

Voorwoord Mohamed Amajoutt

donderdag 4 juni 2015

1:23

Bij het schrijven van dit eindwerk wil ik graag een woordje van dank neerschrijven voor de mensen die mij geholpen hebben bij de uitwerking van dit eindwerk.

Allereerst wil ik docent Jef Inghelbrecht bedanken voor de raad en steun die hij mij heeft geboden tijdens de uitwerking van het project en het schrijven van dit eindwerk. In de loop van mijn opleidingsjaren op CVO Antwerpen was hij steeds de docent waar ik bij terecht kon voor allerhande vragen en opmerkingen, wat mij veel steun heeft geboden om te blijven verder doen.

Daarnaast wil ik ook mijn dank richten aan trajectbegeleider Ivan Robeyns en Departementshoofd hoger beroepsonderwijs Christiane Andries voor altijd klaar te staan als er problemen, vragen of aanpassingen waren in verband met mijn studies. Ook wil ik docenten Tom Verbesselt, Dimitri Sturm, Nico Picavet, Felix Mertens en alle andere docenten bedanken waar ik veel van heb kunnen leren en praktiseren.

Graag wil ik mijn steun betuigen aan de vrienden, mede-studenten en familie van de overledene Wouter Aerts. De eerste kennismaking met Wouter was in de les Programmeren 4 - JAVA, waar ik soms met hem een gesprek kon voeren over taken en projecten. Ik kan hem in die paar lessen beschrijven als een joviale, sympathieke en leergierige jongeman die gedreven en gepassioneerd programmeerde.

Verder wil ik ook een speciaal woordje van dank richten aan mijn familie en mijn vriendin Brenda, die mij tijdens deze opleiding in goede en slechte momenten hebben gesteund.

Mohamed Amajoutt, 22 jaar

Graduaat in Informatica , afstudeerrichting Programmeren

CVO Antwerpen, Hoboken

juni 2015

Over CVO Antwerpen

donderdag 4 juni 2015

1:26

CVO Antwerpen

CVO Antwerpen is een centrum voor volwassenonderwijs met diverse campussen in en rond de stad Antwerpen in het Vlaamse gewest van België. CVO Antwerpen is een van de grootste centra voor volwassenonderwijs gecertificeerd en gefinancierd door de Vlaamse Overheid.

Men organiseert meer dan 950 modulaire dag- en avondcursussen per jaar. Deze cursussen zijn opgedeeld in:

- cursussen voor immigranten (Nederlands als een tweede taal, ICT, andere talen);
- formele beroepsonderwijs en opleiding op het niveau van secundair onderwijs (tweede kans om te leren, Die kan leiden tot een diploma tot het behalen van een diploma van het secundair onderwijs op het gebied van algemene administratie, lassen en multimedia graphics).
- hoger beroepsonderwijs (boekhouding, bouwkundig tekenaar, ICT);
- waaier van taalcursussen (Chinees, Engels, Frans, Duits, Italiaans, Portugees, Russisch en Spaans);
- Ondernemende trainingen, maritieme opleidingen, lascursussen, cursussen in applicatie software en koken.

CVO Antwerpen biedt een toenemend percentage van de cursussen als blended learning. Het centrum is, was en blijft een toonaangevende speler in de introductie van verschillende types van leren op afstand met behulp van e-learning platform, het ontwikkelen van interactieve content (leerobjecten), multimedia en video-opname, social software, geautomatiseerde evaluatie, mobiel leren ... Het centrum telt een team van ICT-personeel en natuurlijk ontwerpers om de uitvoering van haar onderwijs op afstand te ondersteunen.

De belangrijkste campussen (o.a. campus Hoboken) bieden een open leercentrum, het verstrekken van faciliteiten voor intake en adviesdiensten, zelfstudie boeken, advies, individuele begeleiding en studiegroepen.

Huidige werking

CVO Antwerpen heeft een secretariaat en trajectbegeleiders ter beschikking om de studenten verder te helpen met allerhande vragen. Als u vragen hebt over uw lessenrooster, attesten, facturatie, attesten van ziekte, etc dan moet u in de meeste gevallen tot het secretariaat wenden. Wanneer u vragen hebt over uw traject op de school, persoonlijke- en schoolproblemen, deliberatedatum, inschrijven tweede zit, punten, etc dan moet u zich tot de trajectbegeleider van de opleiding wenden. Om met de trajectbegeleider te spreken moet men altijd een afspraak die meestal per mail gebeurd.

Er wordt gebruik gemaakt van het online leerplatform Moodle om informatie met cursisten te delen. Tevens beschikt de school over een eigen website en is ook actief op onder andere: facebook en youtube.

Huidige systeem

Vandaag gebruikt de school een systeem dat bestaat uit één centrale databank waarin alle gegevens van cursisten, docenten en cursussen worden beheert.

Namelijk in de SQL Server Administratix databank.

Auteurs:

Sonja Van Rompaey
Mohamed Amajoult

Opdracht

donderdag 4 juni 2015

1:28

Project: MobileCVO

Een cursistenportaal op meerdere platforms voor CVO Antwerpen

Past in beleidsplan CVO Antwerpen:

SD 6. Wij bouwen en integreren slimme digitale oplossingen om een boeiende en efficiënte leer- en werkomgeving te creëren;
OD 6.1. Wij openen onze administratie voor cursisten en medewerkers via een e-loket.

Probleemstelling en huidige situatie:

- cursisten volgen wijzigingen van lesmomenten niet;
- cursisten kennen het cursuslokaal niet;
- cursisten vergeten startdatum van nieuwe cursus;
- cursisten weten niet wanneer er wordt gedelibereerd;
- cursisten weten niet dat er een inschrijving voor 2e zittijd nodig is;
- cursisten kennen examendatum niet;
- cursisten kennen datum 2e zit niet;
- cursisten vergeten datum (deadlines) van opdrachten;
- cursisten verschijnen niet op afspraken met trajectbegeleider;
- cursisten weten niet dat examenresultaten zijn gepubliceerd;
- cursisten willen punten weten, dit kunnen ze nu enkel op hun attest consulteren of op verzoek aan een docent na de deliberatie.

Oorzaken:

Deze informatie (les data en lesmomenten, examenresultaten, e.d.) wordt nu verspreid via

- mail;
- website;
- mondeling;
- moodle;
- inschrijvingsfiche;
- ECTS-fiche;
- moodle-forum van verschillende cursussen;
- secretariaat.

Conclusie:

- teveel verschillende kanalen waarlangs informatie wordt gecommuniceerd;
- het gebeurt via media die niet altijd frequent worden geconsulteerd door de cursist.

Deelprojecten voor het cursistenportaal

- cursistaccounts;
- mobiele kalender als basis:
 - 90% van de gegevens bevat kalendergegevens, dus een centrale agenda is nodig;
 - centrale agenda moet maximaal automatisch worden ingevuld wat betreft cursusplanning en evaluatie;
 - centrale agenda moet gepersonaliseerd kunnen worden;
 - centrale agenda moet online consulteerbaar zijn via pc en via smartphone, bij

voordeur via automatische synchronisatie met persoonlijke kalender.

- studieresultaten: mobiele resultaten, app waarmee cursisten hun eigen resultaten kunnen consulteren;
- EVC-EVK vrijstellingsdossiers (Erkenning van Verworven Competenties / Kwalificaties)
- online (her)inschrijven voor cursus(sen) o.a. met EID en betaling, tariefsuggestie;
- cursist kan eigen traject consulteren en met trajectbegeleiding afspraak cursist vastleggen;
- persoonlijke lesroosters, waarbij ook examen, 2e zit, enz. worden vermeld;
- inschrijven voor examen 2e zit.

Doel:

Maak een werkend prototype van een cursistenportaal voor CVO Antwerpen: een webapplicatie, met als voornaamste target mobiele apparaten, naast de klassieke desktop-browser.

Implementatie met moderne technologieën (HTML5, CSS3, JavaScript, Dart, enz. / geen silverlight, klassieke ASP.NET, enz.).

De docenten Jef Inghelbrechts en Ivo Balbaert treden op als groepsmanagers en mogen in die hoedanigheid evalueren.

Milestones:

- M1 Project Requirements, 26/02/2015.
- M2 Functionele Specificaties (1 Project Requirements document per groep), 5/03/2015.
- M3 Use Cases, 12/03/2015.
- M4 Klassendiagram, 19/03/2015.
- M5 ER Diagram, 26/03/2015.
- M6 Programming 1^e fase, 2/04/2015.
- M7 Programming 2^e fase, 23/04/2015.
- M8 Programming 3^e fase, 7/05/2015.
- M9 Technische Documentatie, 6/06/2015.
- M10 Handleiding, 6/06/2015.
- M11 Videopresentatie, 6/06/2015.
- M12 Verdediging Project, 16/06/2015.

Auteurs:

Sonja Van Rompaey

Mohamed Amajoutt

Groep samenstelling

donderdag 4 juni 2015
1:33

Leden

Onze groep bestaat uit 6 cursisten, zijnde:

- Mohamed Amajoult;
- Sonja van Rompaey;
- Nikos Vanden Broek;
- Jo Ronsse;
- Benjamin Peeters;
- Andreas de Bresser.

Contactpersonen

donderdag 4 juni 2015

1:34

Medewerkers CVO Antwerpen:

Een aantal medewerkers van het CVO die te maken hebben met cursistenbegeleiding en cursistenzaken kunnen worden uitgenodigd voor gebruikersvergaderingen, bv.:

Marijke Quanjard: adjunct-directeur cursistenzaken	marijke.quanjard@cvoantwerpen.be
Ivan Robeyns: trajectbegeleider HBO Informatica	ivan.robeyns@cvoantwerpen.be
Christiane Andries: adjunct-directeur departement hoger beroepsonderwijs	candries@cvoantwerpen.be
Yves Bouillon: hoofd IT CVO Antwerpen	yves.bouillon@cvoantwerpen.be

Docenten:

De docenten treden op als groepsmanagers en mogen in die hoedanigheid evalueren.

Jef Inghelbrechts: docent & groepsleider	jef.inghelbrechts@cvoantwerpen.be
Ivo Balbaert: docent & groepsleider	ivo.balbaert@cvoantwerpen.be

Meetings met coördinatoren:

Christiane Andries: Adjunct-directeur Departement Hoger Beroepsonderwijs.

Datum: 26/02/2015

Opmerking:

Besprekning van de functionele specificaties voor de applicatie.

Deze zijn onderverdeeld in drie hoofdgroepen: een kalender, de resultaten en mogelijkheid om online in te kunnen schrijven (haalbaarheid?).

Yves Bouillon: Verantwoordelijke IT Infrastructuur.

Datum: 5/05/2015

Opmerking:

Rondleiding in de Administratix database, richtlijnen en extra uitleg over de functionaliteit van de tabellen.

Ivan Robeyns: Trajectbegeleider HBO Informatica.

Datum: 12/03/2015

Opmerking:

Gesprek met Ivan over zijn verwachtingen van de applicatie, deze stemmen overeen met die van Christiane Andries.

Later is wegens tijdsnood beslist om enkel een interface voor studenten te maken en niet voor docenten of trajectbegeleiders.

Jef Inghelbrecht: Docent Projectwerk Programmeren

Datum: 21/03/2015

Opmerking:

Feedback over Use Cases, Functionele Specificaties en Klassendiagrammen.

De use cases zijn goed gemaakt, een klassendiagram en erd moet gemaakt worden aan de hand van de bestaande database.

Datum: 3/04/2015

Opmerking:

Positieve feedback op getekende user interfaces.

Nieuwe tabellen dienen aangemaakt te worden in de Admistratix database.

Datum: 19/04/2015

Opmerking:

Possitief over mijn aanzet tot de bll en dal implementatie in het project.

Ik heb alle nodige klassen in de bll geplaatst en voorbeeld code in de dal geschreven, zodat de groep deze verder kan uitwerken.

Datum: 3/05/2015

Opmerking:

Overlopen van mijn geschreven stored procedures, die functioneel zijn maar waar nog enkele joins en links met tabellen ontbraken.

Datum: 9/05/2015

Opmerking:

Mijn logo ontwerp wordt goedgekeurd, mockup is nog te complex en moet vereenvoudigd worden.

Datum: 28/05/2015

Opmerking:

Testen van gemaakte pages en bespreken van de mockup.

Alles wordt omgezet naar tegels omdat dit beter bruikbaar is mobile.

Auteurs:

Sonja Van Rompaey

Mohamed Amajoult

Logingegevens MobileCVO

maandag 1 juni 2015

17:31

Project Mobile-CVO

<http://studyplanit.com/mobilecvo/>

Login

Toegang tot de applicatie kan met volgende cursistnummers:

26544 : Sonja Van Rompaey;

46371 : Nikos VandenBroek;

46408 : Jo Ronsse;

26946 : Benjamin Peeters;

43703 : Mohamed Amajoutt.

Communicatie Tools

woensdag 3 juni 2015
15:34

Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt

Facebook

Via de sociale media applicatie van Facebook is er groepschat aangemaakt.
Hierdoor kan er altijd gecommuniceerd worden met elkaar op een moment naar keuze.
De bevindingen van deze methode van communicatie zijn positief bevonden aangezien iedereen consequent gebruik maakt van Facebook.
Bijna dagelijks worden er vragen gesteld, feedback gegeven en overlegd.

OneNote

Meteen nadat de groep is gevormd hebben we besloten om OneNote te gebruiken.
Via deze methode kunnen we ons project visualiseren door secties op te bouwen en ideeën/code/ontwerpen uit te wisselen.
In OneNote heb ik onder andere een sectie agenda aangemaakt waarin verslagen staan van bijeenkomsten, zodanig dat deze altijd kunnen nagelezen worden en iedereen weet wat er juist werd afgesproken.

Versioning Tool

donderdag 4 juni 2015

11:35

Auteurs:

Sonja Van Rompaey

Mohamed Amajoult

Git

Om gemakkelijk en veilig code uit te wisselen en samen één project op te bouwen hebben we besloten om Git te gebruiken.

Er werd een gezamelijke repository aangemaakt waar iedereen toegang tot heeft.

Op deze manier kan iedereen code toevoegen aan het project en kunnen we elkaar aanvullen en helpen bij het programmeren.

Een voordeel van Git is ook dat er terug in de tijd kan gegaan worden om bijvoorbeeld een vorige staat van het project op te vragen.

Functionele specificaties

maandag 1 juni 2015

17:28

Lessenrooster:

- Je moet als student informatie uit de lessenrooster kunnen opvragen;
- Je moet als student de lessenrooster kunnen bekijken.

Puntenoverzicht:

- Je moet als student een overzicht krijgen van de ingeschreven cursussen;
- Je moet als student een overzicht krijgen van de behaalde cursussen;
- Je moet als student een overzicht krijgen van de lopende cursussen;
- Je moet als student een overzicht krijgen van de nog te starten cursussen;
- Je moet als student bij de lopende cursussen deliberatie en examen datums kunnen opvragen;
- Je moet als student de uitkomst van de deliberatie kunnen opvragen;
- Je moet als student de puntenverdeling en resultaten van een cursus kunnen opvragen;
- Je moet als student de datum van 2de zit kunnen opvragen;
- Je moet als student kunnen inschrijven voor de 2de zit.

Cursus informatie:

- Je moet als student informatie over een docent kunnen opvragen;
- Je moet als student informatie over een cursus opvragen (lokaal, datum, examendatum, docent,...);
- Je moet als student deadlines en datums van opvragen.

Overzicht traject:

- Je moet als student een overzicht kunnen bekijken van alle cursussen die geslaagd moeten worden voor het traject;
- Je moet als student kunnen zien welke cursussen geslaagd moeten zijn om de volgende cursus te mogen volgen;
- Je moet als student kunnen zien of een cursus geslaagd, gefaald, lopend of nog te volgen is;
- Je moet als student kunnen bekijken of hij/zij al is ingeschreven voor een vak of dat de student zich nog moet inschrijven.

Contact:

- Je moet als student email adressen/ telefoon nummers kunnen opzoeken van; docenten, medecursisten,... ;
- Je moet als student openingsuren kunnen opvragen van het secretariaat.

Agenda:

- Je moet als student een overzicht krijgen d.m.v. een kalender met lesmomenten, vakantiedagen, ... per dag.

Inschrijvingen:

- Je moet als student kunnen inschrijven voor evenementen;
- Je moet als student kunnen inschrijven voor de 2de zit.

Inlogscherm:

- Je moet als student kunnen inloggen enkel met een cursist-nummer.

Auteurs:

Sonja Van Rompaey
Mohamed Amajoult

Project Requirements

donderdag 4 juni 2015

1:36

Lessenrooster:

- Naam van de cursus
- Start en eind uur van de cursus
- Lokaal van de cursus
- Belangrijke informatie zoals lokaalwijziging en ziekte docent
- Aanduiding begin (misschien), examen
- Deliberaties + mogelijk geslaagd, deliberatie, ... (zie punten overzicht)
- Link naar cursus informatie (ref.)
- Vakantiedagen (optie)
- Zowel kalender vorm als lijn vorm (lesrooster secretariaat)
- Afdruk mogelijkheid

Puntenoverzicht:

- Overzicht van cursussen die van belang zijn voor de student
- Gescheiden afgeronde en cursussen die nog lopen of moeten beginnen
- Lopend: datum van deliberatie
- Afgerond: geslaagd of niet + punten percentage
- Overzicht punten verdeling & resultaten
- Niet geslaagd: datum 2^{de} zit + manier om in te schrijven
- Afdrukken
- Aanvraag om examen in te kijken

Cursus informatie:

- Informatie docent: naam, contact, ..
- Lokaal & datum (nog te beginnen: eerste les / lopende cursus: volgende les)
- Examen datum (aantal dagen tot)
- Netwerk login & paswoord
- Deadline openstaande taak(en)
- Enquête bij afloop cursus

Overzicht traject:

- Overzicht van alle cursussen die geslaagd moeten worden voor het traject
- Aanduiding van cursussen die geslaagd moeten zijn om de volgende cursus te mogen volgen
- Aanduiding of een cursus geslaagd, gefaald, lopend of nog te volgen is
- Weergave of de student al is ingeschreven voor een vak of dat de student zich nog kan inschrijven
- Mogelijke vrijstellingen
- Certificaat status

Contact:

- Een mail systeem om berichten te sturen naar docenten of medecursisten
- Basis zoals een standaard e-mail programma
- Mogelijkheid om berichten per persoon te bekijken zoals smartphone sms
- (details) Ontvangers weergeven in categorieën; docent, medecursisten per vak, administratie, ...
- Favorieten / veel voorkomende contacten / recent

Overzicht meldingen:

- Belangrijke informatie zoals : Leerkracht afwezig, lokaalwijziging , punten van een vak zijn bekend
- Melding bij start van een module (vak+lokaal+uur)
- Bericht volgende les examen
- Keuzemogelijkheid welk soort berichten wel of niet weergegeven worden

Agenda:

- Kalender met lesmomenten, afspraken, vakantiedagen, ... per dag
- Gebruik van kleurenschema voor onderscheid tussen lesdag, examen, vrije dagen

Inschrijvingen:

- Inschrijven voor evenementen
- Inschrijven voor 2^{de} zit
- Overzicht aantal vrije plaatsen per vak
- Inschrijving vakken (optioneel)
- Uitschrijven voor een vak (?)

Inlogscherm:

- Inloggen met cursist login (Moodle)
- Wachtwoord vergeten

Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt

Flowchart

maandag 1 juni 2015
17:36

Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt

Beschrijving

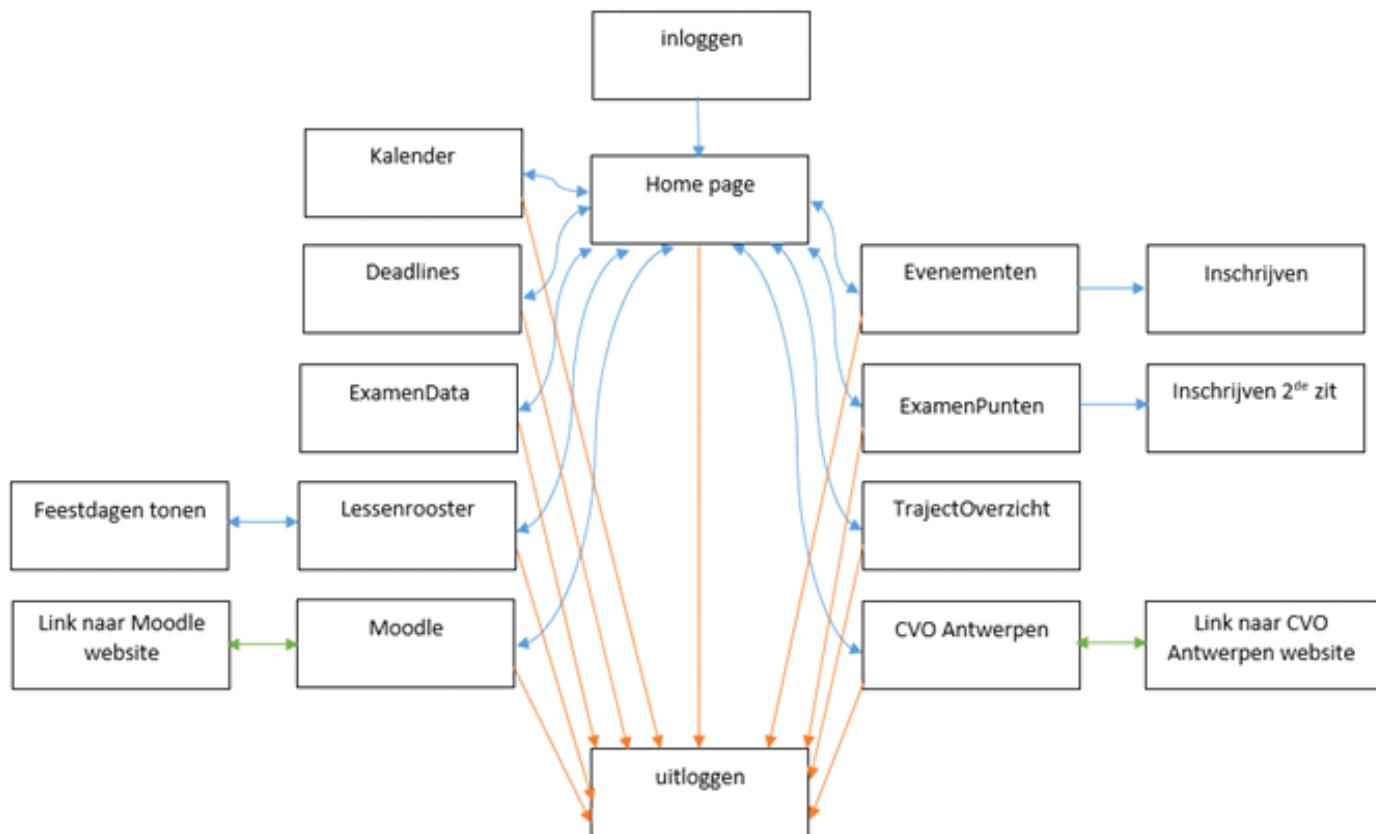
Wanneer de cursist zich heeft aangemeld met zijn cursist nummer komt deze terecht op de onthaalpagina van onze applicatie.
Nadien kan de cursist kiezen uit volgende tegels:

- Kalender
- Deadlines
- Examendata
- Lessenrooster
- Evenementen: evenementen bekijken en inschrijven
- Puntenoverzicht: resultaten bekijken en inschrijven voor 2de zit
- Trajectoverzicht
- Moodle
- CVO Antwerpen.

Binnen elke pagina is het mogelijk om zich uit te loggen. Hierna komt de gebruiker terug naar het inlogscherm.

Door tekort aan tijd zijn er de tegels:

- Moodle: die de gebruiker doorverbind naar de website van Moodle om informatie te verkrijgen over de lessen.
- CVO Antwerpen: die je naar de cvo antwerpen website brengt om informatie te verkrijgen over de school.



Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt

Use cases

maandag 1 juni 2015

17:27

Auteurs:

Sonja Van Rompaey
Mohamed Amajoult

Inloggen op de website:

Naam: Inloggen op de website.

Samenvatting: Cursist heeft toegang tot de website.

