controller;



/**
 * CountryController implements the CRUD actions for country model.
 * Code by Ayoub
 */

class ExampleController extends Controller
{
  public function actionSay($message = '')
  {
    echo "Hello $message";
    exit;
  }
}
```

Vervang de naam `Ayoub` in de code 2X.

Ga nu naar http://localhost:8080/..... maar vervang de puntjes zodat je de functie (method) *actionSay* van de *ExampleController* uit laat voeren.

Extra uitdaging (optioneel)

Je kunt via de URL ook een parameter meegeven, zodat in plaats van "Hello Ayoub" er "Hello <jouw naam>" op het scherm komt.
Je hoeft hiervoor niets aan de code te veranderen. Je moet alleen de URL aanpassen, weet jij hoe?

Inleveren

1. Het bestand: controllers/CountryController.php
2. Het bestand: controllers/ExampleController.php
3. Screendump van het volledig scherm met browser waarop Hello gevolgd door jouw naam op het scherm verschijnt.

--

Verdieping (manual routing)

In de vorige les hebben een instelling in de config/web.php aangepast. Hierdoor hebben we eenvoudige routing aan gezet. Als je dit uit zet dan werkt de routing iets anders. Voor nu niet belangrijk maar als je wilt weten hoe het precies zit, kijk dan naar dit Engelse Youtube filmpje:

https://www.youtube.com/embed/QcZiS43iVxU

Verder is er nog meer informatie te vinden op de offici le Yii documentatie pagina's:

https://www.yiiframework.com/doc/guide/2.0/en/runtime-routing
--

04 Routing Quiz

05 Menu

In deze les gaan we routing zoals we dat in de vorige les hebben geleerd toepassen. We gaan een menu maken en we gaan de menu's koppelen aan de routing

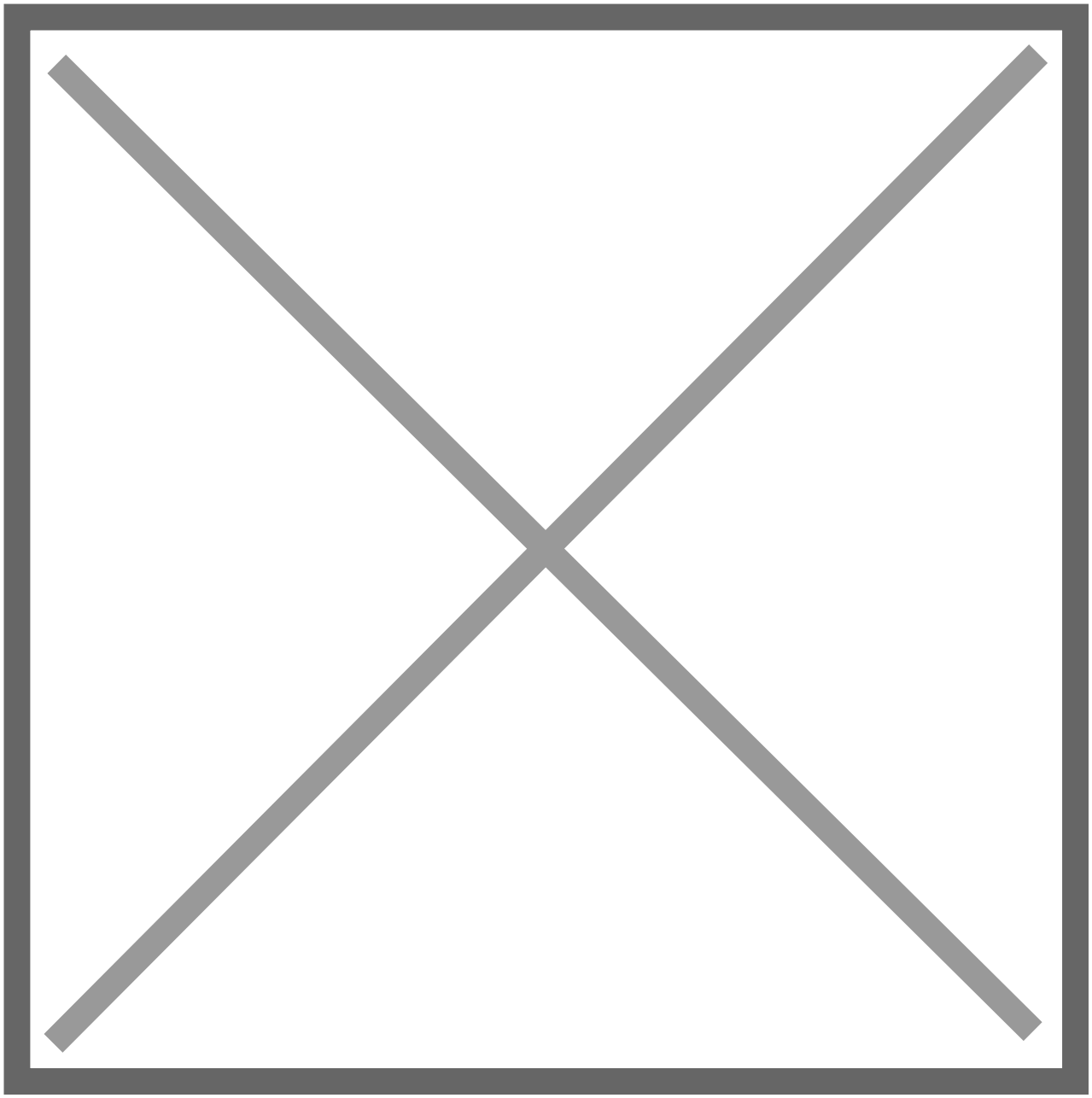
Views

In de folder views zien we drie folder staan:

1. **country**
   In de country folder staan de views die we met de CRUD-generator hebben gegeneerd.
2. **lay-outs,**
   hierin staat de standaard lay-out van elke Yii pagina.
3. **site**
   hierin staan een aantal standaard Yii pagina's zoals de *about* en de *contact* page.

Elke pagina die je laadt met Yii wordt gevormd door main.php in de lay-outs view.

In de main.php staat `<?= $content ?>` dit is kort voor `<?php echo $content ?>` . Het toont jouw pagina die in de variabele $content staat:



Haal de regel

