

Distelvinklaan 22

2660 Hoboken

+32 3 830 4105

[www.cvoAntwerpen.be](file:///E:\Bibliotheken\MIJN%20DOCUMENTEN\Opleiding%20Informatica\63F\A3%20Communicatie%20en%20organisatietechnieken\2012-2013\www.cvoAntwerpen.be)

[info@cvoantwerpen.be](mailto:info@cvoantwerpen.be)

**HBO Informatica**

**Module B3 Programmeren 4**

**TV 11727**

**Projectdossier**

**Instant Interview**

**Dieter Benoot**

**Cedric Jacobs**

**David Vlaminck**

Academiejaar 2014 – 2015

# Inhoudsopgave

[1 Inhoudsopgave 3](#_Toc421467319)

[2 Inleiding 5](#_Toc421467320)

[3 Projectvereisten 6](#_Toc421467321)

[3.1 Concept 6](#_Toc421467322)

[3.2 Concreet 6](#_Toc421467323)

[3.3 Gebruikte concepten en frameworks 6](#_Toc421467324)

[4 Analyse 7](#_Toc421467325)

[4.1 ERD 7](#_Toc421467326)

[4.2 Use cases 7](#_Toc421467327)

[4.2.1 Maken van een pdf na het selecteren van een functie 7](#_Toc421467328)

[4.2.2 Een nieuwe vraag toevoegen 8](#_Toc421467329)

[5 Indeling van het project 9](#_Toc421467330)

[5.1 Mappenstructuur 9](#_Toc421467331)

[5.2 Map app 9](#_Toc421467332)

[5.3 Map BLL 9](#_Toc421467333)

[5.4 Map DAL 9](#_Toc421467334)

[5.5 Map temp 9](#_Toc421467335)

[5.6 Map vendor 10](#_Toc421467336)

[5.7 Map web 10](#_Toc421467337)

[5.8 Composer & namespaces 10](#_Toc421467338)

[5.8.1 Aanpassing aan autoload\_real.php 10](#_Toc421467339)

[5.8.2 Toevoeging van autoload\_II.php 10](#_Toc421467340)

[6 Opbouw 11](#_Toc421467341)

[6.1 Index.php en Javascript 11](#_Toc421467342)

[6.2 MVC 11](#_Toc421467343)

[6.3 BLL 11](#_Toc421467344)

[6.4 DAL 11](#_Toc421467345)

[7 Uitbreiding 12](#_Toc421467346)

[7.1 Uitbreiding blijft mogelijk 12](#_Toc421467347)

[7.2 Use case diagram 12](#_Toc421467348)

# Inleiding

Dit document is het dossier van het project Instant Interview. Dit project is het resultaat van de opdracht die tijdens het vak Programmeren 4 (PHP) gegeven is. In dit document worden de verschillende aspecten van dit project beschreven en gedocumenteerd.

Eerst wordt een beschrijving gemaakt van de projectvereisten. Daarna volgt de analyse die ons bij dit project geholpen heeft. Vervolgens wordt de structuur van de code uitgelegd en tot slot dieper ingegaan op de concepten en hoe dit in de code uitgewerkt is.

Notities:

1. Presentatie
   1. Je stelt voor hoe je je project hebt opgebouwd waarbij je uitlegt hoe je OOP, n-tier, MVC en Tests hebt toegepast.
   2. Je geeft een demo.

# Projectvereisten

## Concept

Het concept Instant Interview werd bedacht door Cedric. Hij doet interviews met kandidaten voor een sollicitatie. Hierbij gebruikt hij een lijst met vragen die gerangschikt staan per competentie in een Excel lijst. Verder heeft hij ook manier om voor een bepaalde functie een vaste lijst vragen te laten opstellen. Maar deze manier van werken is weinig overzichtelijk, niet gebruiksvriendelijk en moeilijk te onderhouden. Bovendien wil hij dit eventueel op termijn verder uitbouwen naar een online formulier waar hij ook meteen de antwoorden van de kandidaat kan invullen tijdens het interview. Daarom hebben wij, Cedric, Dieter en ik, een deel van dit concept uitgewerkt als project voor dit vak.