Actoren: Cursist

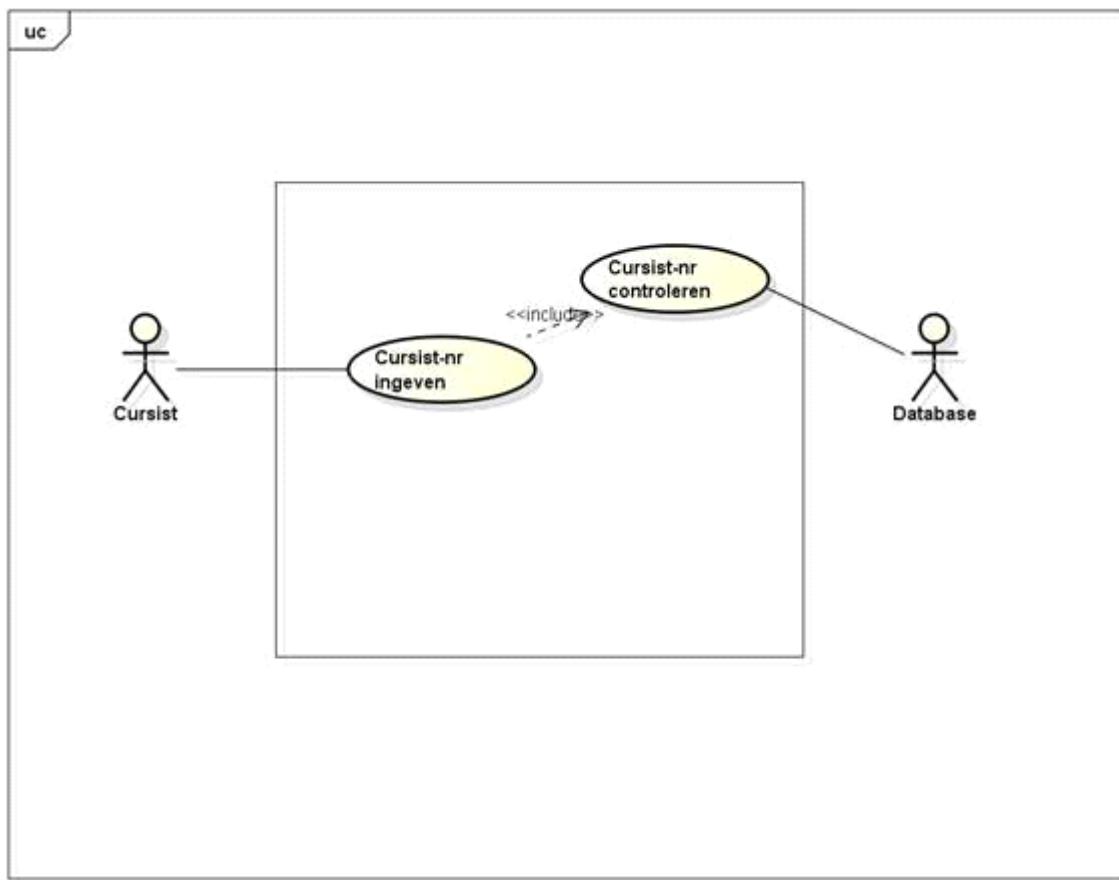
Aannamen: Cursist beschikt over een cursistnummer.

Beschrijving: 1. Cursist geeft zijn cursist-nummer in.
2. Cursist heeft toegang tot de website.

Uitzonderingen: -Cursist heeft nog geen cursist-nummer.

-Cursist heeft geen internet toegang.

Resultaat: Cursist is ingelogd op de website.



Kalender bekijken:

Naam: Kalender bekijken.

Samenvatting: Cursist krijgt een overzicht per maand met aanduiding van de dag, feestdagen, deadlines en lesmomenten.

Actoren: Cursist

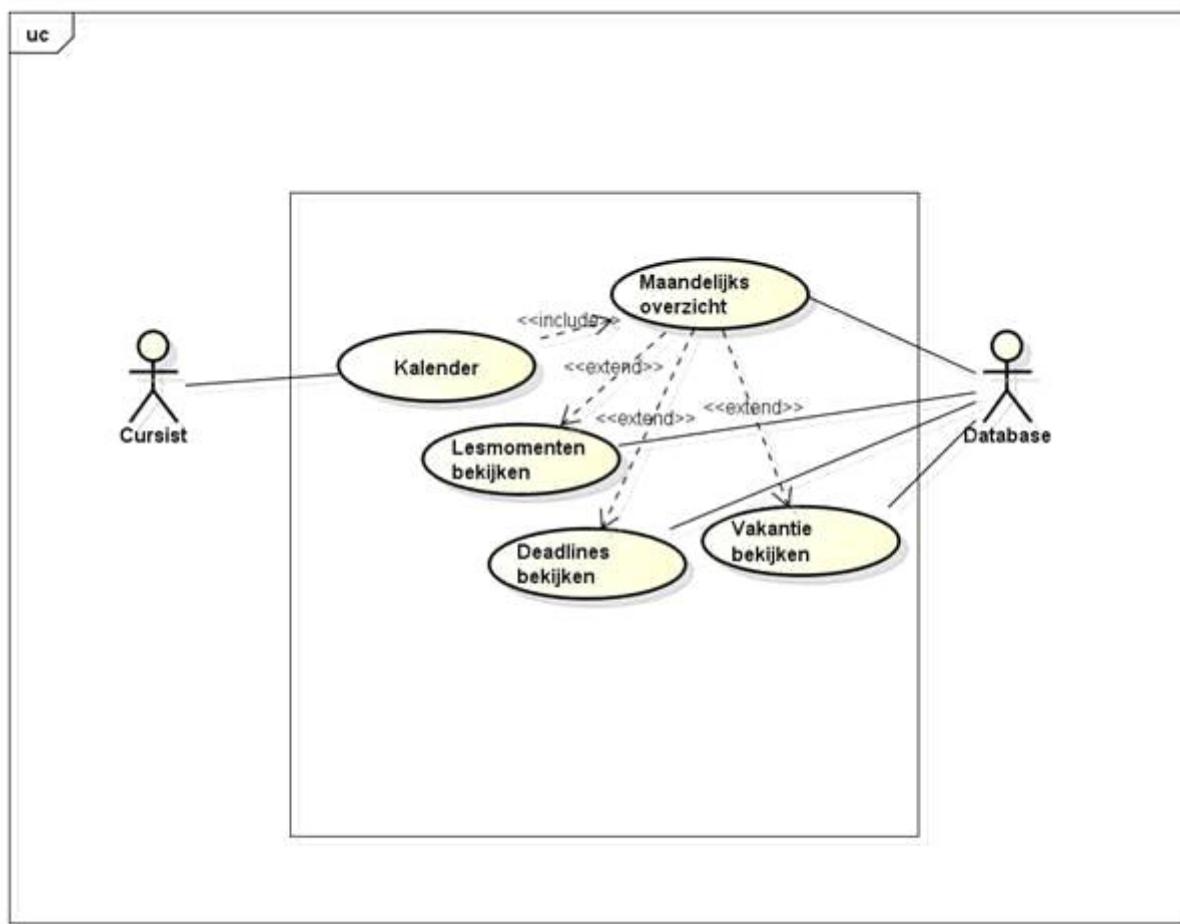
Aannamen: Cursist is ingelogd in het systeem.

Beschrijving:

1. Cursist logt in op de website.
2. Cursist gaat naar de tegel Kalender.
3. Cursist krijgt een overzicht van zijn lesmomenten, deadlines en vankantiedagen.

Uitzonderingen: Foutieve inloggegevens

Resultaat: Cursist krijgt een kalender voor zich.



powered by Astah

Lesrooster bekijken:

Naam: Lesrooster bekijken.

Samenvatting: Cursist moet een overzicht krijgen van de ingeschreven lessen.

Actoren: Cursist

Aannamen: Cursist is ingelogd in het systeem.

Beschrijving:

1. Cursist logt in op de website.

-juiste inloggegevens

2. Cursist gaat naar de tegel Lesrooster.

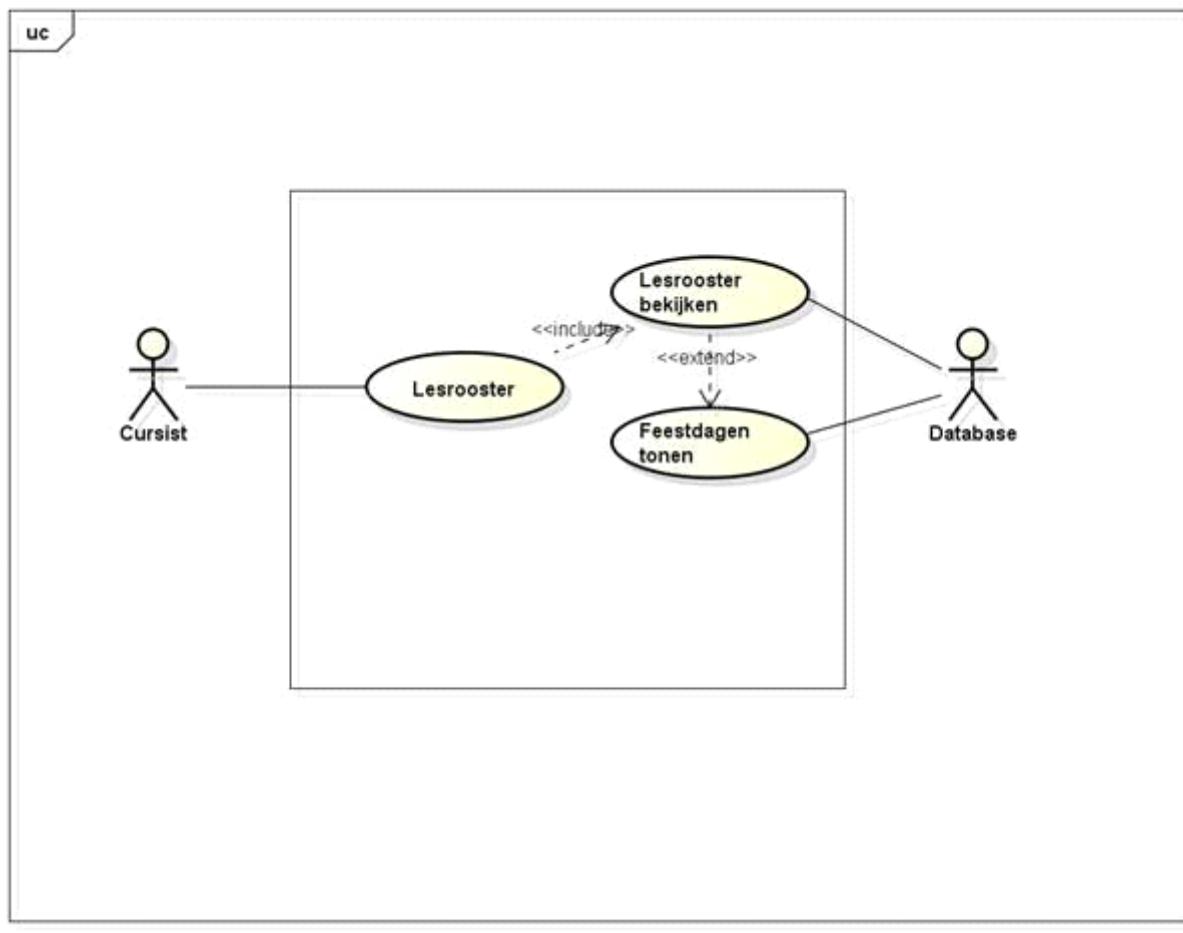
3. Cursist krijgt een overzicht van zijn lesrooster met de mogelijkheid om feestdagen te tonen.

-foutieve inloggegevens

2. Cursist krijgt een foutbericht te zien.

Uitzonderingen: -Ongeldige login.
-Cursist is niet ingeschreven voor modules.

Resultaat: Cursist kan zijn persoonlijke lesrooster bekijken.



powered by Astah

Inschrijven voor een evenement:

Naam: Inschrijven voor een evenement.

Samenvatting: Cursist kan zich inschrijven voor een evenement.

Actoren: Cursist

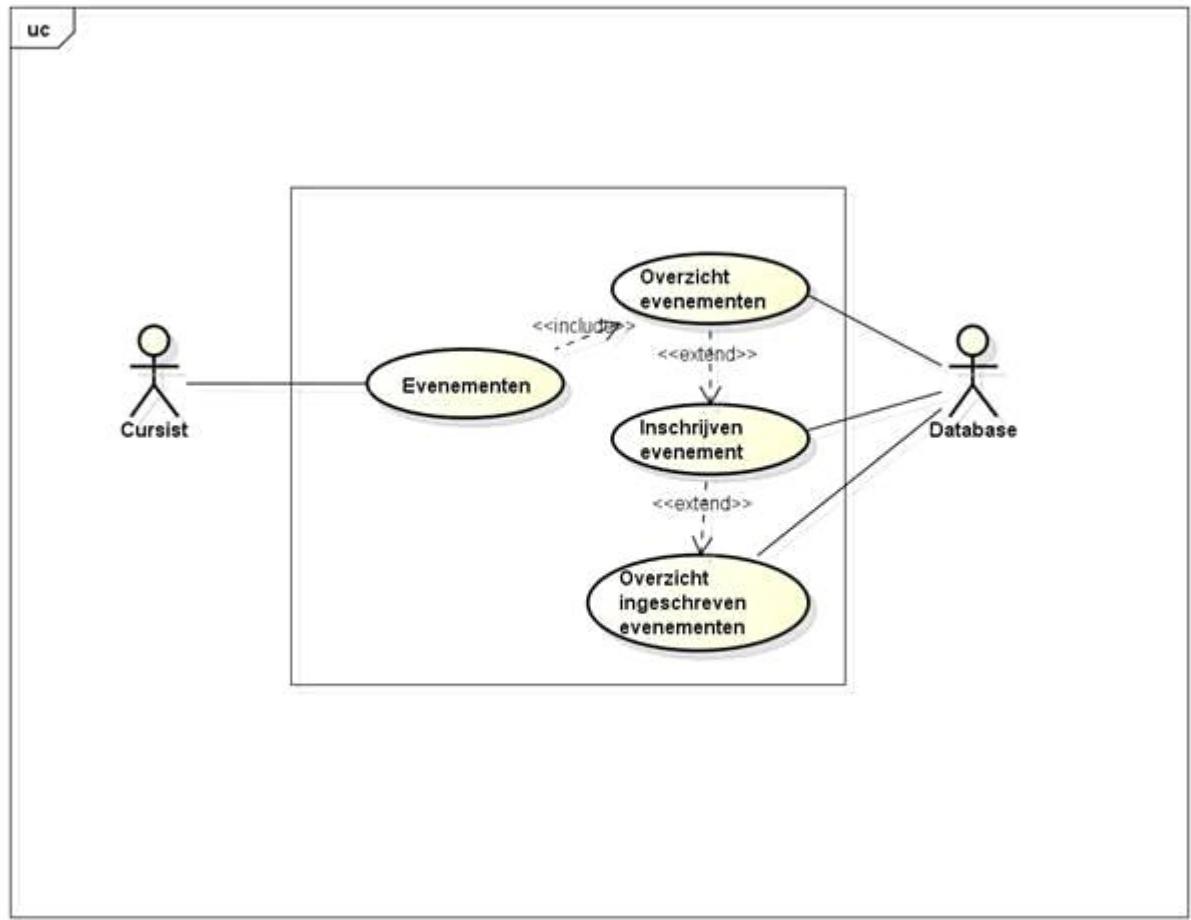
Aannamen: Cursist is ingelogd in het systeem.

- Beschrijving:
1. Cursist is ingelogd op het systeem.
 2. Cursist gaat naar de tegel Evenementen.
 3. Cursist krijgt een overzicht van alle evenementen.
 4. Cursist klikt op de knop inschrijven voor een evenement naar keuze.

Uitzonderingen
-Er zijn geen evenementen georganiseerd.
: -Foutieve cursistnummer

Resultaat: Cursist is ingeschreven voor een evenement.





powered by Astah

Feestdagen bekijken:

Naam: Alle vakantiedagen bekijken.

Samenvatting: Cursist kan zien op welke dagen er vakantie is.

Actoren: Cursist

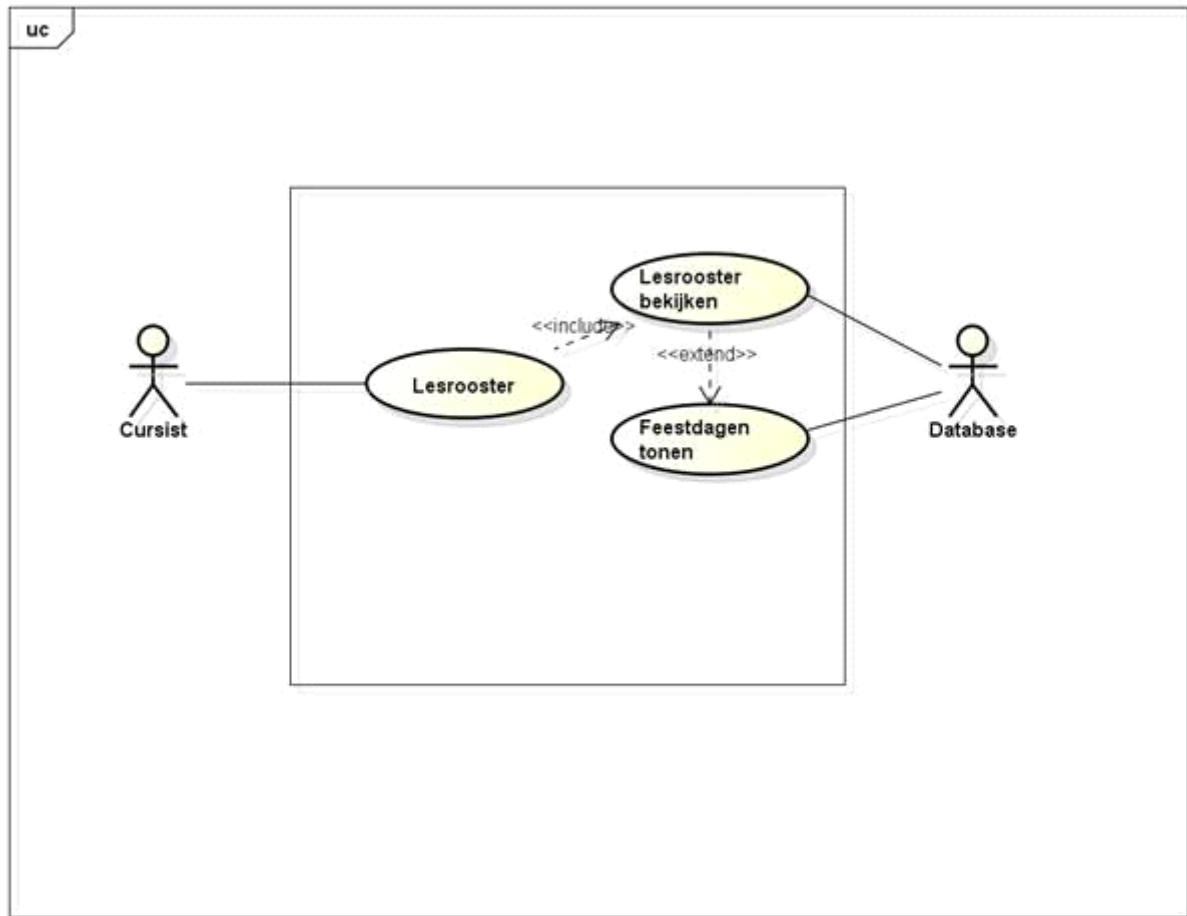
Aannamen: Cursist is ingelogd in het systeem.

Beschrijving:

1. Cursist is ingelogd op het systeem.
2. Cursist gaat naar de tegel Lesrooster.
3. Cursist kan kiezen om feestdagen te tonen of te verbergen.

Uitzonderingen: -Er zijn geen vakanties ingegeven.
-Foutieve

Resultaat: Cursist krijgt een overzicht van de lessen samen met de feestdagen.



powered by Astah

Deadlines bekijken:

Naam: Deadlines bekijken.

Samenvatting: Cursist kan kijken welke opdrachten hij moet inleveren.

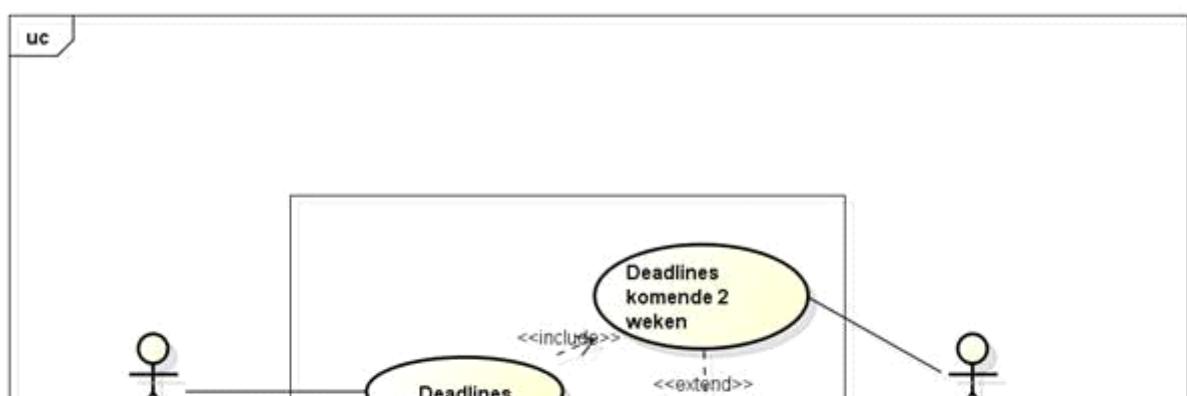
Actoren: Cursist

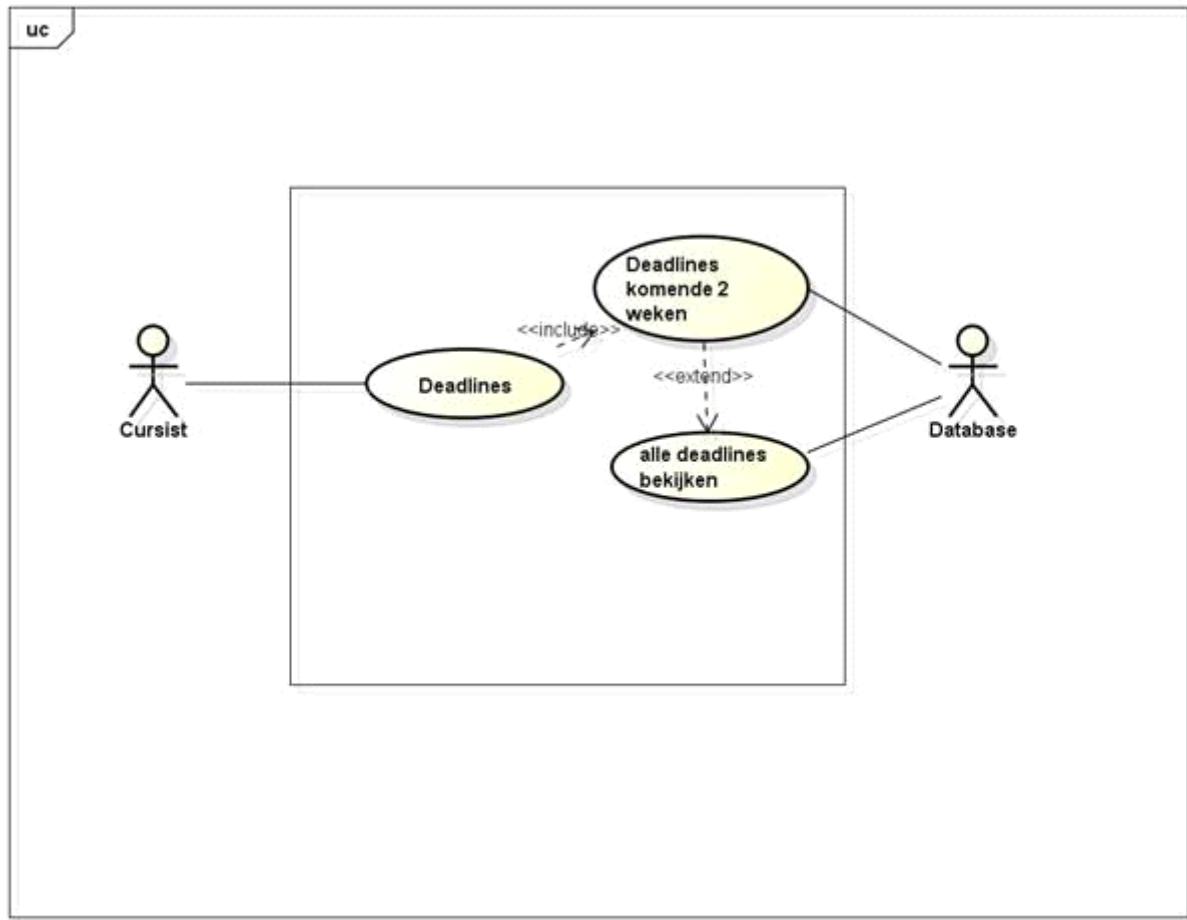
Aannamen: Cursist is ingelogd met een cursistnummer

- Beschrijving:
1. Cursist logt in op het systeem.
 2. Cursist gaat naar de tegel Deadlines.
 3. Cursist heeft een overzicht van de deadlines.