```
<?= $content ?>
```

maar eens even tijdelijk weg en kijk wat er gebeurt!

Nav Widget in main.php

In de main.php in de folder lay-outs staat de standaard Yii pagina structuur. Het menu is hier ook een onderdeel van.

In de main.php staat een stuk PHP-code die gebruik maakt van een Yii widget. Een Yii widget is eigenlijk een plug-in zoals we die ook bij WordPress kennen.

De code voor de menu bar heeft de volgende structuur:

```
<?php
// plaats deze code in de main.php en vervang daarmee de standaard menu

    NavBar::begin([

     // hier wordt het type en de stijl van de menu bepaald
       'brandLabel' => Yii::$app->name,  // de naam van het menu
       'brandUrl' => Yii::$app->homeUrl, // de home page waar je naar toe gaat als je op de
naam klikt
       'options' => [
          'class' => 'navbar-expand-md navbar-dark bg-dark fixed-top',
        ],

   ]);



    echo Nav::widget([

      // hier worden de menu's en menu items bepaald
        'options' => ['class' => 'navbar-nav navbar-right'],
        'items' => [
            [ 'label' => 'Country',
                'items' => [
                    ['label' => 'Overzicht', 'url' => ['/country/index', ''] ],
                    ['label' => 'Voeg toe', 'url' => ['/country/index', ''] ],
                    ['label' => 'Europa', 'url' => ['/country/index',
'CountrySearch[Continent]' => 'Europe'] ],
                ],
            ],
        ],
    ]);

   NavBar::end();
 ?>
```

In Yii worden parameters voor Widgets via assiociative arrays mee gegeven. Deze hebben de structuur:

```
[ 'key1' => 'waarde1', 'key2' => 'waarde2' ]
```

de waarde kan dan ook weer een assiociative array zijn:

```
[ 'key1' => 'waarde1', 'key2' => [ 'andere_key1'=> 'andere_waarde1', ........ ]  ]
```

Opgaven

1. Pas het menu aan en plaats de code zoals hierboven beschreven:

   image-1613993167595.png

2. Om een country toe te voegen moet in de controller *CountryController* de method/function *actionCreate* () worden uitgevoerd.
   Bedenk hoe je de routing naar deze pagina opzet en pas het menu aan ' *voeg toe* ' zodat deze is gekoppeld aan *actionCreate* () van de *CountryController* .

3. In elk menu-item staat de route, bijvoorbeeld 'country/index' en daarna staat een ''.
   Deze laatste lege string zijn de parameters. Als we alle landen uit Europa willen filteren dan kunnen we de volgende URL gebruiken:
   image-1613993783698.png

   Het gedeelte na de ? is de parameter. In Yii zet je een paramter als volgt in een assiociatieve array (element): 'countrySearch[Continent]'=>'Europe'
   Het hele menu-item ziet er dus als volgt uit:
   image-1615820099350.png
   Pas je menu aan en test uit of alle drie de menu-items goed werken.

4. Maak nu een extra menu-item en noem dat Asia. Dit menu-item laat alle landen van Asia zien.

5. Maak nu een menu-item voor alle werelddelen: North America, South America, Asia, Europe, Africa, Antarctica en Oceania.

6. Maak een extra menu city met als items 'Overzicht' en 'Voeg toe'.

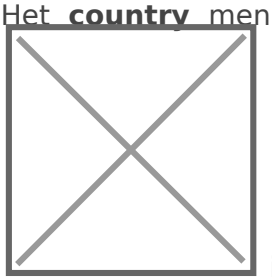
   image-1614009669414.png

Inleveren

Lever jouw views/layouts/main.php file in. Hierin staat als het goed is jouw menu (*Country* en *City*).

Check je werk goed!

Het **country** menu bevat **9** menu-items en het **City** menu bevat **2** menu-items.

 image-1614009669414.png

--

06 Grid View

In deze les leer je aanpassingen in de Gridview maken.

*Je leert hoe je kolommen wel of niet **zichtbaar** kan maken, hoe je de kolom **headers** kan aanpassen, hoe je de kolom **breedte** kan aanpassen, hoe je de **inhoud** van de kolom zelf kan aanpassen en als laatste leren hoe we een **link** kunnen maken van een kolom.*

Country overzicht aanpassen

Onze CRUD-generator (Gii) heeft voor ons een country overzicht gemaakt. Als je de CountryController.php opent dan zie je dat function (method) actionIndex de view index aanroept. Open deze index file. Deze staat in views/country.

Gridview Widget

Je ziet dat er op regel 23 een zogenaamde widget wordt aangeroepen.

Een Widget is een plug-in waarmee je functionaliteit aan Yii toevoegt. De Gridview Widget is misschien wel de meest uitgebreide widget en het zorgt ervoor dat we een mooi overzicht van onze data krijgen.

Opdracht

De opdracht bestaat uit 8 stappen. In deze stappen gaan we de country view aanpassen (views/country/index.php).
Test na elke stap of de aanpassingen werken!

Stap 1 - Kolommen aanpassen

In de parameters van deze widget zie je dat de kolommen *Code* , *Name* , *Continent* , *Region* en *SurfaceArea* worden afgedrukt. Je ziet ook dat bepaalde kolommen niet worden afgedrukt, deze staan in commentaar.

- Verander de lay-out in het country overzicht en laat de volgende kolommen in deze volgorde zien:
  Code , *Name* , *Population* , *Surface Area* en *Capital* .
  Population laat inwoneraantal zien en *SurfaceArea* laat oppervlakte zien.

Test of het werkt!

Stap 2 - Kopjes aanpassen

Stel we willen het kopje boven de kolom veranderen, dan kan dan als volgt:
Vervang bijvoorbeeld 'SurfaceArea' in de GridView door het volgende array.