## Concreet

Dit project is een PHP project met een MySQL database. Sommige van de keuzes die hier gemaakt zijn, zijn omwille van de mogelijkheden of beperkingen die daar bij horen. Zo is beslist om dit project een single page website te maken. Aangezien de gebruiker vragen zal aanvinken of keuzes maken is het niet wenselijk veel van pagina te veranderen.

De gebruiker kan naar de verschillende delen op de website gaan, maar blijft op 1 pagina. Het Instant Interview gedeelte zelf wordt ook binnen deze ene pagina opgebouwd. De gebruiker krijgt een keuzelijst te zien waaruit een functie kan geselecteerd worden. Wanneer de gebruikt een keuze maakt, worden verschillende competenties weergeven die bij deze functie horen. Er worden al een aantal vragen aangeduid, maar de gebruiker is vrij om meer vragen aan te duiden. De vragen worden zichtbaar telkens er op een competentie geklikt wordt.

De gebruiker kan alle competenties zien door op een knop te klikken. Wanneer hij op de “Rapport” knop klikt, verschijnt in een kleiner bovenliggend venster een overzicht van de vragen die aangeduid zijn. Wanneer in dat venster op de “Download PDF” knop wordt geklikt, genereert de server een pdf met de aangeduide vragen en zal de browser dit bestand downloaden.

De laatste mogelijkheid die de gebruiker heeft, is het toevoegen van een vraag. Hiervoor klikt hij op de “Voeg vraag toe” knop die een kleiner venster opent. Hier kan men een competentie uit een keuzelijst selecteren en een vraag in een tekstvak typen. De vraag wordt dan aan de geselecteerd competentie toegevoegd. Hiervoor wordt de pagina wel vernieuwd en verdwijnt de selectie van vragen.

## Gebruikte concepten en frameworks

De volgende zaken worden in dit project gebruikt:

* Bootstrap
* jQuery
* FPDF (netasign)
* Composer
* Helper klassen uit AnOrmApart
* MVC

# Analyse

## ERD

Het is essentieel voor projecten zoals deze om een ERD op te stellen. In onderstaande figuur zijn de 3 entiteiten voor dit project aanwezig.



Figuur 1: ERD

Bij het omzetten van dit model naar een database diagram krijgen we 1 tabel per entiteit en 1 extra tussentabel voor de veel op veel relatie tussen entiteiten Vraag en Functie.

## Use cases

Deze twee use cases zijn de voornaamste bewerkingen die een gebruiker zal uitvoeren op deze website.

### Maken van een pdf na het selecteren van een functie

|  |  |
| --- | --- |
| Naam | Maken van een pdf na het selecteren van een functie |
| Samenvatting | De gebruiker zal een functie selecteren en vervolgens de pdf laten genereren en dan downloaden |
| Actoren | De gebruiker, het systeem |
| Precondities | De gebruiker bevindt zich in het Instant Interview gedeelte op de website. De keuzelijst bevat de mogelijke functies. |
| Verloop | 1. De gebruiker selecteert een functie uit de keuzelijst. 2. Het systeem genereert het deel van de webpagina met competenties en vragen en selecteert de vragen die bij deze functie horen. De competenties zonder geselecteerde vragen zijn verborgen 3. De gebruiker klikt op de knop “Rapport”. 4. Een bovenliggend scherm verschijnt en bevat een overzicht van geselecteerde vragen. 5. De gebruiker klikt op “Download PDF”. 6. Het systeem genereert een PDF. 7. Het systeem stuurt het bestand door naar de browser, die het automatisch downloadt. |
| Postcondities | De gebruiker heeft het bestand op de downloadlocatie staan. |
| Alternatieve wegen | 5a. De gebruiker klikt op “Sluiten”. Het bovenliggende scherm sluit en de gebruiker kan verder vragen selecteren. |