Uitzonderingen: -Er zijn geen opdrachten of taken.

Resultaat: Cursist heeft een overzicht van de deadlines.





powered by Astah

Trajectoverzicht opvragen:

Naam: Trajectoverzicht opvragen

Samenvatting: Een overzicht van alle modules in een traject en de melding of deze reeds voldaan is, nog te volgen en de nodige voorkennis.

Actoren: Cursist

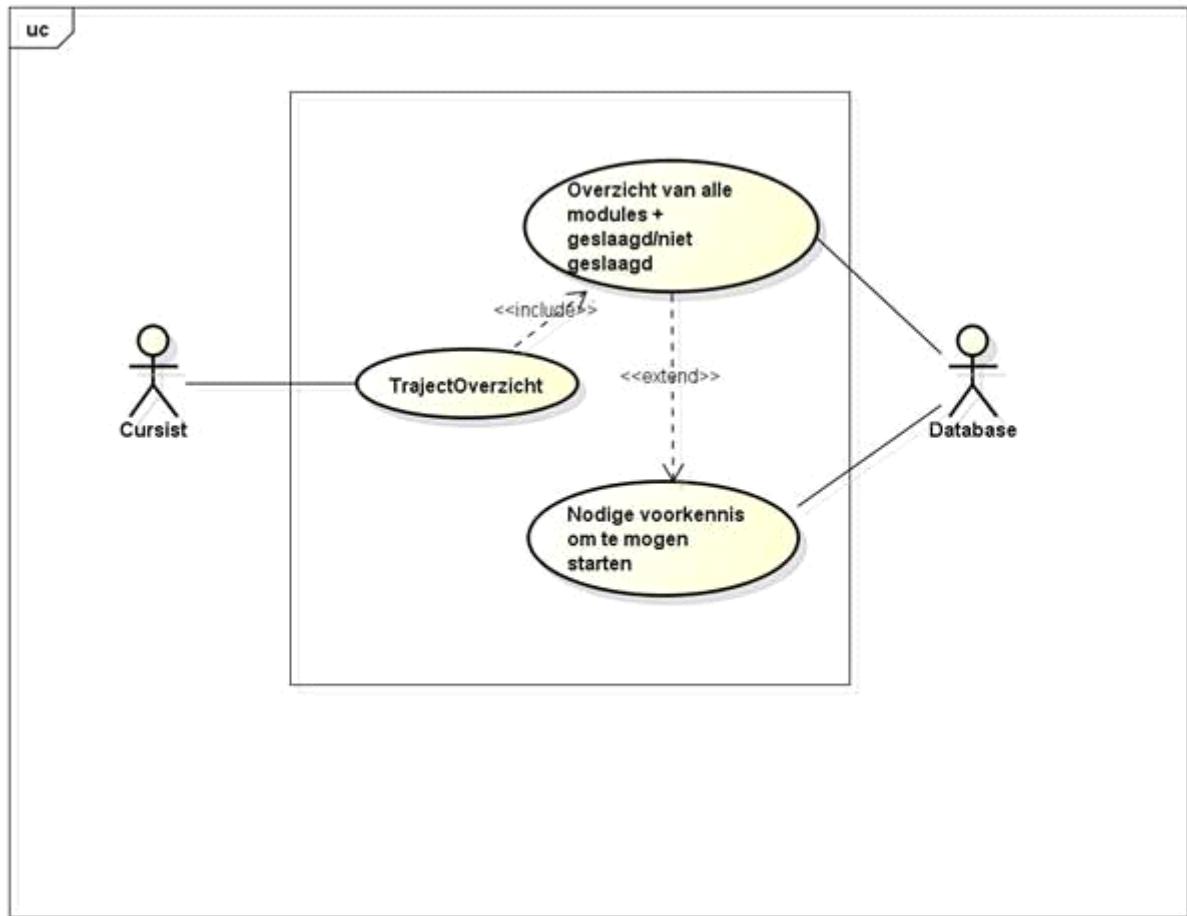
Aannamen: Cursist is ingelogd met cursistnummer.

Beschrijving:

1. Cursist logt in op het systeem.
2. Cursist gaat naar de tegel TrajectOverzicht.
3. Cursist heeft een overzicht van alle modules die hij/zij moet volgen inclusief de aanduiding deze reeds geslaagd is of niet + de nodige voorkennis.

Uitzonderingen: -Cursist heeft geen toegang tot het systeem, ongeldige login.

Resultaat: Cursist komt te weten welke modules al gevuld zijn en welke nog te volgen + welke volgorde van modules hij/zij moet respecteren.



powered by Astah

Puntenoverzicht opvragen:

Naam: Puntenoverzicht opvragen.

Samenvatting: Cursist komt te weten of hij al dan niet geslaagd is voor een module en vraagt zijn punten op.

Actoren: Cursist

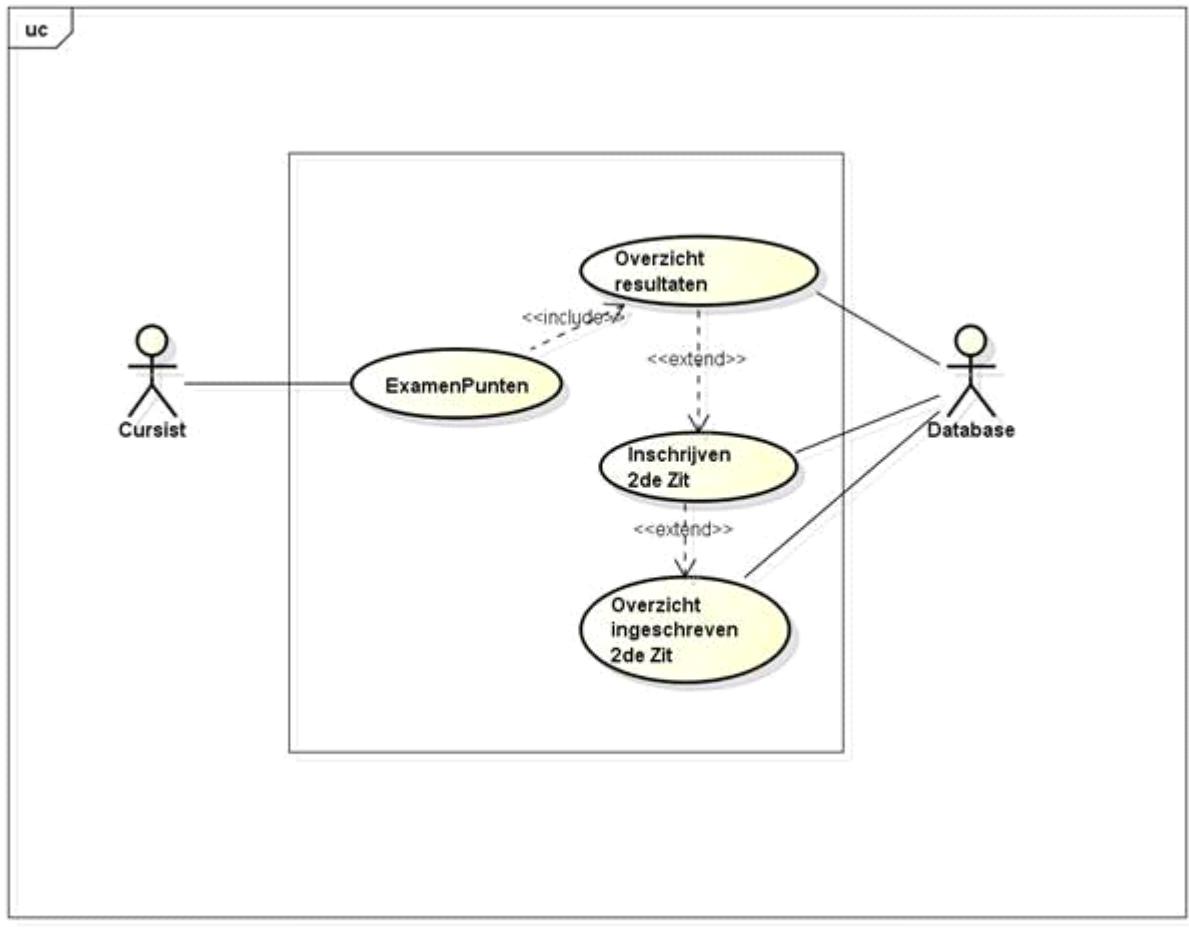
Aannamen: Cursist is ingelogd in het systeem en heeft een volledige module doorlopen.

Beschrijving:

1. Cursist logt in op de website.
2. Cursist gaat naar de tegel ExamenPunten.
3. Cursist krijgt een overzicht van alle afgelegde modules met de behaalde scores voor zowel examens/permanente evaluatie/2de zit/puntentotaal en na deliberaties.

Uitzonderingen: -Punten zijn niet beschikbaar voor aanvraag.

Resultaat: Cursist krijgt een weergave van afgelegde modules met het resultaat.



powered by Astah

Examendata opvragen:

Naam: Examendata opvragen

Samenvatting: Cursist kan kijken wanneer de deliberatie/examen/2dezit plaatsvindt per module.

Actoren: Cursist

Aannamen: Cursist is ingelogd met cursistnummer

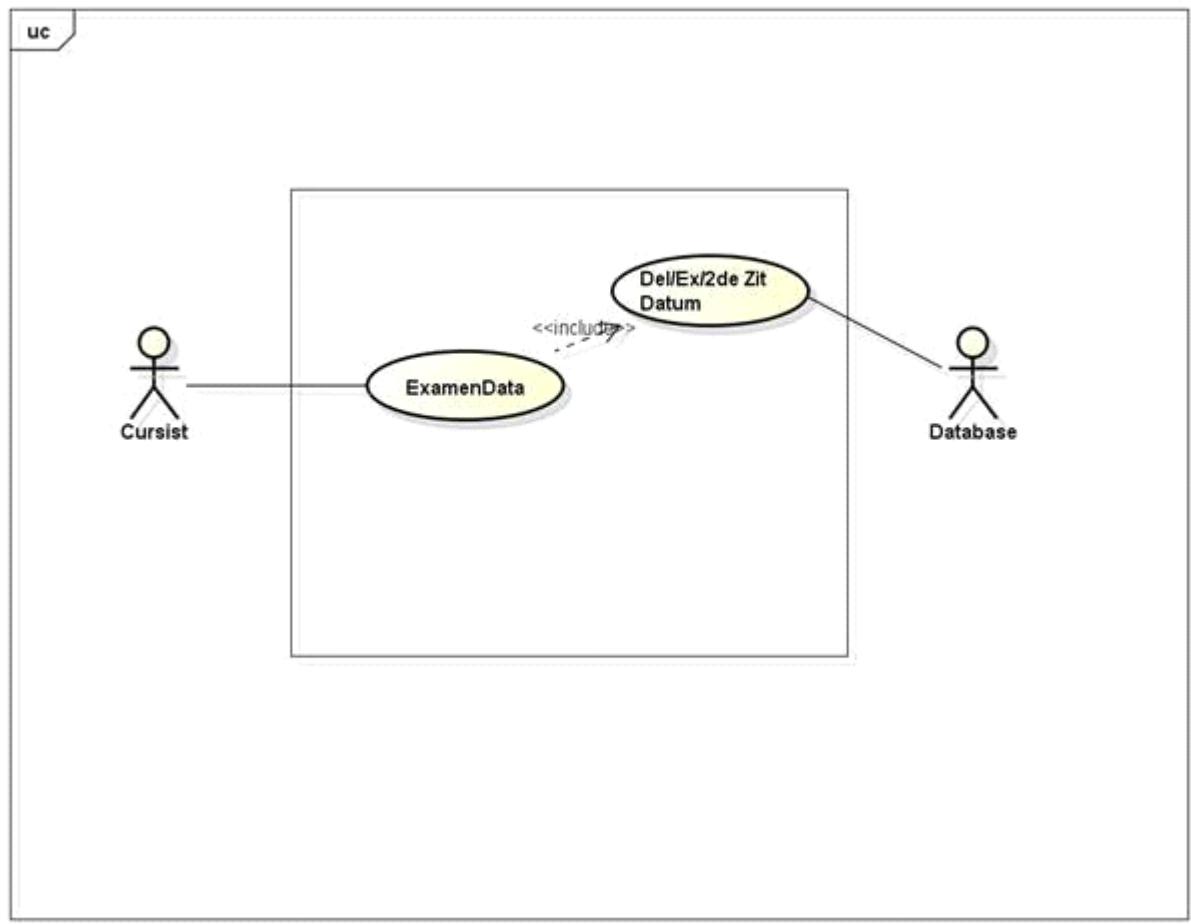
Beschrijving:

1. Cursist logt in op het systeem.
2. Cursist gaat naar de tegel ExamenData.
3. Cursist heeft een overzicht van de deliberatie datums/ examen datums en 2^{de} zit.

Uitzonderingen: -Datums zijn nog niet beschikbaar.

-Foutieve inloggegevens

Resultaat: Cursist krijgt een persoonlijke lijst van examendata per ingeschreven module.



powered by Astah

Inschrijven voor een 2de zit:

Naam: Inschrijven voor een 2de zit.

Samenvatting: Cursist kan zich inschrijven voor een 2de zit van een ingeschreven module.

Actoren: Cursist

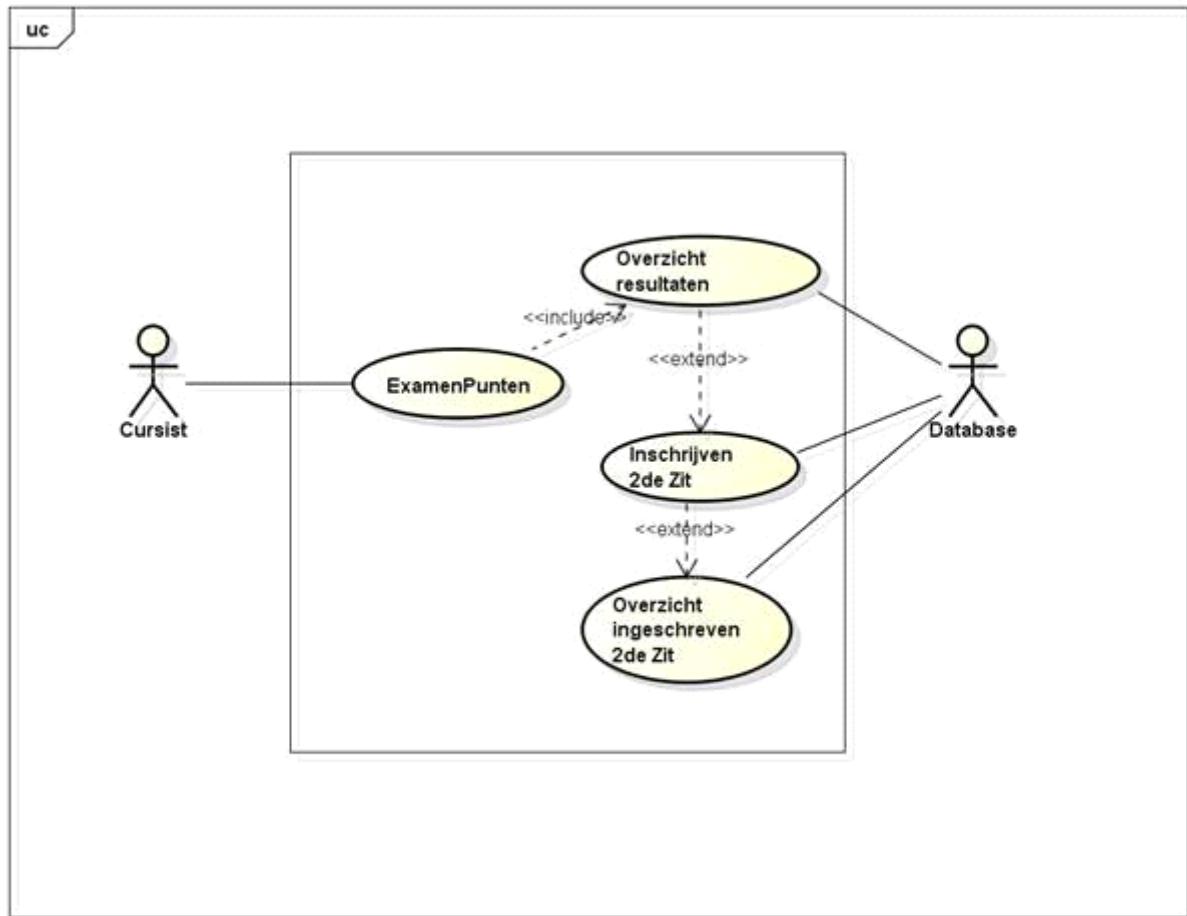
Aannamen: Cursist is ingelogd met cursistnummer en heeft de deliberatie resultaten bekeken.

- Beschrijving:
1. Cursist is ingelogd op het systeem.
 2. Cursist gaat naar de tegel ExamenPunten.
 3. Cursist bekijkt zijn resultaten.
 4. Cursist klikt op de knop inschrijven voor 2de zit.

Uitzonderingen -Er is geen 2de zit georganiseerd.

- : -Cursist heeft zich al ingeschreven in de 2dezit van een module.
- Foutieve inloggegevens.

Resultaat: Cursist is ingeschreven voor een 2dezit.



powered by Astah

Contacteren Trajectbegeleider:

Naam: Informatie opvragen trajectbegeleider.

Samenvatting: Wanneer je een trajectbegeleider wil contacteren kan je gegevens opzoeken zoals email adres en telefoonnummer.

Actoren: Cursist

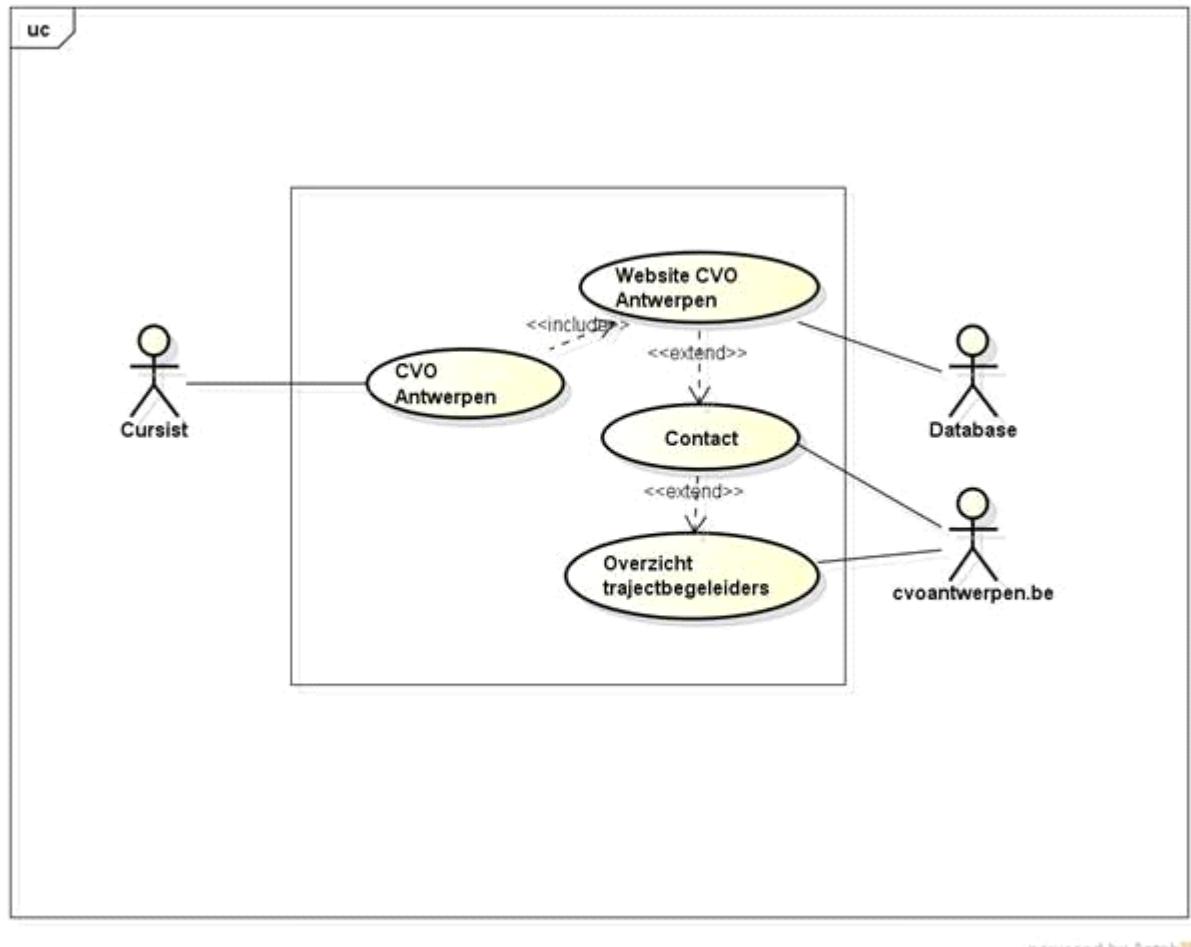
Aannamen: Cursist heeft toegang tot het systeem.

Beschrijving:

1. Cursist logt in op het systeem.
2. Cursist gaat naar de tegel CVO Antwerpen.
3. Cursist wordt gelinkt naar de website van cvo antwerpen waar hij/zij gegevens kan opzoeken.

Uitzonderingen: -Cursist heeft geen toegang tot het systeem, ongeldige login.

Resultaat: Cursist komt het email adres van de trajectbegeleider te weten en kan hem/haar contacteren.



powered by Astah

Contacteren Docent:

Naam: Informatie opvragen over docent.

Samenvatting: Wanneer je een docent wil contacteren kan je gegevens opzoeken zoals email adres en telefoonnummer.