```
[ 'label' => 'Oppervlakte',
  'attribute' => 'SurfaceArea',
],
```

Je ziet dat we de surface area nu hebben vervangen door een assiociative array. Dus waar in de code het woord 'SurfaceArea' staat plaats je het assiociative array zoals hierboven is aangegeven.

Dit verteld de GridView snippet dat het label Oppervlakte moet zijn (dit wordt boven aan de pagina gezet) en dat de waarde in de kolom moet worden gevuld met 'SurfaceArea'.

- Pas nu de kolom *Population* aan; en zet in de kolom header 'Inwoners'.

Test of het werkt!

Stap 3 - Kolom breedte aanpassen

We gaan de kolombreedte aanpassen. Vervang het blok van net door het volgende:

```
[ 'label' => 'Oppervlakte',
  'attribute' => 'SurfaceArea',
  'contentOptions' => ['style' => 'width:30px; white-space: normal;'],
],
```

- Pas de breedte van de kolom *Inwoners* (*Population*) ook aan; zet de breedte op 30 pixels.

Test of het werkt!

Stap 4 - Waarde van kolom aanpassen

Stel we willen km2 achter de oppervlakte plaatsen, dan kunnen we de waarde van de kolom als volgt aanpassen:

```
[
    'label' => 'Oppervlakte',
    'attribute' => 'SurfaceArea',
    'format' => 'raw',
    'value' => function($data) {
        return sprintf("%8d k&#13217", $data->SurfaceArea);
    }
]
```

Regel 5 t/m 7 in deze code verandert de waarde van de kolom. Dit is een functie en de return waarde van de functie wordt in de kolom afgedrukt.

Test of het werkt!

Als je achter return bijvoorbeeld "Hallo" zet dan wordt er in de kolom Oppervlakte overal Hallo afgedrukt. Probeer maar!

Stap 5 - Link maken van een kolom

In de Capital kolom zien we geen plaatsnaam, maar een nummer. Dit is het ID van een plaats in de city tabel.

Dus de *Capital* kolom uit *country* is de foreign key die wijst naar de primary key *ID* in *city.*

In de city view kun je een plaatsnaam zoeken door bijvoorbeeld de volgende URL te gebruiken:

http://localhost:8080/city/index?CitySearch[ID]=179

179 is het ID van Brussels, de hoofdstad van Belgi .

We gaan nu de value van de kolom Capital vervangen door de link.

Laten we om te beginnen de kolom *Capital* een beetje gaan aanpassen:

1. Vervang de kolomnaam van de kolom Capital door 'Hoofdstad'.

2. Verander de breedte van deze kolom naar 30px.

3. Vervang de value van de kolom in `'format' => 'raw', 'value' => function($data) { return "/city/index?CitySearch[ID]=179"; }` (alle kolommen krijgen dus dezelfde waarde).

Je ziet nu (in de browser):

image-1614025244735.png

Test of dit werkt!

Dat is nog geen link. Dat komt omdat we met ... een link moeten maken. Dat kan maar dat is lastig omdat we met " in " quotes zitten. Dus dubbele quotes in dubbele quotes. We kunnen een 'gewone' <a href> maken, maar het is een gepriegel.

Met de Yii module Html kunnen we dit eenvoudiger doen.

We vervangen de regel die de waarde returned in:

image-1614026085274.png

Je ziet nu in de browser een nummer (FK naar *city*) en als je daar op klikt, ga je naar de city *Bruxelles [Brussel]*

Test of dit werkt!

We zijn er bijna, we moeten nu alleen de 179 nog variabel maken, anders linkt elke regel naar Brussel.

image-1614026443516.png

Stap 6 - Testen en laatste aanpassing

1. Maak je code werkend met bovenstaande code.

2. Als je nu op het nummer klikt ga je naar de hoofdstad van het betreffende land.
   Test of dit werkt!

3. Verander nu de tekst van de link zodat je in de kolom Hoofdstad, in 'naar hoofdstad'.

4. Als je er op klikt dan ga je dus naar de index view van de hoofdstad.

5. Verander ook de breedte van de kolom naar 200px zodat de kolom netjes past.

De kolom ziet er dus als volgt uit:

image-1614026822965.png

Stap 7, kolommen aanpassen

1. Zet de kolommen in de volgende volgorde:

Code - Name - Hoofdstad - Inwoners- Oppervlakte

2. Verander de kolomnamen als volgt:
   Code - Naam - Hoofdstad - Inwoners - Oppervlakte

3. Zorg dat de naam van het land vet gedrukt wordt (bold).

4. Haal vervolgens de volgnummers (1,2,3,4....) in de eerste kolom weg

5. Verplaats de edit kolom (de laatste) naar voren.

Het hele overzicht ziet er uit zoals te zien is op het volgende plaatje:

image-1617127616569.png

Stap 8 - bevolingsdichtheid

De bevolkingsdichtheid zegt iets over de hoeveelheid inwoners in een land. Natuurlijk heeft Amerika meer inwoners dan Nederland, maar het land is ook vele groter. Om te vergelijken hoe dichtbevolkt een land is, kun je uitrekenen hoeveel mensen er gemiddeld per km2 (dit een gebied van 1000x1000 meter) wonen.

bevolkingsdichtheid = aantal inwoners / oppervlakte.

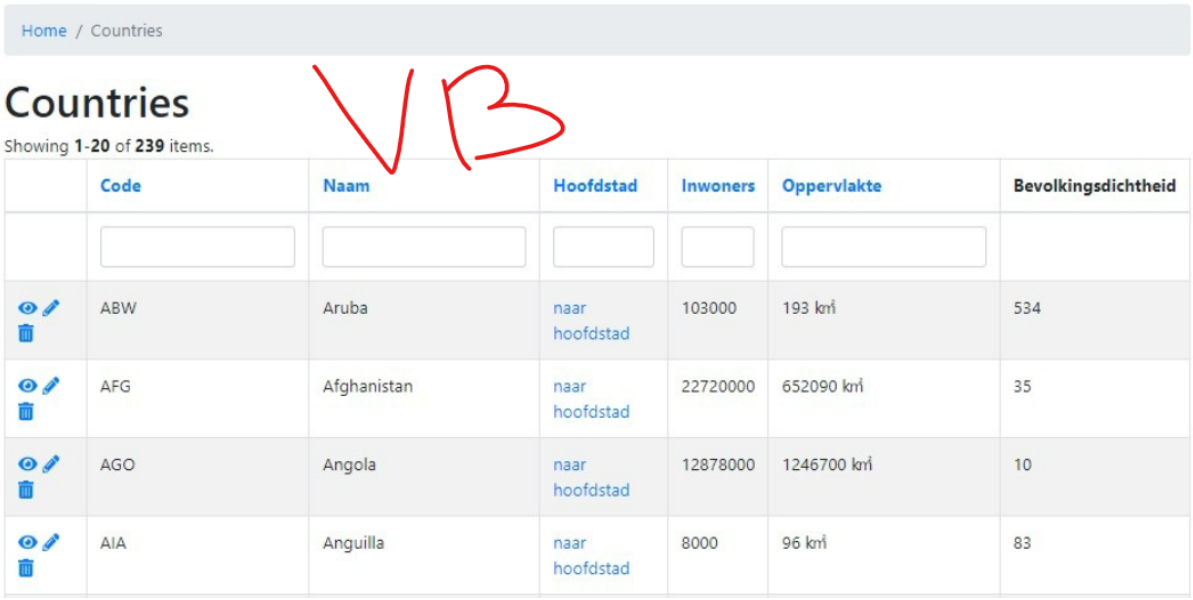
Stel een land heeft een oppervlakte van 100 km2 en je er wonen 200 mensen. Dan heb je in dit land dus gemiddeld 2 inwoners per km2, want:

2 = 200 / 100

- Maak nu een nieuwe kolom in het *country* overzicht en bereken daarin de bevolkingsdichtheid.

- Gebruik de php functie *round()* om de bevolkingsdichtheid met 0 decimalen af te drukken.

Aan het einde heb je jouw *gridview* aangepast en ziet het er zo uit:



Het Eindresultaat

Inleveren

1. Maak een **schermafdruk** van jouw Country's aangepaste *Gridview* en lever deze in.
2. Stuur ook jouw (aangepaste) **country view** in.

--

Yii 2

Debug ready

Opdracht debugging

Om straks de opgave goed te kunnen maken, gaan we een eenvoudige debug functie maken.

Easy Debugging

Als je de aanpassingen in dit onderdeel uitvoert, dan kun je later in de code de volgende twee functies gebruiken:

`_d($objectnaam)` bekijk $object de code loopt gewoon door.
`_dd($objectnaam)` bekijk $object de uitvoering van de code stopt.

Hoe krijg je dit? Eenvoudig twee stappen.

Stap 1 , New file config/functions.php