Figuur 2: Use case Maken van een pdf na het selecteren van een functie

### Een nieuwe vraag toevoegen

|  |  |
| --- | --- |
| Naam | Een nieuwe vraag toevoegen |
| Samenvatting | De gebruiker kan een nieuwe vraag toevoegen bij een competentie |
| Actoren | De gebruiker, het systeem |
| Precondities | De gebruiker bevindt zich in het Instant Interview gedeelte op de website. |
| Verloop | 1. De gebruiker klikt op de knop “Voeg vraag toe”. 2. Een bovenliggend scherm verschijnt en bevat een keuzelijst met alle competenties. 3. De gebruiker selecteert een competentie. 4. De gebruiker typt een vraag in het tekstvak. 5. De gebruiker klikt op de knop “Voeg toe”. 6. Het systeem voegt de vraag toe in de database en geeft op de pagina weer dat de bewerking gelukt is. 7. 3 seconden laten wordt de pagina vernieuwd zodat de nieuwe vraag kan worden aangeduid. |
| Postcondities | De gebruiker bevindt zich in het Instant Interview gedeelte en de nieuwe vraag staat in de database. |
| Alternatieve wegen | 3a. De gebruiker selecteert geen competentie.  3b. Na het klikken op “Voeg toe” zal het systeem een foutmelding laten zien aan de gebruiker.  4a. De gebruiker typt geen vraag in het tekstvak.  4b. Na het klikken op “Voeg toe” zal het systeem een foutmelding laten zien aan de gebruiker.  5a. De gebruiker klikt op “Sluiten”. Terug naar de beginsituatie. |

Figuur 3: Use case Een nieuwe vraag toevoegen

# Indeling van het project

## Mappenstructuur

Het project heeft een aantal mappen om het overzicht en de structuur te bewaren.

|  |
| --- |
| * Root   + app     - controller     - models     - views   + BLL   + DAL     - Helpers   + temp   + vendor     - composer     - setasign   + web     - css     - fonts     - img     - js |

Figuur 4: Mappenstructuur

Deze worden nu kort toegelicht.

## Map app

De map app bevat de MVC structuur. Daarom zijn er ook 3 submappen controller, models en views. Deze hebben elk hun eigen namespaces respectievelijk Controller, Models en Views. De MVC structuur zelf wordt verder toegelicht in een later hoofdstuk.

## Map BLL

Deze map bevat een aantal business logic klassen en ook een aantal php pagina’s waarnaar verwezen kan worden om een taak te laten uitvoeren door de server. De klassen zitten in namespace BLL.

## Map DAL

De DAL map bevat voornamelijk klassen die nodig zijn om met de database te kunnen verbinden en er bewerkingen mee te doen. In de submap Helpers zitten helper klassen uit AnOrmApart. De klassen in de DAL map zitten in namespace DAL.

## Map temp

Wanneer een pdf bestand gegenereerd wordt, kan deze niet rechtstreeks naar de browser als download gestuurd worden omdat dit deel functionaliteit wordt aangeroepen met een POST call via ajax. Daarom worden de gegenereerde pdf’s op de server in een map temp opgeslagen. Elke keer een pdf gemaakt wordt, zullen oudere pdf’s verwijderd worden.

## Map vendor

De vendor map bevat extra packages die we in dit project nodig hebben. Er wordt gebruik gemaakt van FPDF van setasign. Dit wordt door de dependency manager composer (zie [5.8](#_Composer)) verzorgd en dus is ook een map composer bij de vendors.

## Map web

De map web bevat html elementen en de index.php waarmee de opbouw van de webpagina start. Er is de volgende onderverdeling voor submappen:

* css: bevat de bestanden met CSS voor de website. Hier zit ook het bootstrap css bestand.
* fonts: bevat de lettertypes van bootstrap.
* img: het bestand loading.gif zit in deze map om een loading animatie te kunnen gebruiken.
* js: In deze map zitten de javascript bestanden en dus ook een deel van bootstrap.

## Composer & namespaces

In dit project wordt er gebruikt gemaakt van composer, een dependency manager. Na de installatie van de standaard composer zijn de bestanden aangepast.

### Aanpassing aan autoload\_real.php

De getLoader() functie van autoload\_real.php werd aangepast, het volgende stuk code is toegevoegd zodat zelf gemakkelijk de namespaces kunnen beheerd worden in apart bestand.

|  |
| --- |
| $map = **require *\_\_DIR\_\_*** . **'/autoload\_II.php'**; **foreach** ($map **as** $namespace => $path) {  $loader->setPsr4($namespace, $path); } |

Figuur 5: aanpassing aan autoload\_real.php

### Toevoeging van autoload\_II.php

Dit bestand bevat de definitie van een aantal namespaces. Door in de bestanden telkens de autoloader te gebruiken, worden de definities van de namespaces omgezet en wordt telkens het juiste klasse bestand gebruikt.