Actoren: Cursist

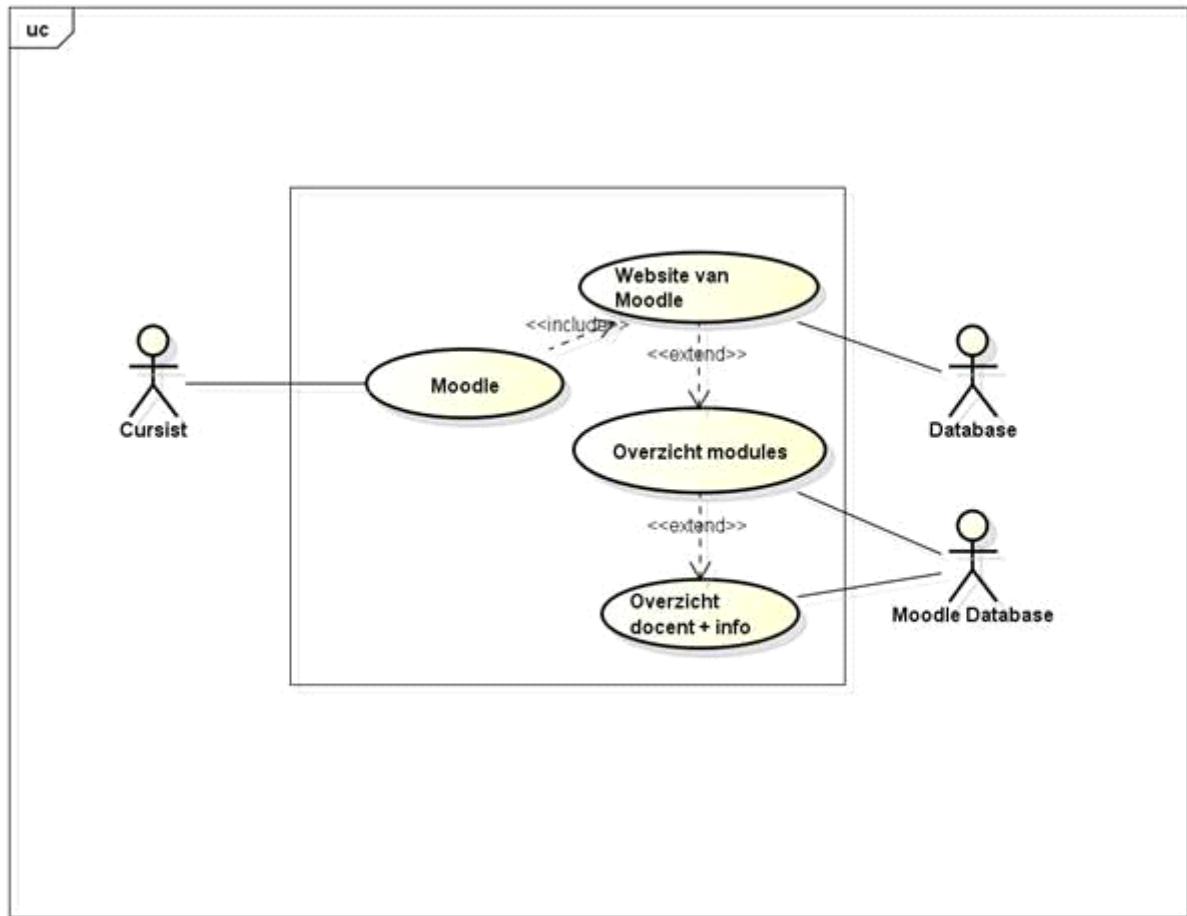
Aannamen: Cursist heeft zich ingelogd met cursistnummer.

Beschrijving:

1. Cursist logt in op het systeem.
2. Cursist gaat naar de tegel Moodle.
3. Cursist wordt gelinkt naar de website van Moodle waar hij/zij de nodige informatie over docenten kan terugvinden.

Uitzonderingen: -Cursist heeft geen toegang tot het systeem, ongeldige login.

Resultaat: Cursist komt het email adres van de docent te weten en kan hem/haar contacteren.



powered by Astah

Contacteren Cursist:

Naam: Informatie opvragen over een cursist.

Samenvatting: Wanneer je een cursist wil contacteren kan je gegevens opzoeken zoals email adres en telefoonnummer.

Actoren: Cursist

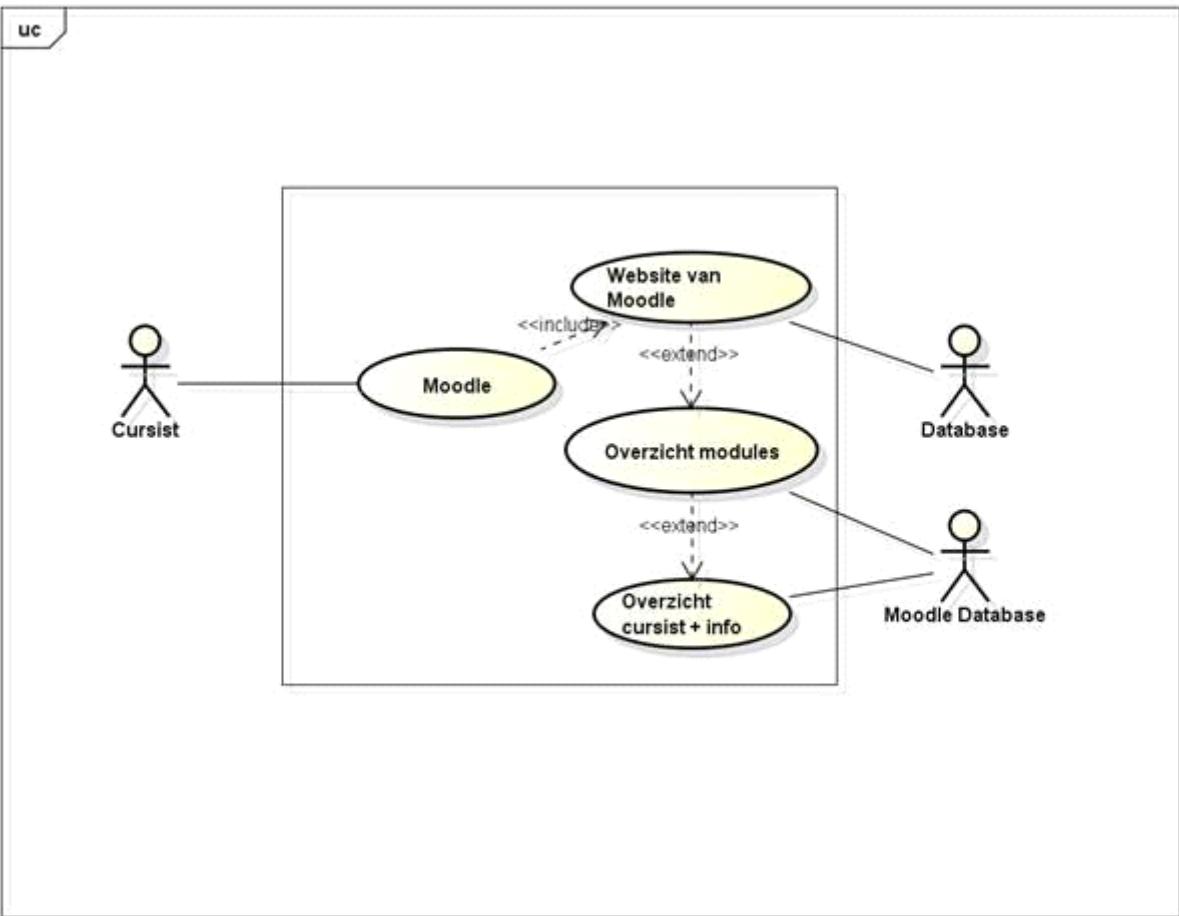
Aannamen: Cursist heeft toegang tot het systeem.

Beschrijving:

1. Cursist logt in op het systeem.
2. Cursist gaat naar de tegel Moodle.
3. Cursist komt op de website van Moodle waar hij/zij medecursisten kan opzoeken.

Uitzonderingen: -Cursist heeft geen toegang tot het systeem, ongeldige login.

Resultaat: Cursist komt het email adres van een cursist te weten en kan hem/haar contacteren.



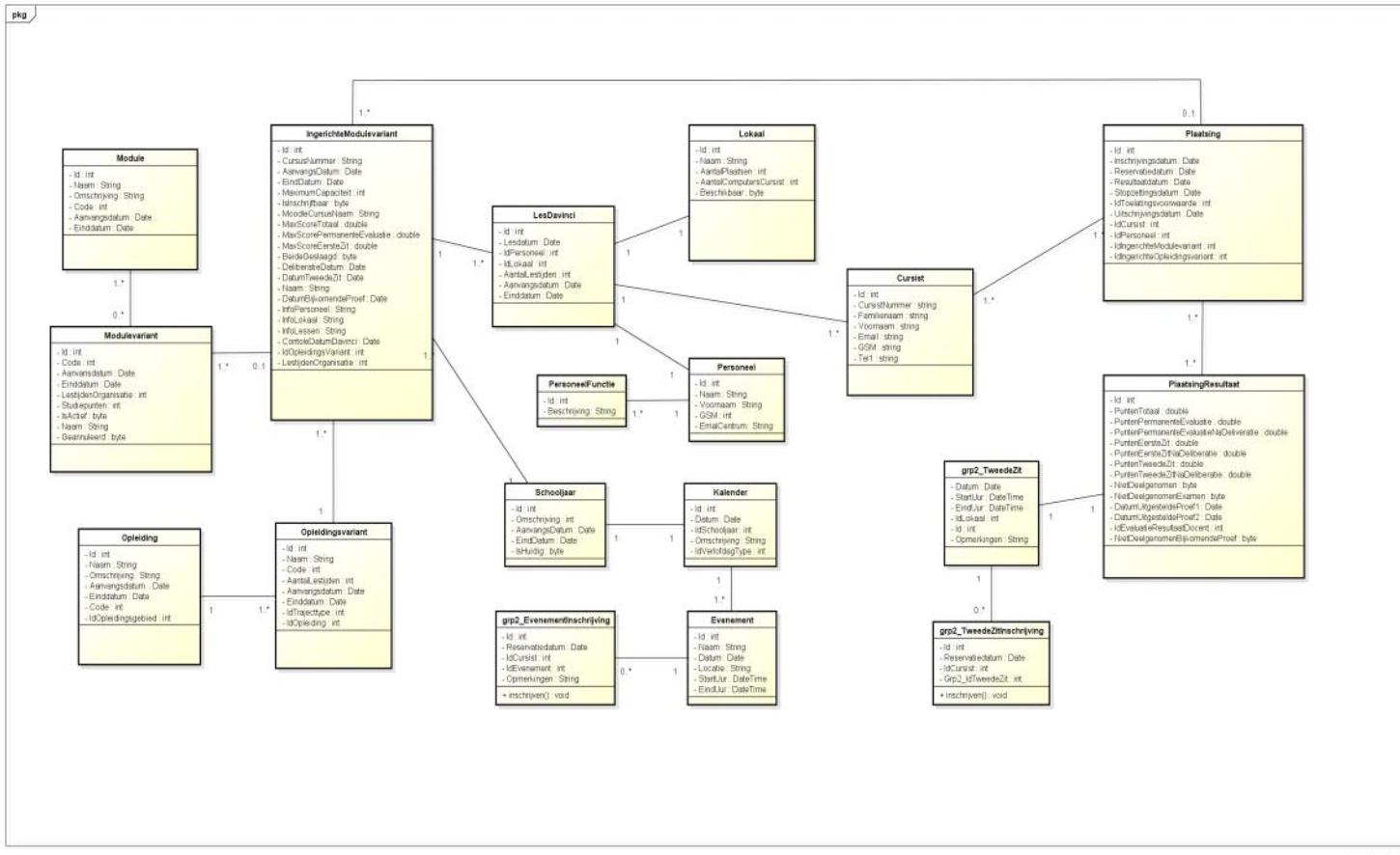
powered by Astah

Klassendiagramm

maandag 1 juni 2015
17:37

Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt



Logo's

maandag 1 juni 2015

17:39

Auteurs:

Sonja Van Rompaey

CVO

Het CVO Antwerpen logo komt van de school website en wordt gebruikt als watermerk in onze webapplicatie.



Onderstaand Mobile CVO logo is aangemaakt met behulp van Adobe Illustrator. Dit logo zal links bovenaan op elke pagina bevinden.

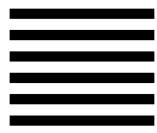


Navigatieknoppen

De navigatieknoppen in onze applicatie zijn afkomstig van IconFinder. Deze iconen hebben een grootte als een letter dus neemt deze weinig geheugen in beslag.

De iconen zijn:

- Hamburger menu: deze leidt de cursist terug naar het tegelmenu;
- Pijl "Links" : deze leidt de cursist naar de vorige keuzepagina van het tegelmenu;
- Pijl "Rechts": deze leidt de cursist naar de volgende keuzepagina van het tegelmenu.



Mockups

maandag 1 juni 2015

17:29

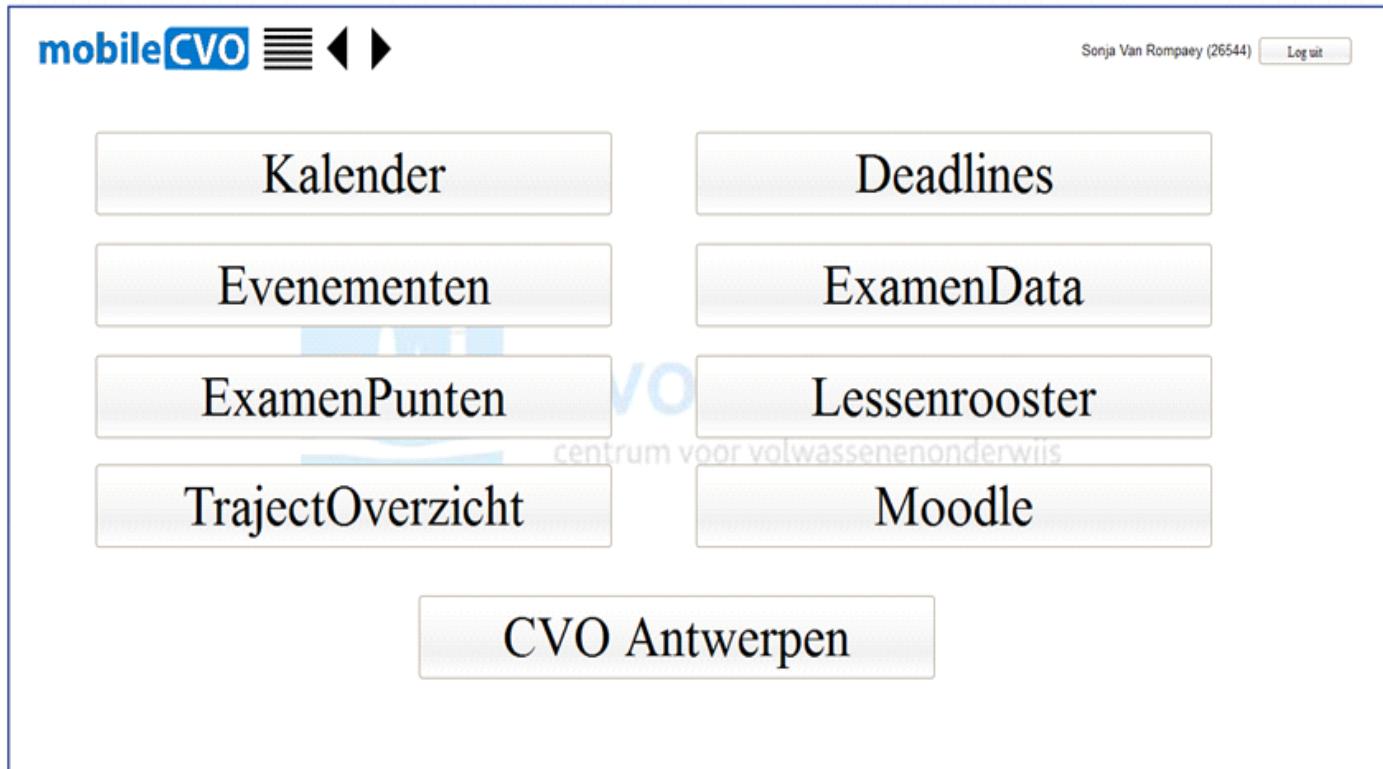
Auteurs:

Sonja Van Rompaey
Mohamed Amajoult

Homepage

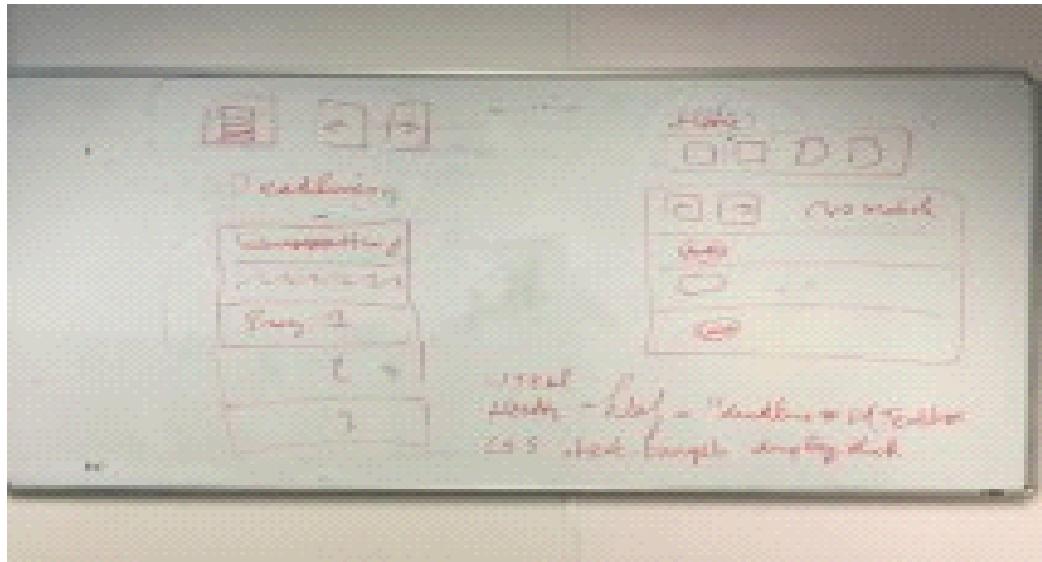
De Homepage bestaat uit:

- Een navigatie, waarmee je terug kan keren naar de homepage of per sectie kan verspringen;
- Een tegel voor elk item;
- Een login.

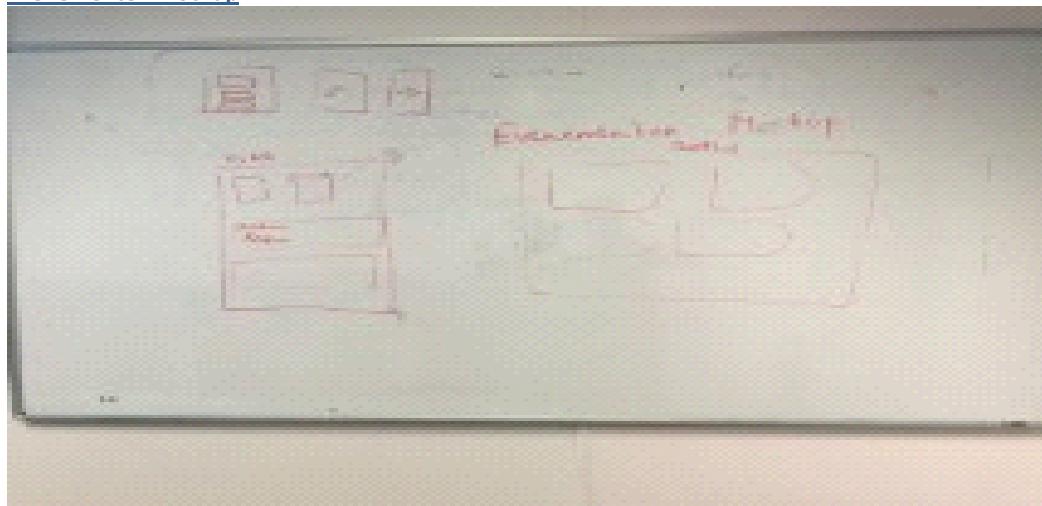


Deelpagina's

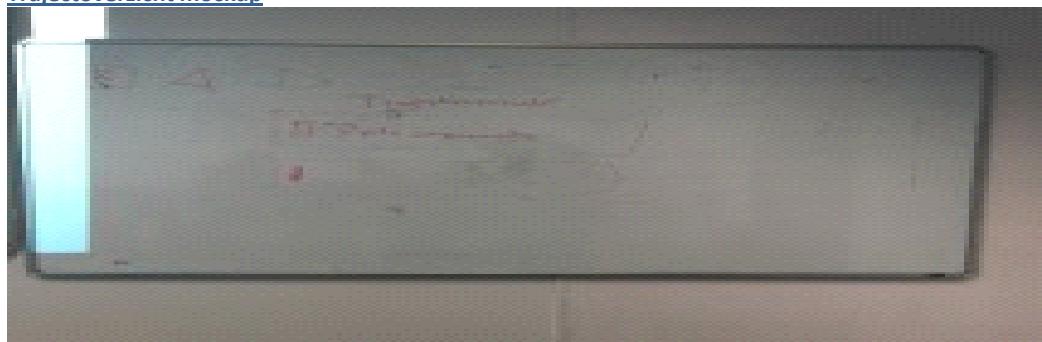
Deadline mockup



Evenementen mockup



Trajectoverzicht mockup



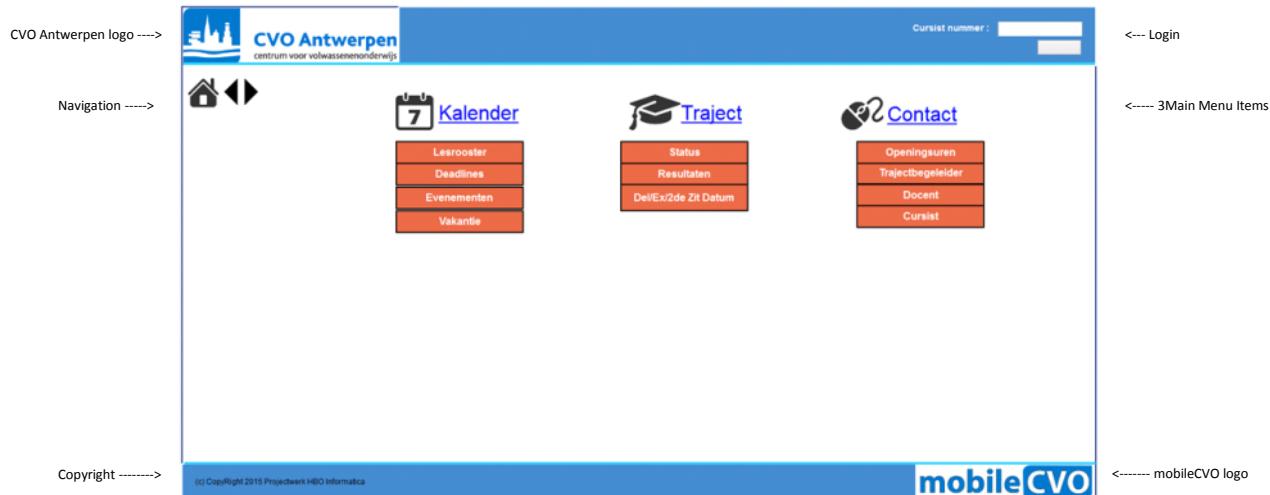
Unused Mockup

woensdag 3 juni 2015
16:10

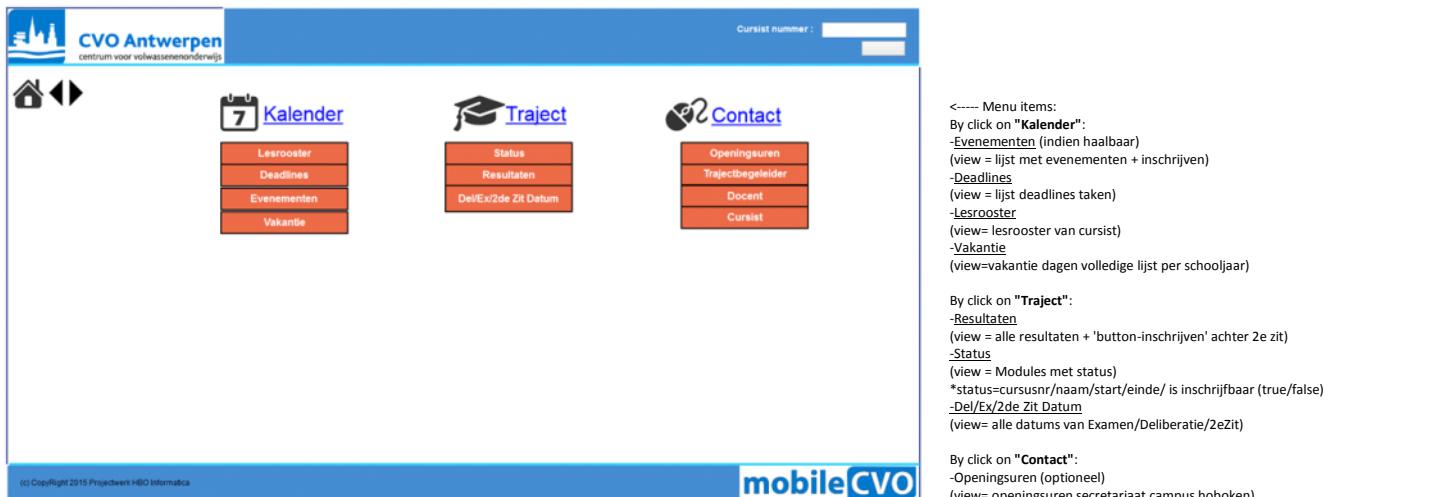
Auteurs:
Sonja Van Rompaey

Waarom we deze layout niet hebben gebruikt:

Onderstaande mockup is niet gebruikt omdat deze functionaliteit ervan minder toepasbaar is voor mobiele apparaten.
Daarom is beslist om deze layout te vereenvoudigen in een tegel structuur, en deze toe te passen aangezien die beter past voor deze opdracht.