```
<?php
/**
 * Debug function
 * _d($var);
 */
function _d($var,$caller=null)
{
    if(!isset($caller)){
        $caller = debug_backtrace(1)[0];
    }
    echo '<code>Line: '.$caller['line'].'<br>';
    echo 'File: '.$caller['file'].'</code>';
    echo '<pre>';
    yii\helpers\VarDumper::dump($var, 10, true);
    echo '</pre>';
}
```

```
/**
 * Debug function with die() after
 * _dd($var);
 */
function _dd($var)
{
    $caller = debug_backtrace(1)[0];
    _d($var,$caller);
    die();
}
```

Stap 2 , Edit config/web.php - voeg regel 3 en 4 toe.

```
<?php

/* Include debug functions */
require_once(__DIR__.'/functions.php');
```

Inleveren

Jouw aangepaste functions.php

--

Version control, GIT en VSC

Als je met een framework werkt, pas je vaak meerdere bestanden aan. Soms maak je fouten en weet je even niet meer waarom het opeens niet meer werkt. Dan is het handig als je even naar de vorige stap terug kan.

Dat kan als je *version control* toepast. In VCS kun je GIT daarvoor gebruiken.

In de module DevOps is uitgelegd hoe dit werkt. Als deze module niet beschikbaar is, dan loop je wat voor, en zul je zelf even moeten opzoeken hoe je GIT en VSC kan gebruiken.

Als GIT goed is ge nstalleerd, dan zie je het volgende in jouw VSC-editor.

Screenshot 2023-02-13 171251.png

Bij het rode rondje zie je de wijzigingen die je hebt gemaakt.

Bij het oranje rondje kun je een korte versienaam intypen en met commit, kun je deze versie dan 'commiten'.

Inleveren

Laat zien dat GIT werkt in je browser door een scherm afdruk van je gehele VSC in te leven.

Yii Refresh

Grid view (2)

Zorg ervoor dat jouw Yii World project uit Yii Level 1 weer (nog) draait

In deze opdracht hebben we een refresh en gaan we nog een keer kijken naar de standaard Yii gridview.

Wat leren we?

- We frissen onze kennis weer wat op en leren met het MVC model te werken.
- Hoe we de standaard Yii grid view kunnen tweaken (aanpassen).
- Hoe je in PHP een af kan ronden.
- Hoe je de vormgeving met CSS aan kunt passen.

Checklist

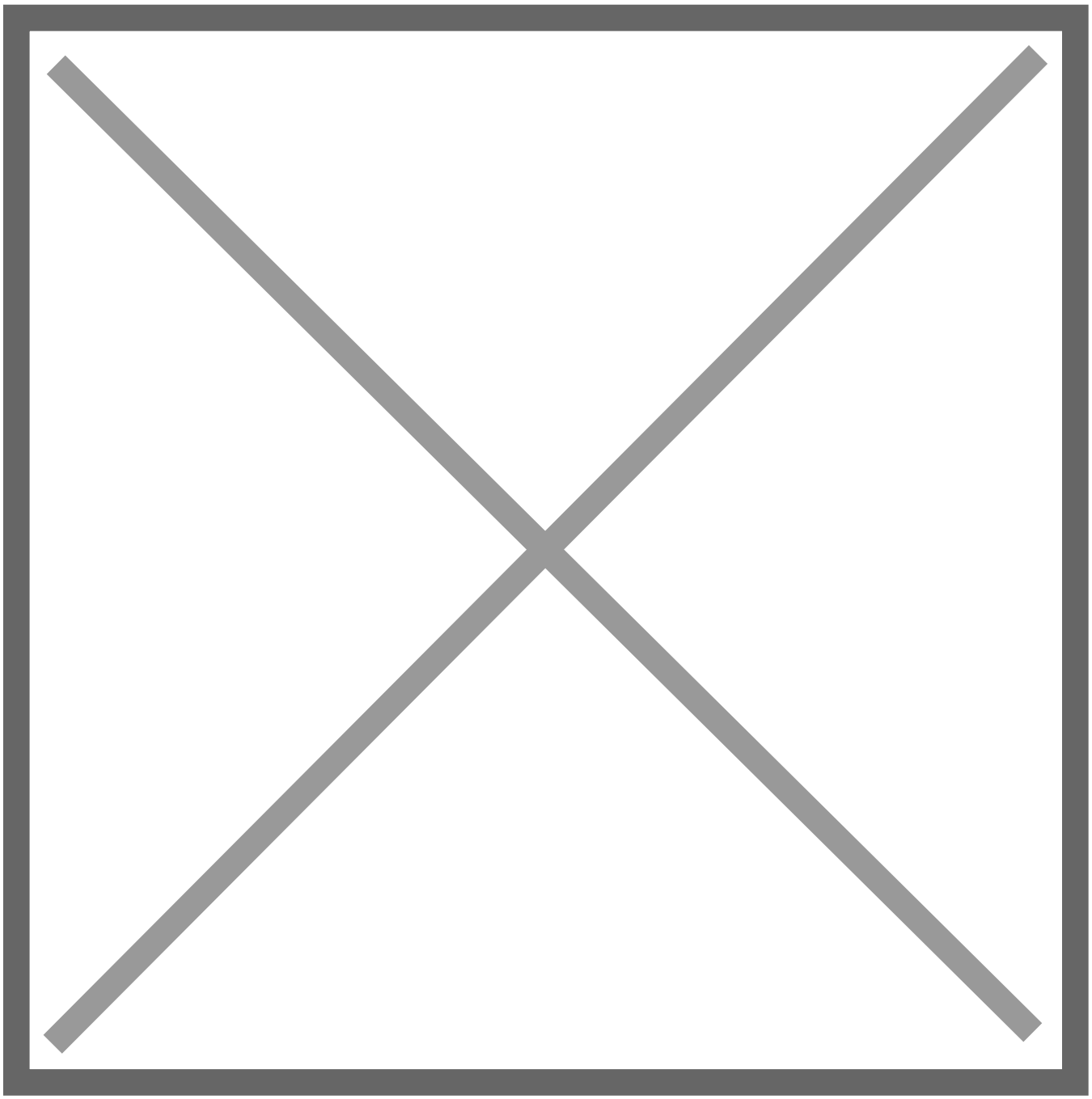
1. Controleer of je de views Country, City en CountryLanguage nog hebt. Heb je de models n de controllers?
2. Open jouw Yii project in VSC

3. Controleer de map models, heb je de 6 bestanden: City.php, CitySearch.php, Country.php, CountrySearch.php, CountryLanguage.php en CountryLanguageSearch.php?
   Als je deze bestanden niet hebt, ga dan terug naar Yii deel 1 en doorloop nog een keer alle stappen.
4. Type `php yii serve` in het terminal window in VCS in en controleer op localhost:8080/country of you website het nog doet.

OK all fine, let's go!

Om er weer in te komen gaan we onze country view aanpassen.

Dit moet er uit komen:



In stap 1 is dit wat er moet uitkomen. Controleer de kolommen. Heb je deze kolommen al? Dat kan want de laatste opgave van Yii Level 1 ging ook over dit onderwerp. Lees de opgave toch even door want je moet het overzicht precies zo maken als in het voorbeeld.

Stap 1 - kolomnamen naar rechts uitlijnen

De laatste drie kolommen zijn naar rechts uitgelijnd.