|  |
| --- |
| **<?php**  $vendorDir = *dirname*(*dirname*(***\_\_FILE\_\_***)); $baseDir = *dirname*($vendorDir); $appDir = *realpath*($baseDir . **'/app/'**); $dalDir = *realpath*($baseDir . **'/DAL/'**); $bllDir = *realpath*($baseDir . **'/BLL/'**);  **return array**(  **'Controller\\'** => **array**($appDir . **'/controller'**),  **'Models\\'** => **array**($appDir . **'/models'**),  **'Views\\'** => **array**($appDir . **'/views'**),  **'DAL\\'** => **array**($dalDir),  **'BLL\\'** => **array**($bllDir), ); |

Figuur 6: Toevoeging van autoload\_II.php

# Opbouw

## Index.php en Javascript

### Head

De head van de index.php pagina bevat de standaard elementen die daar te verwachten zijn:

* Meta tags
* Bootstrap viewport code
* Stylesheet links naar de css bestanden
* Links naar de source bestanden van jQuery en Ajax.

### Pagina-indeling

In de body van index.php staat bovenaan een carousel element. Daaronder staat een menu waarmee je snel naar de verschillende paginaonderdelen kan gaan. Dit wordt door bootstrap gedaan en vraagt dus geen verdere uitleg. Verder staan de paginaonderdelen er zelf op, zoals een info gedeelte met wat info en eentje met contactgegevens.

### Div interviewForm

Dit div element bevat heel weinig maar heel belangrijke code, namelijk:

|  |
| --- |
| **<?php require '../app/views/JobFunctionsView.php'**; **?>** |

Figuur 7: stukje code uit index.php

Dit zal de JobFunctionsView code op deze plaats in de pagina plaatsen.

### Modals

Op deze index.php staan ook nog twee div elementen met class modal. Deze “report-modal” en “add-question-modal”. Dit zijn de bovenliggende vensters die zullen verschijnen wanneer de gebruiker een vraag wil toevoegen of een pdf wil maken van de geselecteerde vragen.

## MVC

### Models

Er zijn in dit MVC model drie entiteiten waar een Model van geïmplementeerd wordt. Deze implementatie is vrij vanzelfsprekend en eenvoudig.

#### Question

|  |
| --- |
| **namespace** models;  **class** Question {  **private $id**;   **public function** setId($id) {  $this->**id** = $id;  }   **public function** getId() {  **return** $this->**id**;  }   **private $fullQuestion**;   **public function** setFullQuestion($fullQuestion) {  $this->**fullQuestion** = $fullQuestion;  }   **public function** getFullQuestion() {  **return** $this->**fullQuestion**;  }   **private $competenceId**;   **public function** setCompetenceId($competenceId) {  $this->**competenceId** = $competenceId;  }   **public function** getCompetenceId() {  **return** $this->**competenceId**;  }   **public function** \_\_construct($id, $fullQuestion, $competenceId)  {  $this->**id** = $id;  $this->**fullQuestion** = $fullQuestion;  $this->**competenceId** = $competenceId;  } } |

#### Competence

|  |
| --- |
| **namespace** Models;  **class** Competence {  **private $id**;  **public function** setId($id) {  $this->**id** = $id;  }   **public function** getId() {  **return** $this->**id**;  }   **private $name**;  **public function** setName($name) {  $this->**name** = $name;  }   **public function** getName() {  **return** $this->**name**;  }   **public function** \_\_construct($id, $name)  {  $this->**id** = $id;  $this->**name** = $name;  } } |

#### JobFunction

|  |
| --- |
| **namespace** Models;  **class** JobFunction {  **private $id**;  **public function** setId($id) {  $this->**id** = $id;  }   **public function** getId() {  **return** $this->**id**;  }   **private $name**;  **public function** setName($name) {  $this->**name** = $name;  }   **public function** getName() {  **return** $this->**name**;  }   **public function** \_\_construct($id, $name)  {  $this->**id** = $id;  $this->**name** = $name;  } } |

### Views

### Controller

## BLL

## DAL

# Uitbreiding

## Uitbreiding blijft mogelijk

## Use case diagram