Logo's:



Tutorial, making a consistent layout with asp.net Razor pages:
<http://www.asp.net/web-pages/overview/ui-layouts-and-themes/3-creating-a-consistent-look>

CVO Antwerpen
centrum voor volwassenenonderwijs

Cursist nummer:

[Kalender](#) [Traject](#) [Contact](#)

Lesrooster:

Cursusnr	Module	Docent	Lokaal	Dag	Datum	Van	Tot
12405	Projectwerk Informatica (TV)	Inghelbrecht Joseph	B16	donderdag	02-04-15	10:45	12:15
12453	Netwerkbeheer 1 TV	Charlier Alphons	B11	dinsdag	21-04-15	13:15	17:00

[Lesrooster opslaan?](#)
[Lesrooster afdrukken?](#)

(c) Copyright 2015 Projectwerk HBO Informatica

mobileCVO

CVO Antwerpen
centrum voor volwassenenonderwijs

Cursist nummer:

[Kalender](#) [Traject](#) [Contact](#)

Evenementen:

Zoeken:

Naam	Datum	Locatie	Start	Ende	Inschrijven
IT DAY	25-05-2015	CVO Antwerpen	10:00	16:00	<input type="button" value="Inschrijven"/>

(c) Copyright 2015 Projectwerk HBO Informatica

mobileCVO

CVO Antwerpen
centrum voor volwassenenonderwijs

Cursist nummer:

[Kalender](#) [Traject](#) [Contact](#)

Vakantie:

Zoeken:

Omachtijd	Van	Tot
Pasvakantie	06-04-2015	19-04-2015

(c) Copyright 2015 Projectwerk HBO Informatica

mobileCVO

CVO Antwerpen
centrum voor volwassenenonderwijs

Cursist nummer :

[Home](#) [Kalender](#) [Traject](#) [Contact](#)

Deadlines in de komende 2 weken

13/5/2015	Netwerkbeheer 1 Programmeren 5	Oefdracht 2 Taak 2: Suney Customers
17/5/2015	Programmeren 5	Taak 3: Maintain Products

Programmeren 5

- Task 1: Display Customers Score: 87/100
- Task 2: Survey Customers Deadline: 13/5/2015
- Task 3: Maintain Products Deadline: 17/5/2015

Projectwerk Informatica

- Project requirements deel 1 - Cursistscore: 93/100
- Insturen Functionele Specificaties Score: 16/100
- Use Cases Score: 54/100
- Insturen KlasseDiagram Deadline: 19/5/2015
- Insturen ER-Diagram Deadline: 21/5/2015
- Code Deel 1 Deadline: 27/5/2015

Netwerkbeheer 1

- Oefdracht 1 Score: 42/100
- Oefdracht 2 Deadline: 13/5/2015
- Oefdracht 3 Deadline: 21/5/2015
- Oefdracht 4 Deadline: 28/5/2015

(c) CopyRight 2015 Projectwerk HBO Informatica

mobileCVO

CVO Antwerpen
centrum voor volwassenenonderwijs

Cursist nummer :

[Home](#) [Kalender](#) [Traject](#) [Contact](#)

Status :

Voldaan	Code	Module	Lestijden	Status	Inschrijfbaar	Plaatsen	Voorkeur
<input checked="" type="checkbox"/>	A1	Basiskennis	60	-	x	x	-
<input checked="" type="checkbox"/>	A2	Softwarepakketten	40	-	x	x	-
<input checked="" type="checkbox"/>	A3	Communicatie en organisatietechnieken	40	-	x	x	-
<input checked="" type="checkbox"/>	A4	Multimedia 1	60	-	x	x	-
<input checked="" type="checkbox"/>	A5	Programmeren 1 Cf	120	-	x	x	-
<input checked="" type="checkbox"/>	A6	Programmeren 2 Cf (of Powershell)	60	-	x	x	-
<input checked="" type="checkbox"/>	A7	IT-organisatie	40	-	x	x	-
<input checked="" type="checkbox"/>	A8	Databanken	60	-	x	x	-
<input checked="" type="checkbox"/>	A9	Datacommunicatie en netwerken	60	-	x	x	-
<input type="checkbox"/>	A10	Netwerkbeheer 1	60	-	x	x	A9
<input checked="" type="checkbox"/>	A11	Bettingsystemen	60	-	x	x	A1

(c) CopyRight 2015 Projectwerk HBO Informatica

mobileCVO

CVO Antwerpen
centrum voor volwassenenonderwijs

Cursist nummer :

[Home](#) [Kalender](#) [Traject](#) [Contact](#)

Resultaten:

Cursusnr	Module	PuntenTotaal	Evaluatie	Twede Zt	Inschrijven 2de Zt	Opmerking
12405	Projectwerk Programmeren	x	x	x	inschrijven	x
12453	Netwerkbeheer 1 TV	x	x	x	inschrijven	x
11628	Programmeren3 TV	75%	60%	x	inschrijven	geslaagd
11627	Programmeren4 TV	x	x	x	inschrijven	bijkomende proef
11619	Programmeren3 TV	75%	60%	x	inschrijven	geslaagd
11618	Analyse TV	45%	50%	x	inschrijven	2de Zt

(c) CopyRight 2015 Projectwerk HBO Informatica

mobileCVO

 **CVO Antwerpen**
centrum voor volwassenenonderwijs

Cursist nummer :

  [Kalender](#)  [Traject](#)  [Contact](#)

Examen - Deliberatie - Tweede Zit Datums :

Cursusnr	Module	Examen Datum	Deliberatie Datum	Tweede Zit
11627	Programmerend TV	26/01/2015	12/02/2015	24/02/2015
11628	Programmerend TV	25/02/2015	4/03/2015	26/03/2015
12495	Projectwerk Programmeren	18/06/2015	26/06/2015	31/08/2015

(c) CopyRight 2015 Projectwerk HBO Informatica

mobileCVO

 **CVO Antwerpen**
centrum voor volwassenenonderwijs

Cursist nummer :

  [Kalender](#)  [Traject](#)  [Contact](#)

Contact Trajectbegeleider :

Filter: Zoeken:

Naam	Gsm	Tel	Email	Functie
Ivan Ribeys	0477 96 82 43	03 369 06 99	ivan.ribeys@osantwerpen.be	Trajectbegeleider

(c) CopyRight 2015 Projectwerk HBO Informatica

mobileCVO

 **CVO Antwerpen**
centrum voor volwassenenonderwijs

Cursist nummer :

  [Kalender](#)  [Traject](#)  [Contact](#)

Contact Cursist :

Filter: Zoeken:

Naam	Gsm	Tel	Email	Functie
Sonja Van Rompaey	0497 29 06 34	-	sonja.vanrompaey@hotmail.com	Cursist

(c) CopyRight 2015 Projectwerk HBO Informatica

mobileCVO



[Kalender](#)

[Traject](#)

[Contact](#)

Contact Docent:

Filter:

Zoeken:

Naam	Gsm	Tel	Email	Functie
Jef Ingelbrecht	-	03 830 41 05	jef.ingelbrecht@cvoantwerpen.be	Docent

ERD

woensdag 3 juni 2015
15:10

Auteurs:

Sonja Van Rompaey
Mohamed Amajoult

Administratix db

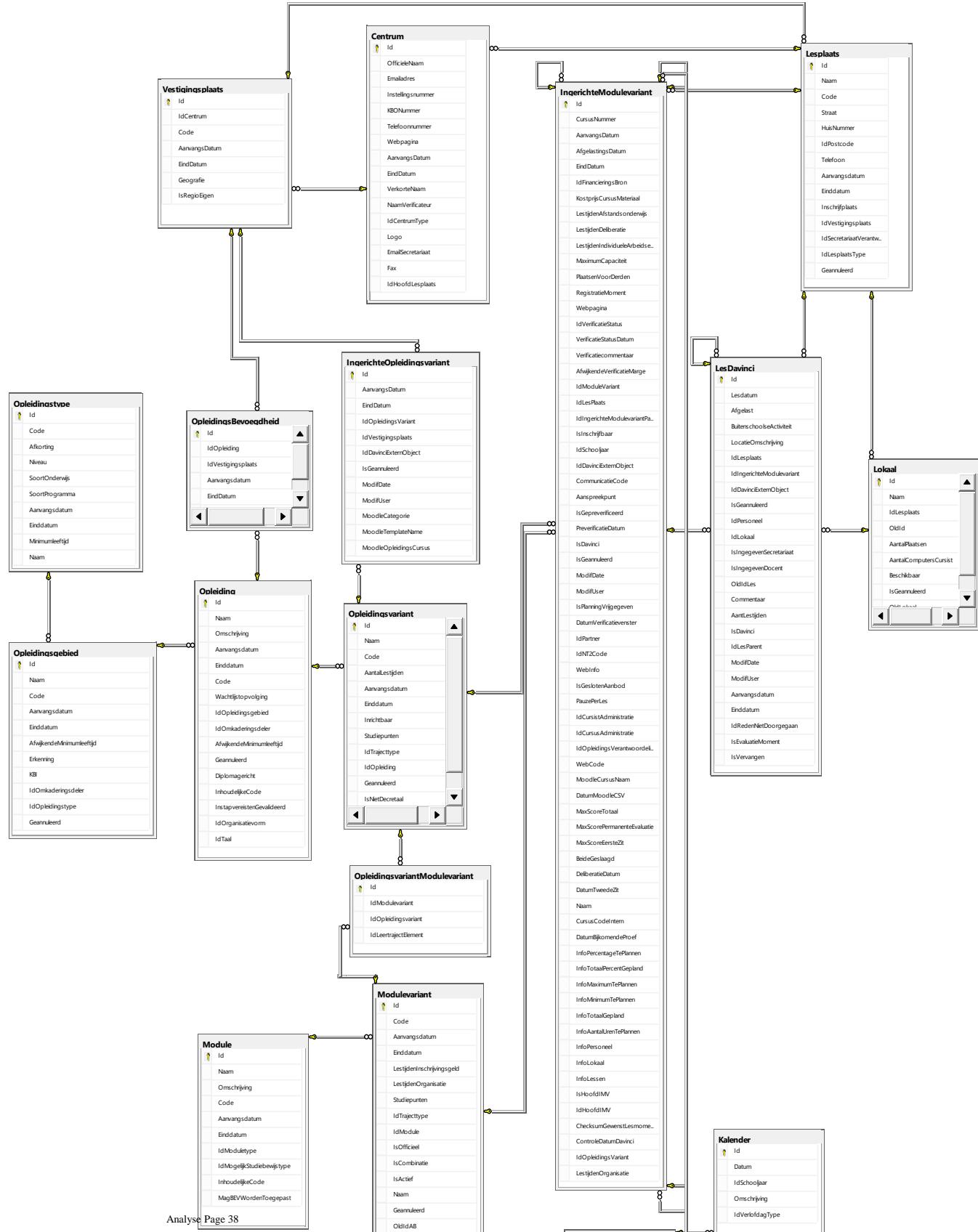
Voor dit project hebben we een database ter beschikking gekregen.

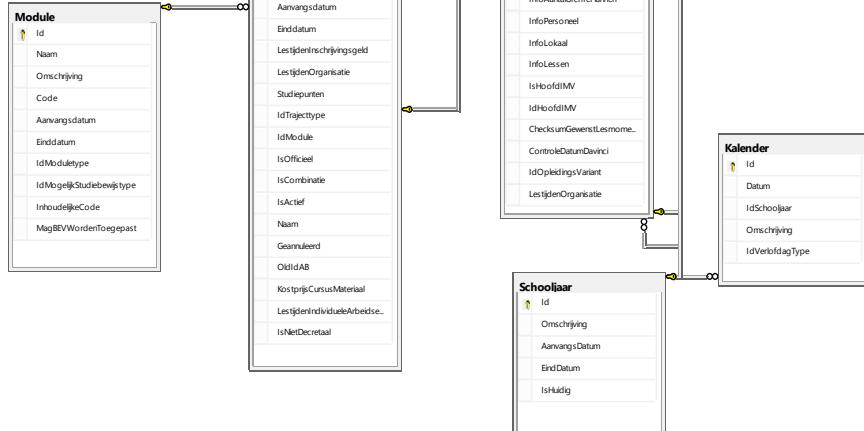
Deze betreft een kopie van de bestaande, namelijk de Administratix database.

Onderstaande diagramma zijn rechtstreeks uit de Administratix database gehaald.

Cursist Plaatsing Diagram

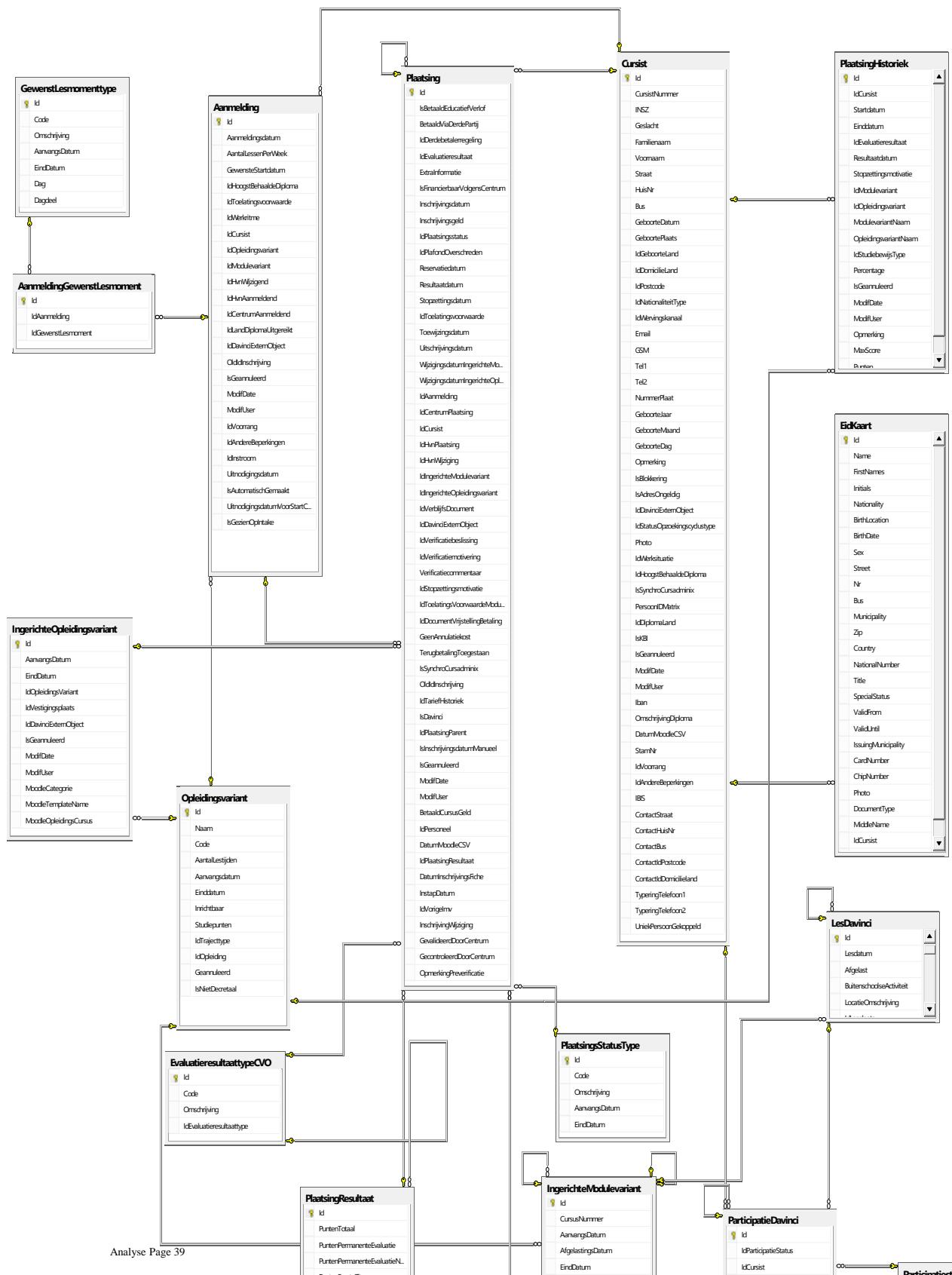
Dit diagram beschrijft de relatie van tabellen die doorlopen worden wanneer een cursist zich aanmeldt op de school en zich wenst in te inschrijven.

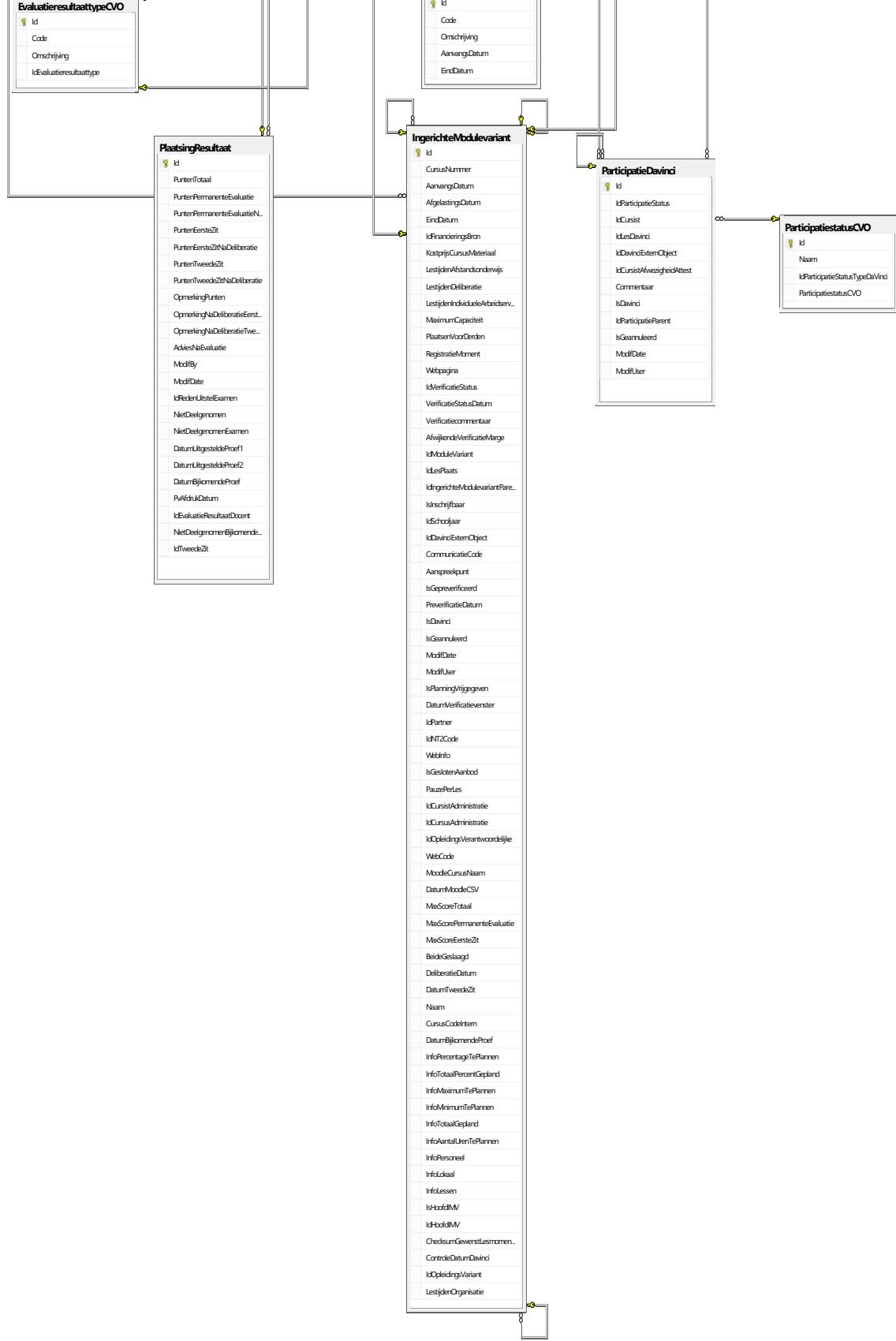




Aanbod Diagram

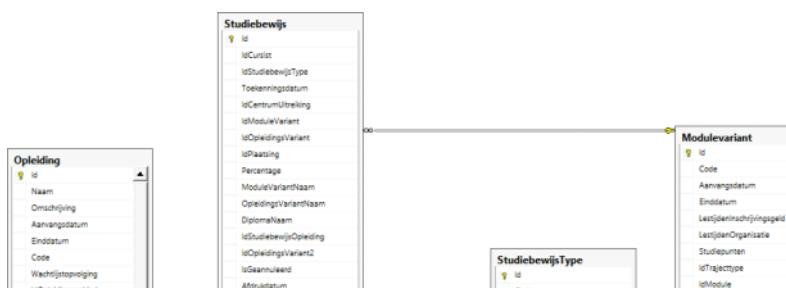
Onderstaand diagram beschrijft welke tabellen gelinkt zijn aan elkaar om modules en informatie over opleidingen en lessen op te vragen.

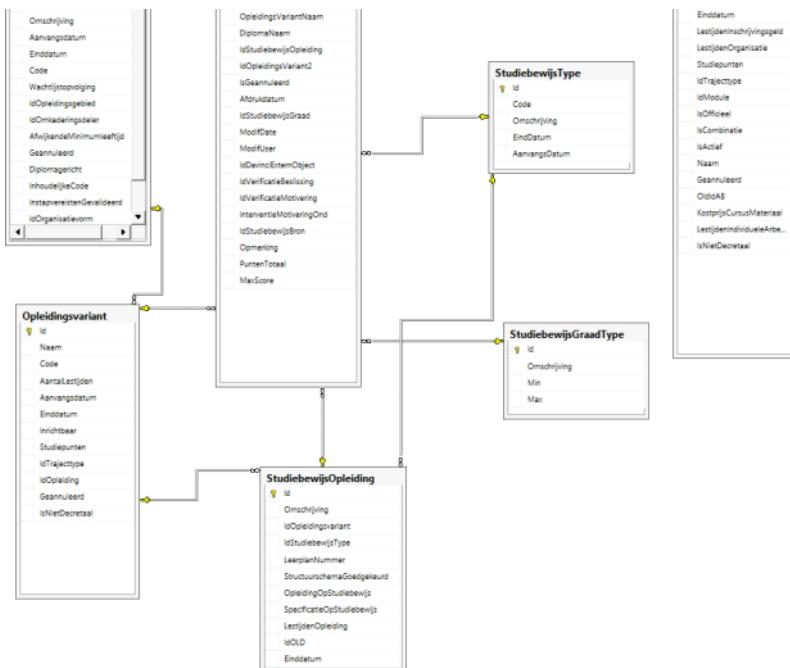




Studiebewijs Diagram

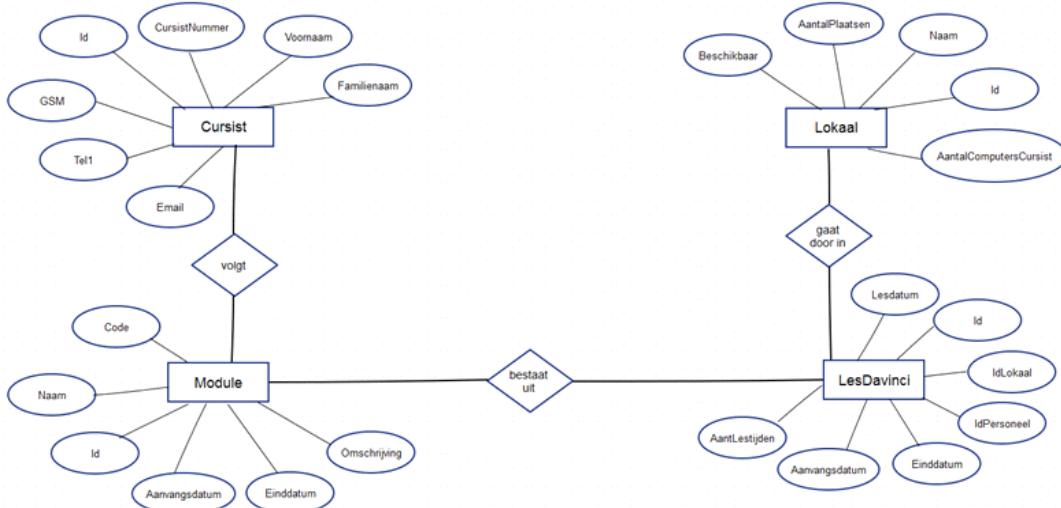
Diagram dat informatie geeft over behaalde certificaten voor modules per opleiding.