De header kan je met de volgende code naar rechts uitlijnen.

'headerOptions' => ['style' => 'text-align:right;'],

Dit voeg je toe aan het array waarin je de kolom in de Gridview.
Dat is het hele code blok dat tussen [en [staat.

Stap 2 - getallen naar rechts uitlijnen

Zorg ervoor dat de inhoud van de kolommen ook naar rechts uitlijnt. Voeg hiervoor de juiste CSS toe aan het codeblokje dat over de kolom gaat.
Weet je niet meer hoe het werkt, kijk dan nog een keer hoe je dit in Yii Level hebt gedaan.

Stap 3 - bevolkingsdichtheid 2 decimalen

Bevolkingsdichtheid rond je af op 2 decimalen (is nu 0).

Stap 4 - inwoneraantal met " *thousand seperator* "

In de kolom Inwoners formatteer je het getal zodat er een spatie tussen de getallen komt. Dit is de zognaamde "*thousand seperator*". Gebruik hiervoor de PHP functie "*number_format*".

Stap 5 - 0 inwoners wordt onbewoond

Landen als *Antarctica* hebben 0 inwoners. Vervang de 0 door " *Onbewoond* ".

Tip: met "if ($data->Population == 0) { ..." kan je testen of het aantal inwoners 0 is.

Stap 6 - Kolom een kleur geven

De kleur van het font van de kolom Naam wordt donkerblauw (CSS *darkblue*). Zie voorbeeld.

Het font blijft wel bold.

Check of alles goed werkt!

Inleveren

1. Screenshot van je aangepaste city view zoals het voorbeeld hierboven. Laat een compleet screenshot zien zodat ook je datum en tijd rechtsonder in het beeld is te zien.
2. Jouw aangepaste code (*view/country/index.ph* p).

Menu (oops)

Inleiding

We gaan een eigen overzicht maken. We beginnen met het aanmaken van een menu, dan een controller en daarna ook nog een view. We leren data op te halen vanuit de controller en tonen die vervolgens onze eigen view.

Het ophalen van data doen we met een eenvoudige query die we in de Controller plaatsen. Vanuit de Controller gaan we via het Model naar de database. De volgorde wordt:

Menu

Om te beginnen voegen we een nieuwe methode (=functie) toe aan de **controllers/CountryControlle** r class. De methode geeft voor nu alleen "oops" terug en doet verder nog even niets. We noemen de methode actionOverzicht, door de hem zo te noemen weet Yii dat het deze functie aan moet roepen als de gebruiker de url ***/country/overzicht*** aanroept. Het koppelen van een url aan een stuk logica, zoals een functie of methode noemen we **routing** .

Opdracht

Maak een menu item *overzicht* onder country dat naar */country/overzicht* gaat.

In Yii level 1 hebben we met menu's geoefend.

image-1614620329168.png

Test het menu uit.

Het menu gaat naar de actionOverzicht in de CountryController en returned "oops". We krijgen dus een leeg scherm met "oops".

Inleveren

- Een schermafdruk van jouw gehele browser (url moet zichtbaar zijn) waarin je het via het menu naar jouw overzicht gaat en waar "oops" wordt afgedrukt.

Controller

We gaan in onze nieuwe controller functie actionOverzicht code plaatsen, deze code haalt alle data van alle countries op en roept vervolgens de view overzicht aan.

Controller

We halen onze "oops" weg en gaan dus een stukje code toevoegen aan onze CountryController.

Let op zet jouw naam op regel 3.