Relaties tussen tabellen

Een concreet voorbeeld van relaties tussen de tabellen:
Cursist, die de entiteiten: Id/ Cursistnummer/ Voornaam/ Familienaam/ Email en GSM heeft.
Is ingeschreven voor een Module, die bestaat uit een LesDavinci, die plaats heeft in een lokaal.



Tests

woensdag 3 juni 2015
16:00

Backend tests

De backend code is onzichtbaar voor de gebruiker. Deze bevindt zich in de data access laag.
Hierin wordt data opgeroepen, het uitgebreid testen en nakijken van deze code is door mij gebeurd.

Bijvoorbeeld:

Oproepen van een lesrooster:

```
-- Mohamed Amajoutt
-- 27/04/2015

--SP geeft een lessenrooster weer adhv de ingegeven cursistnummer

use Administratix_cursist
go

if exists (select 1 from sysobjects where name = 'grp2
_SelectAllLesDavinciByCursistNummer' AND type = 'P') -- P = procedure

begin
    drop proc grp2_SelectAllLesDavinciByCursistNummer
end
go

create procedure grp2_SelectAllLesDavinciByCursistNummer
(
    @CursistNummer int
)

as
begin
select
    IngerichteModulevariant.CursusNummer as Cursusnummer
    ,LesDavinci.Lesdatum as Datum
    ,Personeel.Naam + ' ' + Personeel.Voornaam as Docent
    ,Lesplaats.Naam as Campus
    ,LesDavinci.LocatieOmschrijving as Lokaal
    ,IngerichteModulevariant.Naam as Module
    ,LesDavinci.AanvangsDatum as Van
    ,LesDavinci.Einddatum as Tot
from LesDavinci
inner join IngerichteModulevariant on LesDavinci.IdIngerichteModulevariant =
IngerichteModulevariant.Id
inner join Lesplaats on LesDavinci.IdLesplaats = LesPlaats.Id
inner join Personeel on LesDavinci.IdPersoneel = Personeel.Id
inner join Lokaal on LesDavinci.IdLokaal = Lokaal.Id
inner join Plaatsing on Plaatsing.IdIngerichteModulevariant = IngerichteModulevariant.id
inner join Cursist on Plaatsing.IdCursist = Cursist.Id
where Cursist.CursistNummer = @CursistNummer and LesDavinci.Aanvangsdatum >= GETDATE()
order by LesDavinci.Lesdatum, LesDavinci.Aanvangsdatum
end
```

Waar worden al deze gegevens uit gehaald?

- Ⓐ Uit de tabellen:

IngerichteModulevariant, LesDavinci, Personeel en Lesplaats.

Frontend tests

De gebruikersinterface, deze bevindt zich in de presentatielaag.

Wat is het beste voor een gebruiker?

- Bijvoorbeeld: Zo weinig mogelijk input velden hebben.

Hoe moet een applicatie eruit zien?

- Bijvoorbeeld: Overzichtelijk met een goede navigatie.

Werkt het met verschillende browsers?

- Bijvoorbeeld: testen met Internet Explorer, FireFox en Google Chrome.

Concrete toepassing:

Onderstaande zie je de page Deadlines.

Er staat een deadline in voor 1 januari 1970, een deadline in het verleden kan uiteraard niet.

The screenshot shows a web-based application interface for managing project deadlines. At the top, there's a header with the mobileCVO logo and navigation icons. Below the header, a red banner displays the text "Komende 2 weken". Underneath this, a message states "Je momenteel hebt geen deadlines in de komende 2 weken." To the right, a large box lists various project tasks with their respective deadlines:

Deadline	Task
1/1/1970 00:00	Logboeken cursisten
25/6/2014 23:55	Project Requirements deel 1 - Cursisten
3/7/2014 23:55	Insturen Functionele specificaties
10/7/2014 23:55	Use Cases
24/7/2014 09:30	Inleveren klassediagram
24/7/2014 23:55	Inleveren ER-diagram
31/7/2014 22:55	Code deel 1

Auteurs:

Sonja Van Rompaey

Technologieën

donderdag 4 juni 2015
11:18

Verantwoording

Gebruikte technologieën zijn overgenomen van de school.

Aangezien de school een database heeft in MSSQL en reeds gebruikt maakt van de ASP.NET toepassing hebben we voor ons project deze overgenomen.

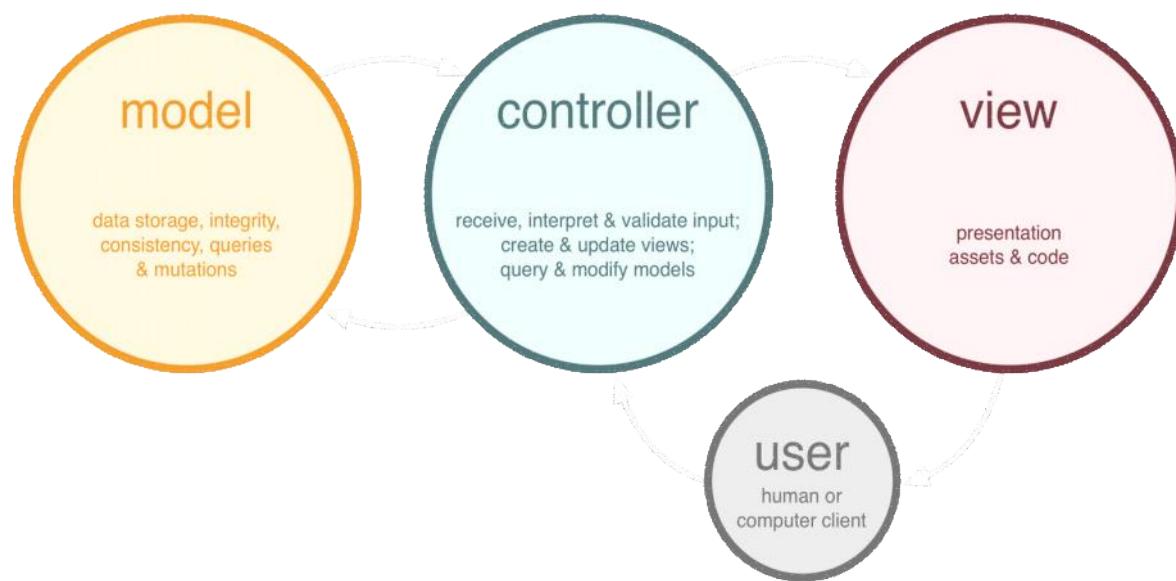
Verder hebben we gekozen voor het gebruik van HTML en CSS met de Razor syntax omdat deze perfect implementeerbaar is voor dit project.

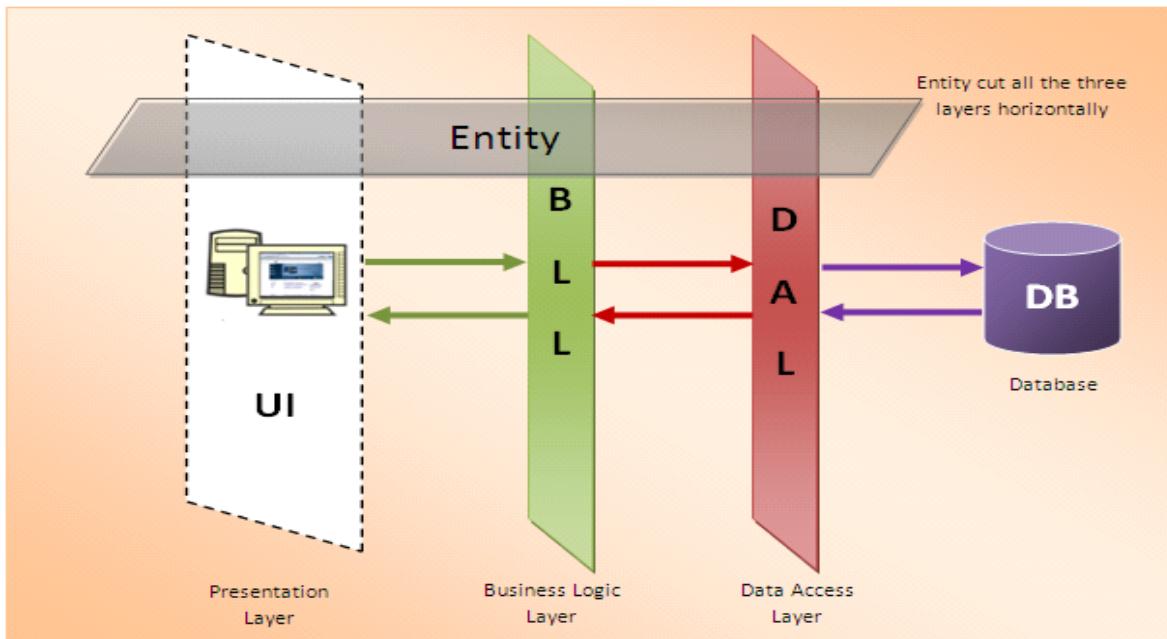
Inleiding

Eerst hebben we de analyse uitgewerkt samen met de groep en dan hebben we het werk verdeeld per use case.

We hebben alle functies geprogrammeerd in ASP.NET met Razor syntax. In de View gebruiken we HTML5, CSS3 om de webpagina's weer te geven en zodanig responsive te maken. We hebben telkens een MasterPage die de header en de footer bevat, omdat deze in elke pagina voorkomen. Elke use case uit de analyse die mij werd toegekend om uit te werken heb ik telkens uitgewerkt in 3 lagen: BLL, DAL, UI. Ook hebben we gebruik gemaakt van het MVC-patroon: Model, View, Controller voor elke use case.

Hierdoor hebben we telkens overzichtelijker, consistentere en performantere code uitgewerkt.





[Basic 3-Tire architecture]

Applicatie

Framework

.NET

Het project is opgebouwd in Visual Studio versie 2013 waar we een ASP.NET Empty web Application hebben aangemaakt dat gebruik maakt van de programmeertaal C#.

Er is gebruik gemaakt van het .NET Framework 4.5.

```
<system.web>
    <compilation debug="true" targetFramework="4.5"/>
    <httpRuntime targetFramework="4.5"/>
</system.web>
```

BLL

Hierin worden klassen aangemaakt die objecten uit de Administratix databank controleert om later op te roepen en te gebruiken.

- ④ Bijvoorbeeld: de klasse LesDavinci die nodig is om een lesrooster te tonen voor een cursist.

```

    /// <summary>
    /// Class definition LesDavinci
    /// Used in DAL.LesDavinci
    /// </summary>
16 references
public class LesDavinci
{
    2 references
    public string Cursusnummer { get; set; }
    4 references
    public DateTime Datum { get; set; }
    2 references
    public string Campus { get; set; }
    3 references
    public string Module { get; set; }
    2 references
    public string Docent { get; set; }
    2 references
    public string Lokaal { get; set; }
    3 references
    public DateTime Aanvangsdatum { get; set; }
    2 references
    public DateTime Einddatum { get; set; }

}

```

DAL

Deze laag gaat al de nodige data uit de Administratix databank halen.

```

Web.config
MobileCVO.DAL.IDal<TypeEntity>
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Web;

namespace MobileCVO.DAL
{
    6 references
    interface IDal<TypeEntity>
    {
        0 references
        string Message {get;}
        //bool Update(TypeEntity entity);
        0 references
        int Insert(TypeEntity entity);
        //bool Delete(int Id);
        7 references
        TypeEntity SelectOne(int id);
        0 references
        List<TypeEntity> SelectAll();
        9 references
        List<TypeEntity> SelectAllByCursistNummer(int cursistNummer);
    }
}

namespace Administratix.DAL
{
    1 reference
    public class StatusTraject
    {
        1 reference
        public static List<BLL.StatusTraject> SelectStatusByCursistNummer(int cursistNummer)
        {
            List<BLL.StatusTraject> resultaten = new List<BLL.StatusTraject>();

            SqlConnection connection = new SqlConnection();
            connection.ConnectionString =
                System.Configuration.ConfigurationManager.

```

Model

In de model klasse wordt alle nodige data getoont.

```
AdministratixModel.cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Administratix
{
    public class Model
    {
        public class LesDavinci
        {
            public static List<BLL.LesDavinci> LesDavinciSelectAllByCursistNummer(int cursistNummer)
            {
                return DAL.LesDavinci.SelectAllByCursistNummer(cursistNummer);
            }

            public static List<BLL.LesDavinci> LesDavinciSelectAllByCursistNummerAndDates(int cursistNummer, DateTime begin, DateTime einde)
            {
                return DAL.LesDavinci.SelectAllByCursistNummerAndDates(cursistNummer, begin, einde);
            }

            public static List<BLL.LesDavinci> LesDavinciSelectAllByCursistNummerWithVakantieDagen(List<BLL.LesDavinci> lessenrooster, List<BLL.Kalender> vakantiedagen)
            {
                foreach (BLL.Kalender k in vakantiedagen)
                {
                    if (!k.Omschrijving.Equals(""))
                    {
                        BLL.LesDavinci dag = new BLL.LesDavinci();
                        dag.Datum = k.Datum;
                        dag.Module = k.Omschrijving;

                        lessenrooster.Add(dag);
                    }
                }
                return lessenrooster.OrderBy(o => o.Datum).ThenBy(o => o.Aanvangstijd).ToList();
            }
        }
    }
}
```

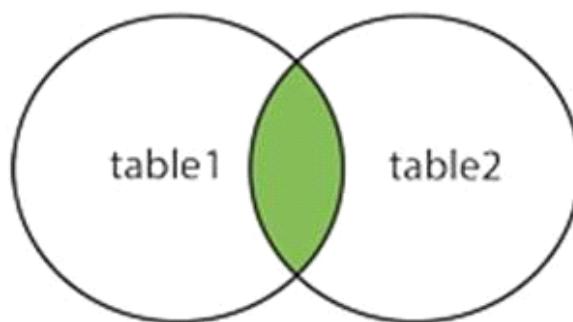
Stored Procedures

De stored procedures zijn nodig om gegevens uit de databank te halen.

Een groot deel is door mezelf geschreven en later nog geupdate door derden.

Belangrijk bij het schrijven hiervan was het gebruik van joins, deze zijn nodig om data uit andere tabellen te kunnen gebruiken.

INNER JOIN



Oplossing:

Stored Procedure om een cursist een geplande tweede zit te laten zien.

```
--author SVR 3/05/2015
--updated JR on 9/05/2015

--stored procedure
--TweedeZit opvragen

use Administratix_cursist
go

if exists (select 1 from sysobjects where name = 'grp2_SelectTweedeZitByCursistNummer' AND type = 'P') -- P = procedure
begin
    drop proc grp2_SelectTweedeZitByCursistNummer
end
go

create procedure grp2_SelectTweedeZitByCursistNummer
(
@CursistNummer int
)

as
begin

select
Datum as Datum,
IngerichteModulevariant.Naam as Module,
StartUur as Van,
EindUur as Tot,
Lokaal.naam as Lokaal,
PlaatsingResultaat.PuntenTotaal as Punten,
Ingeschreven
from TweedeZit
inner join PlaatsingResultaat as PlaatsingResultaat on PlaatsingResultaat.IdTweedeZit = TweedeZit.Id
inner join Plaatsing as Plaatsing on Plaatsing.IdPlaatsingResultaat = PlaatsingResultaat.Id
inner join Cursist as Cursist on Plaatsing.IdCursist = Cursist.Id
inner join IngerichteModulevariant as IngerichteModulevariant on IngerichteModuleVariant.Id =
plaatsing.IdIngerichteModuleVariant
inner join Lokaal as Lokaal on Lokaal.Id = TweedeZit.IdLokaal
where Cursist.CursistNummer = @CursistNummer
end
```

Commentaar:

- Er wordt in dit script ook data gehaald uit de tabellen:
 - IngerichteModulevariant (Naam) + (IdIngerichteModulevariant)
 - Lokaal (naam) + (id)
 - PlaatsingResultaat (PuntenTotaal) + (IdTweedeZit)
 - Plaatsing (IdCursist) + (IdPlaatsingResultaat)
 - Cursist (CursistNummer)

Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt

Systeem ontwerp

woensdag 3 juni 2015
15:33

Software architectuur

We maken gebruik van een client-server architectuur waarbij de databank wordt bewaard in een SQL Server Database versie 2014, die enkel toegang verleent bij het indienen van de juiste servername en paswoord.

De applicatie bestaat uit verschillende lagen: de businesslaag, de data access laag en de user interface.

Alle lagen werken samen en geven informatie door, met gebruik van stored procedures word de juiste data uit de databank gehaald.

Data storage

Gegevens staan bewaard op een SQL Server in de database Administratix.

Nieuwe tabellen worden zelf aangemaakt en toegevoegd aan de database, de links met bestaande tabellen word gelegd door gebruik te maken van primary en foreign keys.

Data acces laag

Voor toegang tot de data worden stored procedures ingevoerd in de database en vervolgens opgeroepen in de dal.

Business laag

Deze laag word opgebouwd uit klassen die gegevens ophalen en doorsturen naar de data acces laag en user interface.

User interface

In de model worden pagina's één voor één opgeroepen om deze dan te tonen in view pages die bestaan uit webpagina's gemaakt met razor.

Razor webpagina's bestaat uit een technologie die gebruik maakt van de programmeertaal C# in combinatie met html code.

Beveiliging

De toegang wordt beperkt voor gebruikers die beschikken over een geldig cursistnummer.

Er is dus geen toegang voor bezoekers zonder geldig cursistnummer.

Auteurs:

Sonja Van Rompaey

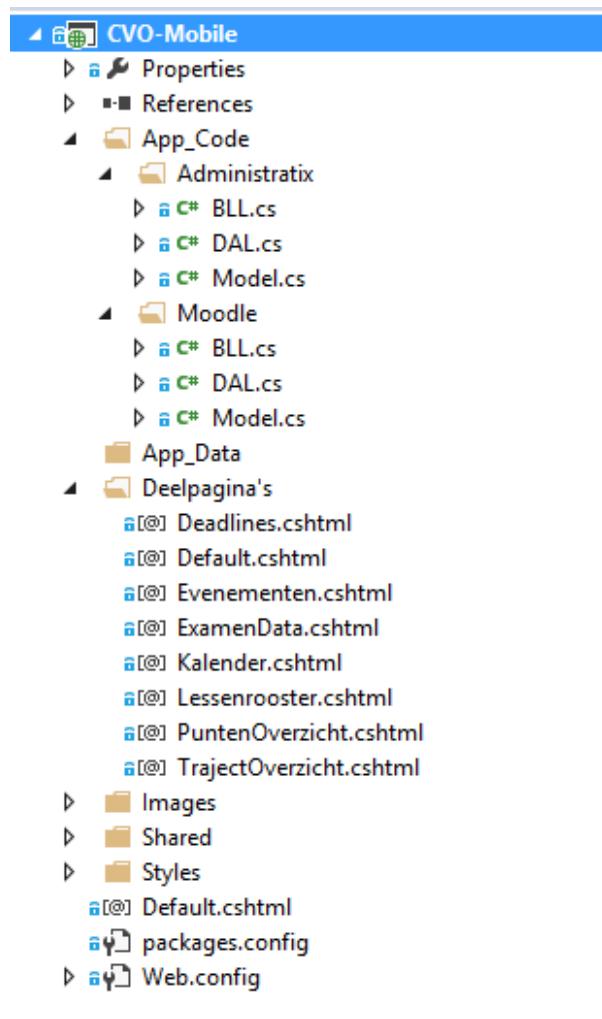
Mappenstructuur

woensdag 3 juni 2015

22:45

Structuur

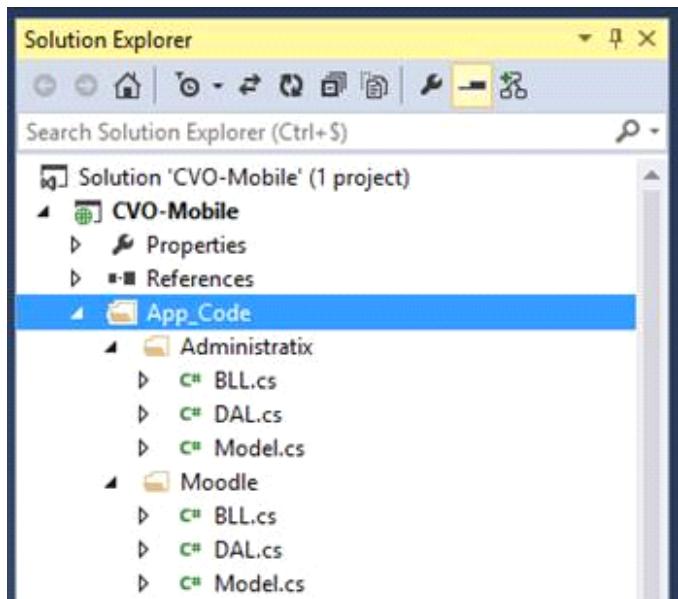
Om de indeling van het programma goed te kunnen begrijpen hebben we gebruik gemaakt van onderstaande mappenstructuur:



App_Code

Het project bevat een voorgeprogrammeerde ASP.NET folder App_Code, hierin staan de data access laag/ business laag en model voor zowel de Administratix database als Moodle.

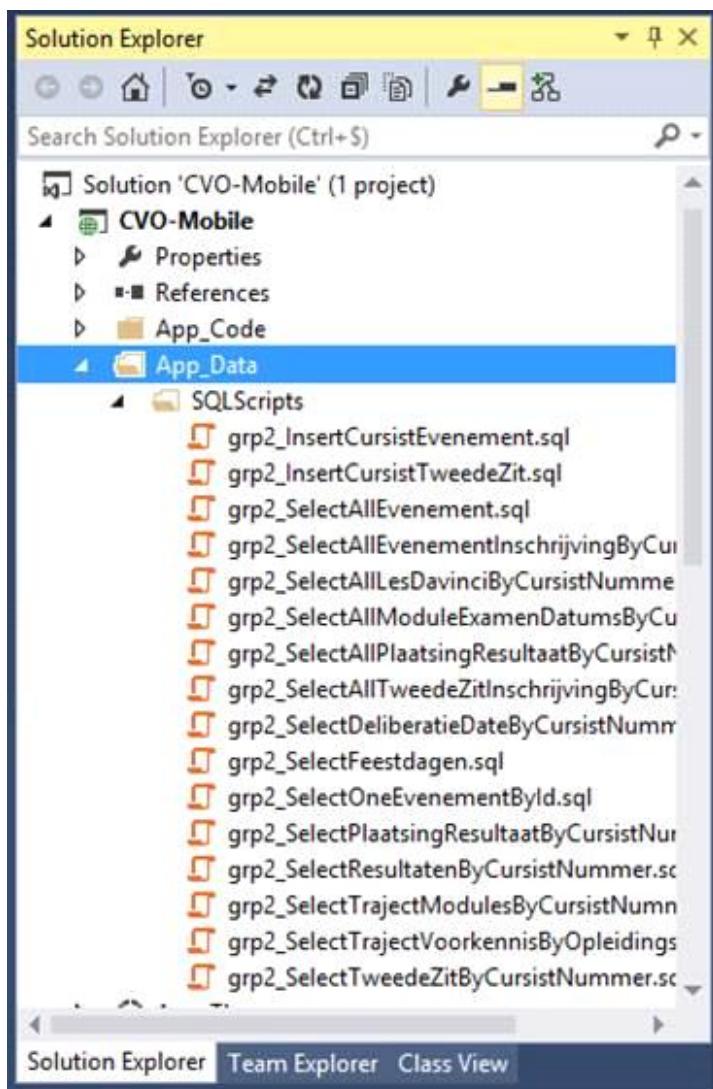
(Voor het project is een eigen Moodle aangemaakt om enkele cursussen en deadlines te kunnen opgeven.)



App_Data

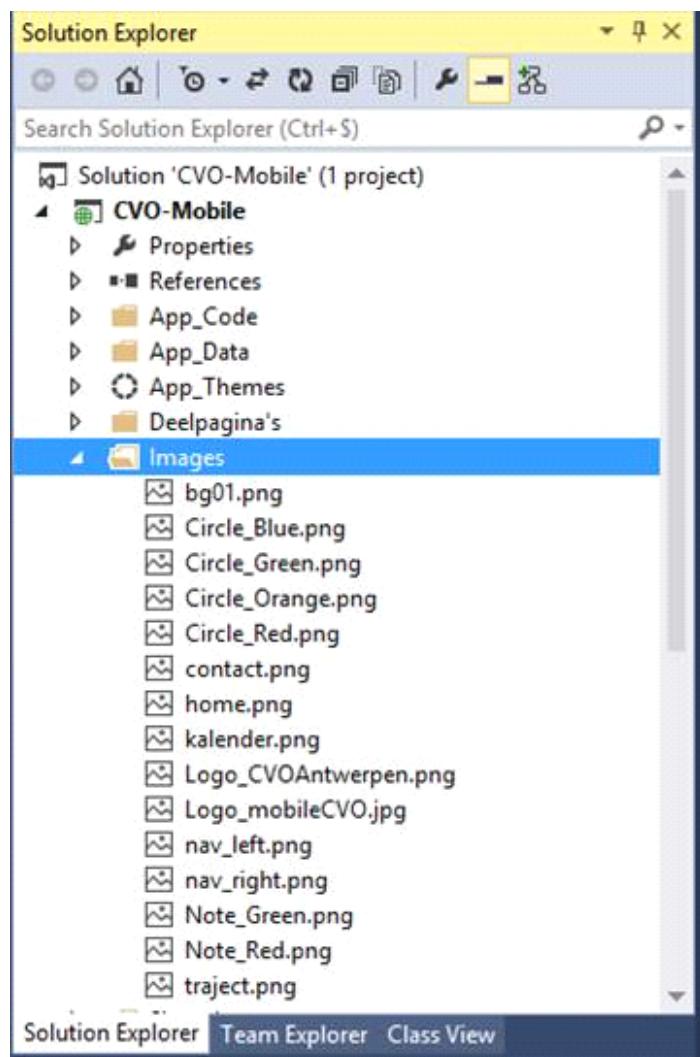
In de App_Data folder worden al de SQL Scripts bewaard in de genaamde folder.

Deze scripts zijn één voor één geschreven en uitgevoerd als stored procedures in de Administratix database.



Images folder

Er is in het project een folder Images gemaakt waarin alle gebruikte logo's staan. Logo's zijn door mij gemaakt of gehaald van IconFinder, zie bronvermelding.

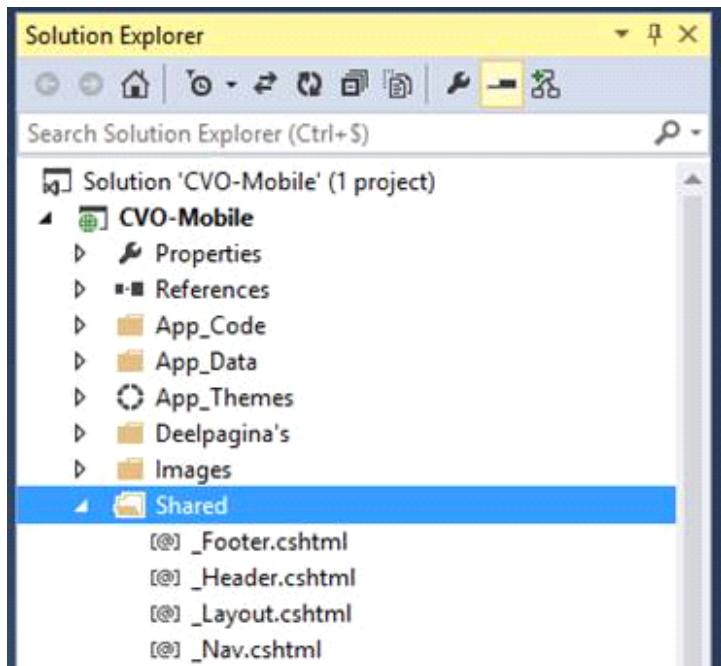


Razor Layout

De Shared folder bevat de layout die gedeeld word door elke razor webpage.

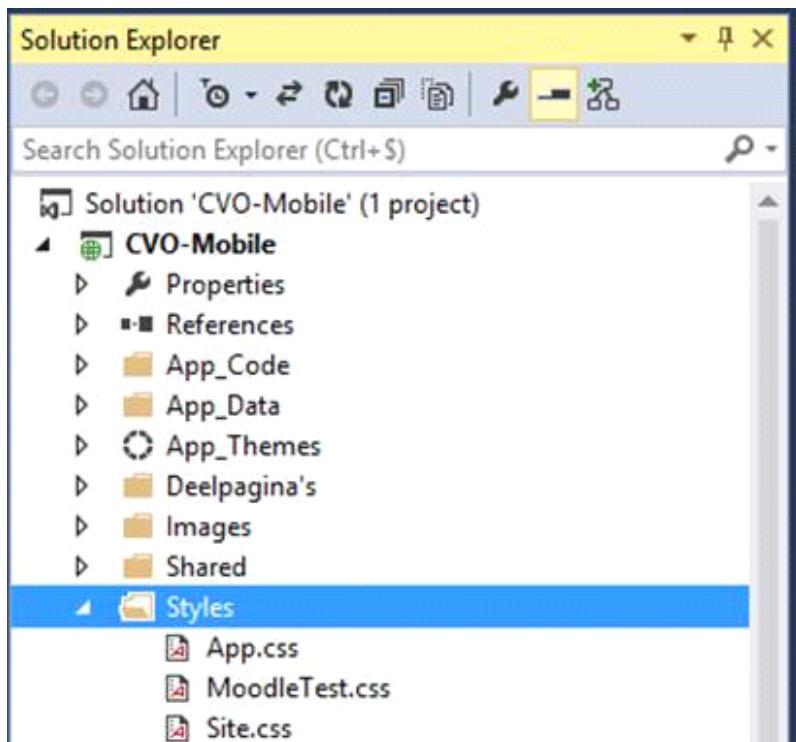
Er is dus voor elke pagina hetzelfde thema.

Ik heb deze gemaakt aan de hand van een tutorial om op een consequente manier ASP.NET Razor webpages te maken.
(zie bronvermelding voor weblink)



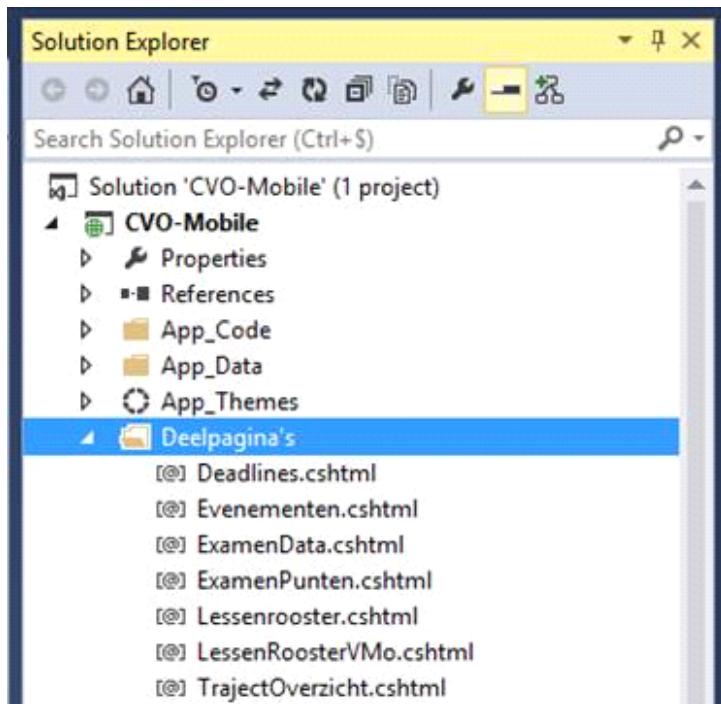
Cascading Style Sheets

Ik heb een Styles folder toegevoegd aan het project die alle css bevat.
Elk een afzonderlijke css file voor de applicatie, website en één voor onze Moodle versie.
(Moodle is authorised door Nikos)



Webpages

Deelpagina's folder bevat elke webpage afzonderlijk.
Elke razor page is afzonderlijk te openen in een webbrowser.



Auteurs:

Sonja Van Rompaey
Mohamed Amajoutt

SQL DDL

woensdag 3 juni 2015

23:06

Probleem

Er zijn verschillende opinies over het al of niet toevoegen van business rules in SQL zelf. De ene zegt dat alle business regels in de business laag van de applicatie moeten zitten. De andere beweert dat je zoveel mogelijk door de SQL server zelf moet laten doen.

Als je zoveel mogelijk door de SQL server laat doen, moet je app minder call's naar de SQL server maken en komt dat de performantie te goed. Vooral in mobiele applicaties. Dat is dan ook de reden waarom we ervoor gekozen heb zoveel mogelijk constraints in de SQL toe te voegen.

Als een kolom niet leeg mag zijn gaan we die constraint in de DDL zelf toevoegen. Als de app een rij wil wegschrijven naar de server en vergeet de waarde van een verplichte kolom mee te geven, gaat de SQL server een fout retourneren.

We hebben een manier nodig om de objecten in de database aan te maken. We doen dat niet met een visuele designer maar door uitvoerbare scripts te schrijven.

Met de Data Definition Language (DDL) definiëren we de objecten in de database waarin we gegevens zullen opslaan.

Met behulp van SQL/DDL kunnen we de objecten in een relationele database, zoals TABLE, VIEW, INDEX, SYNONYM, FOREIGN KEY, TRIGGER enz., te maken, wijzigen of verwijderen.

De DDL-commando's die we hier gebruiken zijn o.a. CREATE, DROP, ALTER, etc... Deze commando's bepalen de structuur van een database, zoals de kolommen, de tabellen, indexen, relaties en andere.

- Create - om een nieuwe database, tabel, index of opgeslagen query te maken.
- Drop - om een database, tabel, index of view te verwijderen.
- Alter - om een bestaand databaseobject te wijzigen.

Design

Tabel grp2_EvenementInschrijving

Reservatiedatum	date	null	
IdCursist	int	not null	Foreign key gekoppeld aan Id van tabel Cursist
IdEvenement	int	not null	Foreign key gekoppeld aan Id van tabel Evenement
Opmerkingen	nvarchar(255)	null	

Tabel grp2_TweedeZitInschrijving

Reservati edatum	date	null	
IdCursist	int	not null	Foreign key gekoppeld aan Id van tabel Cursist
IdTweede Zit	int	not null	Foreign key gekoppeld aan Id van tabel grp2_TweedeZit

Tabel grp2_TweedeZit

Datum	date	not null	
StartUur	datetime2(7))	not null	

EindUur	datetime2(7)	not null	
IdLokaal	int	not null	Foreign key gekoppeld aan Id van tabel Lokaal
Opmerkingen	nvarchar(255)	null	

Oplossing

--Created by Mohamed Amajoutt

--Tabel grp2_EvenementInschrijving toevoegen in database om bij te houden welke cursisten zijn ingeschreven in een evenement

```
create table grp2_EvenementInschrijving(
    Id int IDENTITY(1,1) PRIMARY KEY not null,
    Reservatiedatum date,
    IdCursist int not null,
    IdEvenement int not null,
    Opmerkingen nvarchar(255),
    constraint fk_EvenementInschrijving_IdCursist foreign key (IdCursist)
    references Cursist(Id),
    constraint fk_EvenementInschrijving_IdEvenement foreign key (IdEvenement)
    references Evenement(Id)
)
```

--Created by Mohamed Amajoutt on 21/05/2015

--Tabel grp2_TweedeZitInschrijving toevoegen in database om bij te houden welke cursisten zijn ingeschreven in een tweede zit.

```
create table grp2_TweedeZitInschrijving(
    Id int IDENTITY(1,1) PRIMARY KEY not null,
    Reservatiedatum date,
    IdCursist int not null,
    IdTweedeZit int not null,
    constraint fk_TweedeZitInschrijving_IdCursist foreign key (IdCursist)
    references Cursist(Id),
    constraint fk_TweedeZitInschrijving_IdTweedeZit foreign key (IdTweedeZit)
    references grp2_TweedeZit(Id)
)
```

--Created by Mohamed Amajoutt

--Tabel grp2_TweedeZit toevoegen in database om de 2de zit momenten bij te houden

```
create table grp2_TweedeZit(
    Id int IDENTITY(1,1) PRIMARY KEY not null,
```

```
Datum date not null,  
StartUur datetime2(7) not null,  
EindUur datetime2(7) not null,  
IdLokaal int not null,  
Opmerkingen nvarchar(255)  
constraint fk_TweedeZit_IdLokaal foreign key (IdLokaal)  
references Lokaal(Id)  
)
```

Auteurs:

Mohamed Amajoutt

SQL DML

woensdag 3 juni 2015
23:07

Probleem

Er zijn verschillende opinies over het al of niet toevoegen van business rules in SQL zelf. De ene zegt dat alle business regels in de business laag van de applicatie moeten zitten. De andere beweert dat je zoveel mogelijk door de SQL server zelf moet laten doen.

Als je zoveel mogelijk door de SQL server laat doen, moet je app minder call's naar de SQL server maken en komt dat de performantie te goed. Vooral in mobiele applicaties. Dat is dan ook de reden waarom we ervoor gekozen heb zoveel mogelijk constraints in de SQL toe te voegen.

Design

We geven hier een voorbeeld voor de tabellen Evenement en EvenementInschrijving. Voor het beheren van de database, maken we een aantal Stored Procedures die we later in de Data Acces Laag zullen gebruiken. De benaming van elke Stored Procedure begint met onze groep "grp2_" gevolgd door de CRUD actie:

- Insert
- SelectAll
- SelectAllByCursistNummer

Voor MobileCVO hebben we geen Update, Delete,SelectOne nodig omdat we telkens enkel een lijst gegevens gaan opvragen of gegevens gaan inserten.

De notatie van de benaming van de Stored Procedures zijn volgens de pascalnotatie.

Oplossing

- Insert

Let op de manier waarop we de Id van de laatst toegevoegde rij retourneren. Afhankelijk van de Out parameter kunnen we in de DAL de gepast boodschap sturen naar de gebruiker.

--Mohamed Amajoutt
--10/05/2015
--Stored procedure insert voor de tabel grp2_EvenementInschrijving
--Cursist inschrijven voor evenement

```
use Administratix_cursist
go

if exists (select 1 from sysobjects where name = 'grp2_InsertEvenementInschrijving' AND type =
'P') -- P = procedure
begin
    drop proc grp2_InsertEvenementInschrijving
end
go

-- aanmaken van de procedure
create procedure grp2_InsertEvenementInschrijving
(
```

```

    @CursistNummer int,
    @IdEvenement int,
    @Opmerkingen nvarchar(255),
    @Reservatiedatum datetime,
    -- out omdat de waarde naar de calling
    -- programma geretourneerd moet worden
    @Id int out
)
as
begin
declare @CurrentId int
declare @CursistId int
select @CursistId = Id from Cursist where CursistNummer = @CursistNummer
select @CurrentId = Id from grp2_EvenementInschrijving where IdCursist = @CursistId and
IdEvenement = @IdEvenement
if @CurrentId is not null
begin
    --cursist is al ingeschreven voor evenement
    set @Id = -100
    return
end

--Cursist is nog niet ingeschreven voor evenement
insert into grp2_EvenementInschrijving
(
    Reservatiedatum,
    IdCursist,
    IdEvenement,
    Opmerkingen
)
values
(
    @Reservatiedatum,
    @CursistId,
    @IdEvenement,
    @Opmerkingen
)

-- Retourneert de Id van de nieuw toegevoegde rij.
set @Id = SCOPE_IDENTITY()
return
end

```

- [SelectAll](#)

--Mohamed Amajooutt
--10/05/2015
--Lijst geven van alle evenementen vanaf huidige datum gesorteerd op datum

```

use Administratix_cursist
go
if exists (select 1 from sysobjects where name = 'grp2_SelectAllEvenement' AND type = 'P') -- P =
procedure

begin
    drop proc grp2_SelectAllEvenement

```

```

end
go
create procedure grp2_SelectAllEvenement
as
begin
select
    Id
    ,Naam
    ,Evenement.Datum as Datum
    ,Locatie
    ,Evenement.StartUur as StartUur
    ,Evenement.EindUur as EindUur
    from Evenement where Evenement.Datum >= GETDATE()
    order by StartUur
end

```

- SelectAllByCursistNummer

Afhankelijk van de CursistNummer parameter kunnen we in de DAL een lijst opmaken.

```

-- Mohamed Amajoutt
-- 26/05/2015
--SP toont persoonlijke lijst voor welke evenementen u bent ingeschreven adhv het cursistnummer

use Administratix_cursist
go
if exists (select 1 from sysobjects where name = 'grp2
_SelectAllEvenementInschrijvingByCursistNummer' AND type = 'P') -- P = procedure
begin
    drop proc grp2_SelectAllEvenementInschrijvingByCursistNummer
end
go
create procedure grp2_SelectAllEvenementInschrijvingByCursistNummer
(
    @CursistNummer int
)
as
begin
select
    Reservatiedatum
    , Evenement.Naam as Naam
    , Evenement.Datum as Datum
    , Evenement.Locatie as Locatie
    , Evenement.StartUur as StartUur
    , Evenement.EindUur as EindUur
    , Opmerkingen
from grp2_EvenementInschrijving
inner join Evenement on grp2_EvenementInschrijving.IdEvenement = Evenement.Id
inner join Cursist on grp2_EvenementInschrijving.IdCursist = Cursist.Id
where Cursist.CursistNummer = @CursistNummer and Evenement.Datum >= GETDATE()
order by Evenement.Datum, Evenement.StartUur
End

```

Auteurs:

Mohamed Amajoult

Sonja Van Rompaey

Werkwijze Insert

woensdag 3 juni 2015
17:49

Auteur:
Jo Ronsse
Mohamed Amajoutt

Probleem

Om te kunnen in te schrijven voor evenementen moet er Data worden toegevoegd aan de server.
Hiervoor moest een stored procedure(SP afgekort) voor gemaakt worden die de waardes kon toevoegen .

Design

Om te beginnen duiden we aan welke database we gaan gebruiken.

```
use Administratix_cursist
go
```

Dan kan het zijn dat er een oude procedures verwijderd worden omdat ze dezelfde naam heeft als degene die we gaan aanmaken.
Deze methode kijkt of er andere SP zijn met de naam 'grp2_InsertCursistEvenement'.

Als de procedure bestaat word die verwijderd, hiermee worden duplicates en problemen voorkomen.

```
if exists (select 1 from sysobjects where name = 'grp2_InsertCursistEvenement' AND type =
'P') -- P = procedure

begin
    drop proc grp2_InsertCursistEvenement
end
go
```

Nadat we zeker zijn dat er geen Procedure is met dezelfde naam kunnen we de SP aanmaken, dit doen we met het 'create' commando, hiet geven we ook mee welke data we verwachten van het programma dat deze SP gaat gebruiken.

```
create procedure grp2_InsertCursistEvenement
(
    @CursistNummer int,
    @IdEvenement int,
    @Opmerkingen nvarchar(255),
    @Reservatiedatum datetime,
    @Id int out
)
```

Na we de procedure hebben aangemaakt en hebben aangeduid welke data we verwachten zetten we 'as begin' , dit duid aan dat de code die hierna komt de code is die gaat worden aangeroepen.

Daarna declareren we waarden die we uit de database zelf aanroepen, voor later gebruik of in dit geval om te zien of de gebruiker al is ingeschreven of niet.

```
as
begin
declare @CurrentId int
declare @CursistId int
select @CursistId = Id from Cursist where CursistNummer = @CursistNummer
select @CurrentId = Id from grp2_EvenementInschrijving where IdCursist = @CursistId and
IdEvenement = @IdEvenement
```

Nu we de waardes hebben die we nodig hebben, kunnen we na zien of de gebruiker al is ingeschreven, als dat het geval is stopt het programma en geeft het de return waarde van -100 mee, zodat het programma dat de SP heeft aangeroepen weet dat de gebruiker al is ingeschreven.

Dit gebeurt door 'return' wat de SP forceert te stoppen en de waarde ID terug te geven.

```
if @CurrentId is not null
begin
    set @Id = -100
return
end
```

Als de gebruiker nog niet is ingeschreven (de waarde '@CurrentId' bestaat niet en is dus NULL) dan worden de gegevens die zijn meegegeven met de SP in de database gestoken.

Hierbij is de gebruiker ingeschreven in het evenement van zijn/haar keuze en kan de administratie makkelijk zien wie is ingeschreven en eventueel heeft besteld.

```
insert into grp2_EvenementInschrijving
(
    Reservatiedatum,
    IdCursist,
    IdEvenement,
    Opmerkingen
)
values
(
    @Reservatiedatum,
    @CursistId,
    @IdEvenement,
    @Opmerkingen
)
```

En als laatste sturen we de ID van de nieuwe lijn in de database mee, hiermee weet het programma dat de SP heeft aangeroepen dat de procedure succesvol is afgerond.

```
set @Id = SCOPE_IDENTITY()
return
end
```

Oplossing

--Mohamed Amajoult

--10/05/2015

```
use Administratix_cursist
go

if exists (select 1 from sysobjects where name = 'grp2_InsertCursistEvenement' AND type = 'P') --
P = procedure

begin
    drop proc grp2_InsertCursistEvenement
end
go

create procedure grp2_InsertCursistEvenement
(
    @CursistNummer int,
    @IdEvenement int,
    @Opmerkingen nvarchar(255),
    @Reservatiedatum datetime,
    @Id int out
)
as
begin
declare @CurrentId int
declare @CursistId int
select @CursistId = Id from Cursist where CursistNummer = @CursistNummer
select @CurrentId = Id from grp2_EvenementInschrijving where IdCursist = @CursistId and
IdEvenement = @IdEvenement

if @CurrentId is not null
begin
    set @Id = -100
    return
end

insert into grp2_EvenementInschrijving
(
    Reservatiedatum,
    IdCursist,
    IdEvenement,
    Opmerkingen
)
```

```
)  
values  
(  
    @Reservatiedatum,  
    @CursistId,  
    @IdEvenement,  
    @Opmerkingen  
)  
  
set @Id = SCOPE_IDENTITY()  
return  
end
```

Werkwijze Select

woensdag 3 juni 2015

17:49

Auteur:

Jo Ronsse

Mohamed Amajoutt

Probleem

Voor het project moeten er veel waarden worden afgehaald van de database , voor elke pagina wel een andere waarde, daarvoor werden Stored procedures(SP afgekort) geschreven voor elke tabel waar data werd uitgehaald, hier is een voorbeeld.

Design

Om te beginnen duiden we aan welke database we gaan gebruiken.

```
use Administratix_cursist  
go
```

Dan kan het zijn dat er een oude procedures verwijderd worden omdat ze dezelfde naam heeft als degene die we gaan aanmaken.

Deze methode kijkt of er andere SP zijn met de naam '`grp2_SelectAllLesDavinciByCursistNummer`' .

Als de procedure bestaat word die verwijderd, hiermee worden duplicates en problemen voorkomen.

```
if exists (select 1 from sysobjects where name = 'grp2_SelectAllLesDavinciByCursistNummer' AND  
type = 'P') -- P = procedure  
  
begin  
    drop proc grp2_SelectAllLesDavinciByCursistNummer  
end  
go
```

Nadat we zeker zijn dat er geen Procedure is met dezelfde naam kunnen we de SP aanmaken, dit doen we met het 'create' commando, hier geven we ook mee welke data we verwachten van het programma dat deze SP gaat gebruiken.

```
create procedure grp2_SelectAllLesDavinciByCursistNummer  
(  
    @CursistNummer int  
)
```

Nu we de waardes hebben die we willen gebruiken in de SP kunnen we de waarden selecteren die we willen terug geven , we beginnen met 'begin' om aan te duiden dat de code die word opgeroepen hier start.

Daarna worden waardes geselecteerd van de database, deze kunnen van andere tabellen worden gehaald (bv Personeel.Naam).