```
public function actionOverzicht()
{
    // Zet hier jouw naam.
    // Dit is de query, dit is te vergelijken met select * from Country
    $countries = Country::find()->all();
```

```
    // De code hieronder genereert de html, dit noemt men de view binnen MVC (model view
controller).
    // Zoals je kunt zien geven we de $countries array door aan de "overzicht" view.
    // In de overzicht View kunnen we nu $countries gebruiken.

    return $this->render('overzicht', [
        'countries' => $countries,
    ]);
}
```

Opdracht

Maak de function *actionOverzicht* in de *CountryController* zoals hierboven is aangegeven.

Als je alle stappen goed hebt doorlopen krijg je als deze code aanroept een foutmelding.

image-1614623472969.png What? Een fout? Waarom nu?

Leg uit waarom je een foutmelding krijgt.

Inleveren

1. Uitleg waarom je een foutmelding krijgt.
   (maak een .txt bestand met notepad en zet daarin de uitleg)
2. CountryController.php

View

View

Het menu 'overzicht' gaat naar de *actionOverzicht* deze voert een query uit en roept de view overzicht op. Dit overzicht gaan we nu maken.

We gaan dus in de folder views/country een nieuwe file overzicht.php aanmaken en plaatsen daar de volgende code in.

```
<?php
    foreach ($countries as $country) {
        echo $country->Name;
        echo " - ";
        echo $country->Code;
        echo " - ";
        echo number_format($country->Population, 0, ',', ' ');
        echo "<br>";
    }
?>
```

Opdracht

Maak de view *overzicht* zoals hierboven is aangegeven.

Het overzicht is bijna niet geformatteerd.

image-1614627165703.png

Weet je nog hoe een HTML table er uit ziet?

```
<table>

<tr>
    <td> .. </td>
        <td> .. </td>
        <td> .. </td>
     </tr>

     <tr>
    <td> .. </td>
        <td> .. </td>
        <td> .. </td>
     </tr>

</table>
```

Hierboven staat een skelet van een table met twee rijen en drie kolommen.

Vind je dit lastig? Kijk dan de video.

Verander de /view/country/overzicht nu zo dat het netjes in een table wordt gezet. De output komt er als volgt uit te zien:

image-1614627534845.png

De kolommen staan nu dus netjes onder elkaar.

Je hebt nu zelf een view opgebouwd zonder gebruik te maken van de Gridview widget. Het kost meer tijd en moeite om zelf een overzicht zonder Gridview widget te maken, maar je hebt wel veel meer controle over hoe jouw overzicht er uit komt te zien.

Inleveren

- Een screenshot van je browser (met URL) met daarin het overzicht dat je hebt gemaakt en
- de aangepaste en werkende view/country/overzicht.php

--

Europa

We hebben een eigen overzicht gemaakt. Dat gaan we nog een keer doen en nu met alle landen van Europa, een selectie dus.

Opdracht

- Maar een nieuw menu-item en noem dit " *Overzicht Europe*" .
- Maak een nieuwe function *actionOverzichtEurope,* zet deze in de juiste controller en koppel deze aan het menu-item.

  (let op de vertaling naar de routering; in de routering staat nooit een hoofdletter en woorden worden gescheiden door een streepje "-" , zie les 2 over routing).

- Laat in dit overzicht alle landen van Europa zien. Op de vorige pagina staat uitgelegd hoe dit moet.

- Druk de volgende kolommen af:
  Name en *Surface Area*

- Zet de twee kolommen netjes onder elkaar (in table)

- Formateer de Surface Aera netjes met spatie tussen de duizendtallen en lijn ze recht uit.

- Maak de headers **bold** .

- Sorteer het overzicht op Surface Area van grootste land naar kleinste.

  Het overzicht komt er dus zo uit te zien:

image-1614629726011.png

Je hebt geleerd dat je een overzicht kan maken via een (Yii-)query. Met de query kan je de selectie en sortering aanpassingen. De query wordt geplaatst in de *controller* en maakt gebruik van het model. In dit voorbeeld het model van *country* .

Inleveren

- Jouw aangepaste en werkende view/country/overzicht.php in tabel vorm, en
- Een schermafdruk yii-04a-jouw-naam van jouw gehele browser waarin je je overzicht laat zien.

Selectie (SurfaceArea < 100000)

Opdracht

- Pas het overzicht van de vorige opdracht aan zodat je alleen alle kleine Europese landen ziet.

- Een land is klein als `SurfaceArea < 100000`
  Pas hiervoor de controller aan.

- Je hebt dus te maken met **twee** zoekcriteria.

  Zoek zelf op internet uit hoe je in een 'Yii-query' twee zoekcriteria kan combineren met *andWhere()* of gebruik de query methode om met Yii data uit de database te halen.

Inleveren

- Jouw aangepaste en werkende countryController.php

Hoofdstad

Opdracht

Lees de uitleg (op de vorige pagina) goed door en plaats de kolom hoofdstad in het overzicht.

image-1615887892142.png

Maak hiervoor twee aanpassingen:

1. maak de relatie in het model van *Country* , en
2. pas de view *overzicht* aan door daar de kolom *Hoofdstad* aan toe te voegen.

Inleveren

1. Een schermafdruk yii-05a-<jouw naam> van je country view waarin de kolommen Naam, Hoofdstad en Oppervlakte worden afgedrukt, zoals in het voorbeeld hierboven. Maak een afdruk van je gehele browserscherm.

Clickable hoofdstad

We gaan de Hoofdstad clickable maken. Als je naar http://localhost:8080/city gaat dan zie je het overzicht van steden. Klik op een oogje op een van de steden. Let op de URL. Je ziet http://localhost:8080/city/view?id=<NR> als url en <NR> is het ID van de stad.

Dus als je bijvoorbeeld klikt op http://localhost:8080/city/view?id=5 dan zie je informatie over Amsterdam.

Opdracht

Maak elke hoofdstad in het overzicht dat je bij opdracht 1 hebt gemaakt *clickable. Als je op de link klikt dan* open je het overzicht van de stad, bijvoorbeeld http://localhost:8080/city/view?id=5 .

Tip: je kunt Html:a() gebruiken om een link te maken. Dit is in de vorige les uitgelegd .

image-1615896595590.png

Inleveren

1. Jouw aangepaste country view. Plaats in je country view commentaar met jouw naam.

Eén taal

Opdracht

Lees de uitleg hierboven goed door. Druk in de laatst kolom n taal af.

image-1616504705123.png

Nu hebben we n taal per land, maar in de meeste landen wordt meer dan n taal gesproken. Daar gaan we in de volgende opdracht naar kijken.

Let op: je hebt in je model een query staan die alleen alle landen uit Europa selecteert. Zorg dat je hierbij alle kolommen selecteert, omdat je anders de kolommen mist die je nodig hebt om de link (join) te maken.

Dus dit is onjuist:
image.png

In dit voorbeeld selecteer je maar drie kolommen (Name, SurfaceArea en Capital) en de kolom Code waarmee je de join naar de tabel Countrylanguages moet maken zit er niet bij. Je voegt dus de juiste kolommen toe of je aat de hele select regel weg zodat alle kolommen worden geselecteerd.

Inleveren

Een schermafdruk yii-06a-jouw-naam met het scherm van jouw gehele browser waarin je het landenoverzicht laat zien en waarbij je n taal afdrukt.

Meer talen

Om alle talen die bij n land horen af te drukken, gaan we met een loop door het array $country->countrylanguages heen.

Gebruik hiervoor de volgende loop.

```
foreach( $country->countrylanguages as $taal) {
// vul zelf de juiste variable in om de taal af te drukken
    echo .....;
echo " <br/>";
}
```

Opdracht

Gebruik de loop zoals hierboven is aangegeven, vul deze aan en maak het volgende overzicht.

image-1616505080100.png

Zorg ervoor (met CSS) dat de tabel er netjes uitgelijnd uit ziet zoals hierboven in het voorbeeld.

Je hebt nu een 1:N (n op meer) relatie gemaakt. In volgende lessen gaan we hier opnieuw mee aan de slag.

Inleveren

- Een schermafdruk yii-06b-jouw-naam.png waarin je je complete scherm laat zien waarin het landenoverzicht te zien is met alle talen zoals in het voorbeeld.

Percentage

Opdracht

In de Countrylanguages tabel staat naast de taal ook het percentage. Dit is het percentage van de bevolking dat deze taal spreekt. Zet dit percentage tussen haakjes achter elke taal. Voorbeeld:

image-1616505413868.png

Inleveren

1. - Een schermafdruk yii-06c-jouw-naam.png waarin je je complete scherm laat zien waarin het landenoverzicht te zien is met talen en percentages zoals in het voorbeeld.

Voorbeeld

Screenshot 2022-12-25 101520.png

Sorteren

Opdracht

Sorteer de talen op percentage aflopend van hoog naar laag. Op die manier komt de meest gesproken van een land bovenaan.

image-1616506111435.png

Tip: gebruik Google om te zoeken hoe je de relatie in het model verandert zodat je **aflopend** kunt sorteren.

Inleveren

1. Een schermafdruk yii-06d-jouw-naam.png waarin je je complete scherm laat zien waarin het landenoverzicht te zien is zoals in het voorbeeld.
2. De code van de view van country. Voeg in jouw code commentaar toe waarin je jouw naam zet.

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)

image-1616594908052.png

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

image-1616594707813.png

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

image-1616595011664.png

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.

image-1616594739223.png

Bestelling

Maak nu de tabel bestelling in de database.

image-1616595477669.png

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:

image-1616595571596.png

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.

Schermafdruk 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 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 assiociative 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 assiociative 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 assiociative 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.

image-1616605065315.png

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

Inleveren

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

image-1616605448841.png

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 bedenk 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 assiociative 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. schermafdruk yii-09-jouw-naam met het form waarin je de drop down van de producten (opengeklapt) laat zien. Maak een schermafdruk 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?

image-1616610519616.png

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:

image-1616611684520.png

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

How to change label text of the ActiveField?

Inleveren

1. schermafdruk yii-010b-jouw-naam met het bestellingen form waarin je de aangepaste labels laat zien. Maak een schermafdruk 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 'ís 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. schermafdruk yii-011a-jouw-naam met de bestellingen index view waarin je laat zien dat de relatie is gemaakt met de tabel medewerker. Maak een schermafdruk 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:

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* .

Inleveren

1. schermafdruk 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 schermafdruk 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.

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!

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.

--

Yii Challange

Yii Challenge

Weet je nog dat je een CRUD opdracht hebt gemaakt voor te-laat-komers.

Het voorbeeld staat hier:

https://stampwerk.nl

image-1655803735254.png

Maak een database en maak de CRUD in yii.

Probeer het overzicht zo goed mogelijk na te maken.

De eisen:

1. Noem de database *telaat* , voor de tabelnaam mag je zelf een naam verzinnen.
2. De kolomnamen moeten overeenkomen met het voorbeeld.
3. De knop *weer eentje te laat* ! moet ook onder aan de pagina staan.
4. De statistieken moeten overeen komen met het voorbeeld.
5. ID's dienen automatisch te worden gegenereerd (dus geen ID's in forms).

De statistieken

Om dit deel te kunnen maken passen moeten we twee zaken aanpassen: de index.php in de view en de controller.

1. We passen eerst de view aan en laten voorlopig nog vaste (statische) waarden zien.
2. Daarna zorgen we ervoor dat er waarden vanuit de controller aan de view worden doorgegeven. We gebruiken hiervoor nog steeds vaste waarden, maar nu worden ze wel in de controller gezet.
3. Als laatse stap voegen we in de controller een query toe en halen de waarden uit de database. De waarden zijn hiermee dynamisch geworden.

De stappen worden hieronder uitgelegd. Je krijgt hier en daar stukjes voorbeeld code, maar je moet de code wel op de juiste manier aanpassen en elke stap goed testen.

Stap 1, aanpassen view (statisch)

We beginnen eerst met de view. Pas de view aan zodat deze er komt uit te zien zoals in het voorbeeld. Neem voor de waarden voorlopig even vaste waarden die (nog) niet uit de database komen. Vul bijvoorbeeld op elke regel 999 als waarde in. Als je tevreden bent met de opmaak dan gaan we de data uit de database halen.

Stap 2, aanpassen controller met statisch waarden

In de controller staat een method/function *actionIndex().* Dit is waar de gegevens worden opgehaald die in de view worden getoond. Op de laatste regel van de actionIndex wordt de view aangeroepen en worden de gegevens vanuit de controller naar de view gestuurd.

```
return $this->render('index', [
    'searchModel' => $searchModel,
    'dataProvider' => $dataProvider,
]);
```

Wij gaan hier gegevens aan toevoegen.

```
return $this->render('index', [
    'searchModel' => $searchModel,
    'dataProvider' => $dataProvider,
 'hoogste' => $hoogste,
    'gemiddelde' => $gemiddelde,
    'totaal' => $totaal,
]);
```

Geef de drie variabelen elk een vaste waarde, bijvoorbeeld.

```
$hoogste=111;
$gemiddelde=222;
$totaal=333;
```

Zet deze code ook in de *actionIndex* en pas de view aan zodat je deze waarden (nog steeds vaste) waarden ziet. Je ziet in de view dus niet meer 999 maar 111, 222 en 333.

Als dit werkt dan heb je ervoor gezorgd dat de waarden die in de controller (actionIndex) worden getoond in de view. Top!

Stap 3, aanpassen controller met dynamische waarden

Nu moeten we de drie waarden uit de database halen. Dat gaat in Yii als volgt.

```
$sql="select max() as hoogst, avg() as gemiddeld, sum() as totaal from ....";
$result = Yii::$app->db->createCommand($sql)->queryOne();
```

Op regel 1 maak je de query af zodat de drie waarden worden berekend.

Tip: Controleer je query eerst in phpmyadmin of die werkt. Pas als de query werkt zet je hem in de controller.

In de query worden aliases gebruikt (hoogst, gemiddeld en totaal), dit ....

Controleer met

```
dd($results);
```

of de query het gewenste resultaat geeft.

Als het goed is heb je nu de gegevens, bijvoorbeeld:

```
$hoogste=$results['hoogste'];
```

Het resultaat van de query wordt nu in de variabele $hoogste gezet. Doe dit met alle drie de variabelen.

Als alles werkt zijn de waarden van de drie variabelen (hoogste, gemiddelde en totaal) nu dynamisch. Ze komen uit de database en worden vanuit de controller gevuld en aan de view doorgegeven.

Top! Je hebt nu helemaal van scratch een volledig CRUD gemaakt. Weet je nog dat je deze CRUD ook in 'vanilla' PHP had gemaakt? je hebt nu hetzelfde met Yii gemaakt. En wat vind je makkelijker?

Evaluatie

Maak een kort documentje waarin je twee voordelen en twee nadelen van het gebruik van Yii ten opzichte van 'vanilla' PHP (php zonder framework) beschrijft.

Onderbouw de voor- en nadelen aan de hand van de opdrachten die jij hebt gemaakt. Maak dus niet een algemeen verhaal, maar wees specifiek waar jij tegenaan gelopen bent of wat jij juist heel handig vond. Benoem de opdracht als onderbouwing.

ChatGPT mag je gebruiken zolang je maar aan de bovenstaande voorwaarde voldoet.

Beschrijf verder welke twee tips jij studenten wilt geven die nog aan de Yii module moeten beginnen.

Je kunt onderstaande template gebruiken.

Evaluatie Yii jouw-naam.docx

Noem het documentje evaluatie-jouw-naam.pdf

Inleveren

Let op je moet vier documenten inleveren: 1 screenshot, 2 php files en een documentje.

1. Screenshot yii-10-jouwnaam met de index view waarbij je de te laat meldingen en de statistieken laat zien. Maak een screenshot van jouw complete browser scherm.
2. De aangepaste controller en plaats in de code commentaar. Leg uit wat de regels die jij hebt toegevoegd doen.
3. De aangepaste view.
4. De evaluatie. Noem het documentje evaluatie-jouw-naam.pdf

Eindgesprek Yii Challenge

Heb je alle Yii opdrachten gedaan en heb je de challenge ingeleverd dan kan je een eindgesprek aanvragen bij een docent.

In het eindgesprek wordt gecontroleerd of je voldoende hebt geleerd.

Vragen die (bijvoorbeeld) je kunt verwachten:

- wat vond je lastig aan Yii?
- hoe ben je aan je antwoorden gekomen?
- kun je uitleggen wat routing is en hoe dat in Yii is vormgegeven?
- kun je uitleggen wat MVC is of kun je hiervan voorbeelden geven aan de hand van je code?
- wat is de Yii grid view en waarom zou je dat wel (of juist niet) gebruiken?
- geef een demo van jouw Challange.
- Stel je wilt een de kleur van iets aanpassen, waar doe je dat in de code?

Zorg dat je de antwoorden weet op de bovenstaanden vragen en dat je in code kan aanwijzen waar bepaalde onderdelen staan.

--