Er word ook weer gegeven van welke tabel je de waardes wil uithalen.

```
as  
begin  
select  
    IngerichteModulevariant.CursusNummer as Cursusnummer  
    ,LesDavinci.Lesdatum as Datum  
    ,Personeel.Naam + ' ' + Personeel.Voornaam as Docent  
    ,Lesplaats.Naam as Campus  
    ,LesDavinci.LocatieOmschrijving as Lokaal  
    ,IngerichteModulevariant.Naam as Module  
    ,LesDavinci.AanvangsDatum as Van  
    ,LesDavinci.Einddatum as Tot  
from LesDavinci
```

Nadat de waardes geselecteerd zijn worden de tabellen samengevoegd met 'inner join', dit zorgt ervoor dat je waardes uit andere tabellen kunt halen en dicteert hoe de 2 tabellen zijn gelinkt aan het programma om er mee te kunnen werken

```
inner join IngerichteModulevariant on LesDavinci.IdIngerichteModulevariant =  
    IngerichteModulevariant.Id  
inner join Lesplaats on LesDavinci.IdLesplaats = LesPlaats.Id  
inner join Personeel on LesDavinci.IdPersoneel = Personeel.Id  
inner join Lokaal on LesDavinci.IdLokaal = Lokaal.Id  
inner join Plaatsing on Plaatsing.IdIngerichteModulevariant = IngerichteModulevariant.id  
inner join Cursist on Plaatsing.IdCursist = Cursist.Id
```

Dan worden de waardes gefilterd op specifieke gegevens, in dit geval volgens de cursistnummer, zodat de gebruiker enkel waardes terug krijgt die voor hem/haar bestemd zijn.

```
where Cursist.CursistNummer = @CursistNummer and LesDavinci.Aanvangsdatum >= GETDATE()
```

Als laatste sorteren we de waardens voor we ze verzenden en sluiten het programma af.

```
order by LesDavinci.Lesdatum, LesDavinci.Aanvangsdatum
end
```

Oplossing

```
-- Mohamed Amajoutt
-- 27/04/2015
use Administratix_cursist
go

if exists (select 1 from sysobjects where name = 'grp2_SelectAllLesDavinciByCursistNummer' AND type = 'P') -- P = procedure

begin
    drop proc grp2_SelectAllLesDavinciByCursistNummer
end
go

create procedure grp2_SelectAllLesDavinciByCursistNummer
(
    @CursistNummer int
)

as
begin
select
    IngerichteModulevariant.CursusNummer as Cursusnummer
    ,LesDavinci.Lesdatum as Datum
    ,Personenel.Naam + ' ' + Personenel.Voornaam as Docent
    ,Lesplaats.Naam as Campus
    ,LesDavinci.LocatieOmschrijving as Lokaal
    ,IngerichteModulevariant.Naam as Module
    ,LesDavinci.AanvangsDatum as Van
    ,LesDavinci.Einddatum as Tot
from LesDavinci
inner join IngerichteModulevariant on LesDavinci.IdIngerichteModulevariant =
IngerichteModulevariant.Id
inner join Lesplaats on LesDavinci.IdLesplaats = LesPlaats.Id
inner join Personenel on LesDavinci.IdPersonenel = Personenel.Id
inner join Lokaal on LesDavinci.IdLokaal = Lokaal.Id
inner join Plaatsing on Plaatsing.IdIngerichteModulevariant = IngerichteModulevariant.id
inner join Cursist on Plaatsing.IdCursist = Cursist.Id
where Cursist.CursistNummer = @CursistNummer and LesDavinci.Aanvangsdatum >= GETDATE()
order by LesDavinci.Lesdatum, LesDavinci.Aanvangsdatum
end
```

Leerkracht dummies

zaterdag 6 juni 2015

13:21

Probleem

Voor privacy redenen was, in de database die we van de school kregen, de tabel Personeel leeg. Omdat we deze wel nodig hebben als we de lessenrooster opvragen, moeten we hier zelf data toevoegen.

Design

Dummies toevoegen

Door connecties te leggen tussen de Personeel Ids is de andere tabellen weten we dat 453 de hoogste Id is die voor ons van belang is.

We willen dus 500 dummy rijen toevoegen in de tabel.

Uiteraard beginnen we door eerst aan te kondigen welke database we willen gebruiken.

```
use Administratix_cursist  
go
```

Vervolgens hebben we een counter nodig die bijhoudt hoeveel dummies we al hebben toegevoegd, beginnende van 1.

```
declare @id int  
select @id = 1
```

Nu kunnen we, zoals in c# een while lus gebruiken. Deze gebruikt wel een iets andere syntax.

```
while @id >=1 and @id <= 500  
begin  
end
```

In deze lus inserten we dan telkens een nieuwe rij op dezelfde manier als we altijd doen.

Om de verschillende dummies uit elkaar te halen voegen we de Id toe aan de naam.

Dit kunnen we doen door de Id te converten naar een varchar.

```
insert into Personeel (Naam, Voornaam, WordtGebruiktDoorSysteem )  
values('Test', 'Dummy' + convert(varchar(5), @id), '1')
```

Nu moeten we uiteraard de Id nog verhogen, anders maken we een endless loop.

```
select @id = @id + 1
```

Specifieke namen updaten

Nu kunnen we specifieke rijen updaten zodat, als we deze opvragen in de lessenrooster, de correcte naam van de docent te zien is.

We kunnen de Ids achterhalen omdat we via de IngerichteModuleVariant weten welke docent de module geeft.

Daarna kunnen we in LesDavinci de lessen van de module opvragen en de PersoneelId zien.

Eerst selecteren we weer de database.

```
use Administratix_cursist  
go
```

Hieraan updaten we de naam en voornaam voor de specifieke Ids in de Personeel tabel.

```
Update Personeel set Naam = 'Balbeart' ,Voornaam = 'Ivo' Where Id = '3'  
go  
Update Personeel set Naam = 'Inghelbrecht' ,Voornaam = 'Joseph' Where Id = '453'  
go  
Update Personeel set Naam = 'Verbesselt' ,Voornaam = 'Tom' Where Id = '194'  
go  
Update Personeel set Naam = 'Picavet' ,Voornaam = 'Nico' Where Id = '383'  
go  
Update Personeel set Naam = 'Van De Velde' ,Voornaam = 'Chris' Where Id = '68'  
go
```

Oplossing

Dummies toevoegen

```
use Administratix_cursist
go

declare @id int
select @id = 1
while @id >=1 and @id <= 500
begin
    insert into Personeel (Naam, Voornaam, WordtGebruiktDoorSysteem )
    values('Test', 'Dummy' + convert(varchar(5), @id), '1')
select @id = @id + 1
end
```

Specifieke namen updaten

```
use Administratix_cursist
go
Update Personeel set Naam = 'Balbeaert' ,Voornaam = 'Ivo' Where Id = '3'
go
Update Personeel set Naam = 'Inghelbrecht' ,Voornaam = 'Joseph' Where Id = '453'
go
Update Personeel set Naam = 'Verbesselt' ,Voornaam = 'Tom' Where Id = '194'
go
Update Personeel set Naam = 'Picavet' ,Voornaam = 'Nico' Where Id = '383'
go
Update Personeel set Naam = 'Van De Velde' ,Voornaam = 'Chris' Where Id = '68'
go
```

Auteurs:

Nikos Vanden Broek

Voorkennis Tabel

vrijdag 15 mei 2015

14:45

Probleem

In een traject hebben sommige modules andere modules als voorkennis.

De cursist moet eerst slagen voor de voorkennis voordat hij of zij aan de andere module kan beginnen.

Omdat deze informatie niet in de database zit, moeten we dit zelf toevoegen.

Design

We maken een tabel die een link maakt tussen de module waarvoor de voorkennis van toepassing is, en de module die als voorkennis dient.

We gebruiken ook de Opleidingsvariant zodat we data per traject kunnen opvragen.

Tabel kolommen

Kolom	Beschrijving	Key	Tabel
Id	Id van de rij.	Primary key	
OpleidingsvariantId	De Id van de opleidingsvariant	Foreign Key	Opleidingsvariant
ModulevariantId	Id van de Module waar de voorkennis van toepassing is	Foreign Key	Modulevariant
VoorkeennisModulevariantId	Id van de Module die als voorkennis dient	Foreign Key	Modulevariant

Tabel aanmaken

We selecteren eerst welke database we gebruiken.

```
use Administratix_cursist  
go
```

Hieraan kijken we of de tabel al bestaat. Indien dit het geval is verwijderen we de tabel uit de database.

Let op! Dit zal ook al de data die in de tabel aanwezig was verwijderen

```
if exists(select * from sys.tables where name='ModulevariantTrajectVoorkeennis')  
begin  
    drop table ModulevariantTrajectVoorkeennis  
end  
go
```

Nu maken we de tabel aan met de nodige tabellen.

We zorgen ervoor dat de Id een identity is en, omdat alle kolommen nodig zijn, maken we deze not null.

```
create table ModulevariantTrajectVoorkeennis(  
    Id int identity(1,1) not null  
    ,OpleidingsvariantId int not null  
    ,ModulevariantId int not null  
    ,VoorkeennisModulevariantId int not null
```

Eens de tabel aangemaakt is moeten we ook de constraints toevoegen.

Eerst zorgen we ervoor dat onze Id een primary key is.

```
constraint PK_ModulevariantTrajectVoorkeennis primary key(id)
```

Daarna voegen maken we al de foreign keys aan voor de kolommen die naar een andere tabel verwijzen.

Dit doen we door de kolomnaam mee te geven als key, en als reference verwijzen we naar de kolom in de andere tabel.

```
constraint FK_ModulevariantTrajectVoorkeennis_Opleidingsvariant foreign  
key(OpleidingsvariantId) references Opleidingsvariant(Id)  
,constraint FK_ModulevariantTrajectVoorkeennis_ModuleVariant foreign  
key(ModulevariantId) references Modulevariant(Id)  
,constraint FK_ModulevariantTrajectVoorkeennis_VoorkeennisModuleVariant foreign  
key(VoorkeennisModulevariantId) references Modulevariant(Id)
```

Oplossing

```
use Administratix_cursist
go

if exists(select * from sys.tables where name='ModulevariantTrajectVoorkennis')
begin
    drop table ModulevariantTrajectVoorkennis
end
go

create table ModulevariantTrajectVoorkennis(
    Id int identity(1,1) not null
    ,OpleidingsvariantId int not null
    ,ModulevariantId int not null
    ,VoorkennisModulevariantId int not null

-- Constraints in zelfde stijl als bestaande keys
constraint PK_ModulevariantTrajectVoorkennis primary key(id)
,constraint FK_ModulevariantTrajectVoorkennis_Opleidingsvariant foreign
key(OpleidingsvariantId) references Opleidingsvariant(Id)
,constraint FK_ModulevariantTrajectVoorkennis_ModuleVariant foreign
key(ModulevariantId) references Modulevariant(Id)
,constraint FK_ModulevariantTrajectVoorkennis_VoorkennisModuleVariant foreign
key(VoorkennisModulevariantId) references Modulevariant(Id)

)
Go
```

Auteurs:

Nikos Vanden Broek

Voorkennis toevoegen

vrijdag 15 mei 2015

15:39

Probleem

Nu we de tabel hebben aangemaakt moeten we al de nodige gegevens nog toevoegen.

Omdat we in het project enkel data voor Informatica gebruiken, voegen we enkel de voorkennis van Informatica toe.

Design

Weer selecteren we eerst de database

```
use Administratix_cursist
go
```

Nu kunnen we met een insert de nodige data toevoegen.

```
insert into ModulevariantTrajectVoorkennis
(OpleidingsvariantId
,ModulevariantId
,VoorkennisModulevariantId
)
```

Omdat we de Ids uit de tabel Modulevariant nodig hebben kunnen we in onze values van de insert een select gebruiken.

Dit doen we door met de select de Id op te vragen voor een specifieke modulenaam.

Omdat er in de tabel Opleidingsvariant meerdere rijen zijn met de naam Informatica, gebruiken we de specifieke Id (140).

```
(140
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
```

Oplossing

```
use Administratix_cursist
go

insert into ModulevariantTrajectVoorkennis
(OpleidingsvariantId
,ModulevariantId
,VoorkennisModulevariantId
)
values
-- Netwerkbeheer 1 TV
(140
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
-- Besturingssystemen TV
,(140
,(select Id from Modulevariant Where Naam = 'Besturingssystemen TV')
,(select Id from Modulevariant Where Naam = 'Basiskennis TV')
)
-- Programmeren 3 TV
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 3 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 3 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 2 TV')
)
-- Analyse TV
,(140
```

```

,(select Id from Modulevariant Where Naam = 'Analyse TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Analyse TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 2 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Analyse TV')
,(select Id from Modulevariant Where Naam = 'Databanken TV')
)
-- Programmeren 4 TV
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 4 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 4 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 2 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 4 TV')
,(select Id from Modulevariant Where Naam = 'Databanken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 4 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 3 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 4 TV')
,(select Id from Modulevariant Where Naam = 'Analyse TV')
)
-- Programmeren 5 TV
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 5 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 5 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 2 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 5 TV')
,(select Id from Modulevariant Where Naam = 'Databanken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 5 TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 3 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Programmeren 5 TV')
,(select Id from Modulevariant Where Naam = 'Analyse TV')
)
-- Projectwerk informatica (TV)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Programmeren 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Programmeren 2 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Databanken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Programmeren 3 TV')
)

```

```

)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Analyse TV')
)
-- Beheer van databanken TV
,(140
,(select Id from Modulevariant Where Naam = 'Beheer van databanken TV')
,(select Id from Modulevariant Where Naam = 'Databanken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Beheer van databanken TV')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Beheer van databanken TV')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
)
-- Netwerkbeheer 2 TV
,(140
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 2 TV')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 2 TV')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
)
-- Datacommunicatie TV, Zo bestaan er blijkbaar 2
,(140
,1709 --(select Id from Modulevariant Where Naam = 'Datacommunicatie TV')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
,(140
,1709 --(select Id from Modulevariant Where Naam = 'Datacommunicatie TV')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
)
-- Internettechnologie, systeem- en netwerkbeheer TV
,(140
,(select Id from Modulevariant Where Naam = 'Internettechnologie, systeem- en
netwerkbeheer TV')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Internettechnologie, systeem- en
netwerkbeheer TV')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Internettechnologie, systeem- en
netwerkbeheer TV')
,(select Id from Modulevariant Where Naam = 'Programmeren 2 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Internettechnologie, systeem- en
netwerkbeheer TV')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 2 TV')
)
-- Projectwerk informatica (TV)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Datacommunicatie en netwerken TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 1 TV')
)
,(140
,(select Id from Modulevariant Where Naam = 'Projectwerk informatica (TV)')
,(select Id from Modulevariant Where Naam = 'Netwerkbeheer 2 TV')
)

```

)

Auteurs:

Nikos Vanden Broek

Logingegevens MSSQL Server

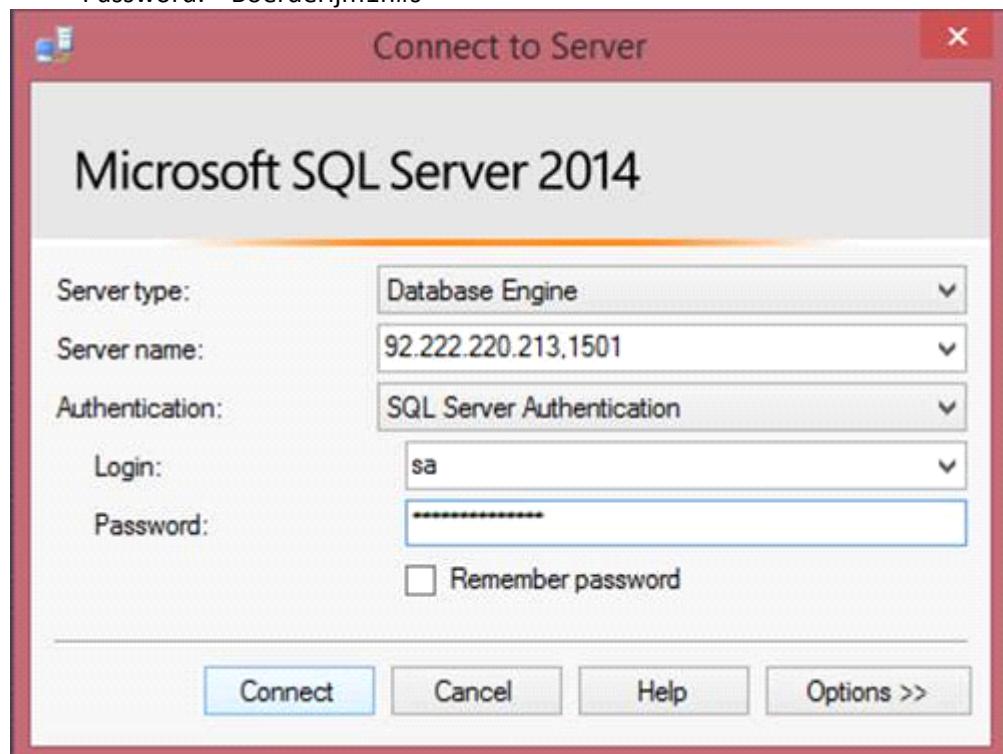
donderdag 4 juni 2015

11:27

Handleiding

Connectie met de database

Open SQL Server versie 2014 als administrator,
geef de juiste servername en paswoord om verbinding te maken.
Servername: 92.222.220.213.1501
Login: sa
Password: Boerderijm1n#s



Auteurs:

Sonja Van Rompaey

Web services Activeren

dinsdag 31 maart 2015

17:47

Toestaan

Om webservices te laten werken op Moodle moeten webservices eerst toegelaten worden door een administrator

Dit kan gedaan worden door als administrator in te loggen en onder *Instellingen -> Sitebeheer -> Geadvanceerd* de optie 'Webservices inschakelen' aan te vinken.

https://docs.moodle.org/22/en/Using_web_services

Inschakelen

Nadat de Webservices ingeschakeld zijn moeten ook de basis webservices nog ingeschakeld worden.

Dit kan gedaan worden door als admin naar *Instellingen -> Sitebeheer -> Plugins -> Webservices -> Externe Services* te navigeren.

Hier kun je voor 'Moodle mobile web service' op bewerken klikken en de optie 'Ingeschakeld' aanvinken

Gebruiker aanmaken

Nu moeten we een gebruiker aanmaken die toegang krijgt om de webservices te gebruiken

Log in als administrator en navigeer naar *Instellingen -> Sitebeheer -> Gebruikers -> Gebruikers -> Toevoegen*.

Vul de verplichte velden in, laat "Kies een methode van authenticatie" op "Manuale accounts" staan en maak de gebruiker aan.

Rollen aanmaken en toewijzen

Nu we een gebruiker hebben aangemaakt moeten we een rol aanmaken waar we de nodige rechten aan kunnen toewijzen.

Ga naar *Instellingen -> Sitebeheer -> Gebruikers -> Rechten -> Definieer rollen* en klik onderaan op Nieuwe rol toevoegen.

Klik vervolgens op Doorgaan, geef de rol een korte naam en volledige naam, en klik op Maak deze rol.

Ge vervolgens links naar Globale rollen toewijzen.

Hier zien we onze nieuwe rol en kunnen we onze gebruiker toewijzen aan deze rol.

Doe dit door op de rol te klikken en in het volgende scherm onze gebruiker in de rechter kolom te selecteren, en op de knop Voeg toe te klikken.

Externe service aanmaken

Nu we een gebruiker hebben die de web services kan gebruiken, moeten we een externe service aanmaken die onze gebruiken kan aanspreken.

Een externe service is een groep van web services waar de gebruiken toegang tot krijgt.

Als admin navigeer je naar *Instellingen -> Sitebeheer -> Webservices -> Externe services* en onderaan klik je op 'Voeg Toe'.

Geef de service een naam en een korte naam. De korte naam hebben we nodig om de service te kunnen gebruiken in onze code.

Vink 'Ingeschakeld' en 'Enkel geautoriseerde gebruikers' aan en voeg de service toe.

Je externe service verschijnt nu in de lijst van Aangepaste services.

Functies toevoegen

Nu moeten we de functies die we nodig hebben, toevoegen aan onze externe service.

In de lijst van de externe service klik je op Functies.

Hier klik je op Functies toevoegen en selecteer je de nodige functies uit de lijst.

Het is mogelijk om, met ctrl of shift , meerdere functies tegelijk te selecteren en toe te voegen.

Klik op Functies toevoegen en de gekozen functies verschijnen in de lijst.

Geautoriseerde gebruiker toevoegen

Nu moeten we onze gebruiker toevoegen aan de lijst van geautoriseerde gebruikers zodat deze de rechten heeft om de externe service te gebruiken.

Ga terug naar de lijst van externe services, en klik op Geautoriseerde gebruikers van onze service. Zoals eerder krijgen we een lijst waar we onze gebruiker in kunnen selecteren, en voegen we hem toe.

Onderaan de pagina komt mogelijk een lijst te zien van alle mogelijkheden die onze gebruiker mist om de gekozen functies uit te voeren.

Mogelijkheden toevoegen aan rollen

Navigeer terug naar de last van rollen (*Instellingen -> Sitebeheer -> Gebruikers -> Rechten -> Definieer rollen*) en klik op het tandwiel icoon rechts van onze eerder aangemaakte rol.

Als we iets naar beneden scrollen zien we een lange lijst van alle mogelijkheden die onze rol kan hebben.

Zoek voor de mogelijkheden de onze rol ontbrak, vink deze aan en klik op de knop Bewaar de wijzigingen.

Leraar zonder bewerken

In onze applicatie willen we de cursisten hun scores op taken kunnen bekijken.

Om onbekende redenen heeft, ongeacht welke mogelijkheden, onze gebruiker geen rechten om de punten van cursisten te bekijken tenzij de gebruiker minstens de rol Leraar zonder bewerken in elke cursus.

Ga naar *Instellingen -> Sitebeheer -> Cursussen -> Beheer cursussen en categoriën* en klik op het tandwiel naast de naam van de cursus.

In de lijst links verschijnt Cursusbeheer. Ga hier naar *Gebruikers -> Aangemelde gebruikers*.

Klik rechtsboven op de knop Gebruikers aanmelden, zoek onze gebruiker, klik op de knop Meld aan en sluit het venster door op de knop beëindig het aanmelden van gebruikers.

Onze gebruiker staat nu in de lijst van aangemelde gebruikers met de rol Leerling. Verwijder deze rol door op de X te klikken

Klik daarna op het icoontje van de persoon met een + teken en klik op Leraar zonder bewerken.

De dit voor al de cursussen die van toepassing zijn en onze gebruiker is klaar om de webservices te gebruiken.

Auteurs:

Nikos Vanden Broek

Json.NET

vrijdag 3 april 2015
15:56

Inleiding

Als we web services gebruiken voor Moodle, zullen we altijd een Json string terug krijgen als response.

Deze string bevat al de informatie die we nodig hebben, maar is als gewone string moeilijk te gebruiken.

Hiervoor bestaat een NuGet package genaamd **Json.NET** die ons kan helpen om de teruggekregen data te structureren.

Link: <http://www.newtonsoft.com/json>

Om deze package toe te voegen aan het project openen we Tools -> NuGet Package Manager, en typen we volgende regel:

```
Install-Package Newtonsoft.Json
```

Nadat te package in geïnstalleerd kunnen we de code gebruiken als we volgende namespace toevoegen:

```
using Newtonsoft.Json.Linq;
```

Voorbeeld

Als we de webservice core_calendar_get_calendar_events gebruiken krijgen we volgende string terug als response:

"events": [{"id": 1, "name": "Project Requirements deel 1 - Cursisten", "description": "\n

CursistenPortaal: welke functionaliteiten vanuit het standpunt van de cursist, en hoe die best opvraagbaar en gepresenteerd kunnen worden.\n
Structuur je idee\n\nvan een tekst, in te leveren als Word-document (minimaal 1 pagina) ten laatste 25-2, individueel op te laden. Hou ook je logboek bij (cfr. logboek template)!<\\p>\", \"format\": 1, \"courseid\": 2, \"groupid\": 0, \"userid\": 5, \"repeatid\": 0, \"modulename\": \"assign\", \"instance\": 2, \"eventtype\": \"due\", \"timestart\": 1403740500, \"timeduration\": 0, \"visible\": 1, \"uuid\": \"\", \"sequence\": 1, \"timemodified\": 1403089640, \"subscriptionid\": null}, {"id": 2, "name": "Insturen Functionele specificaties", "description": "\n<p>Insturen Functionele specificaties</p>\", \"format\": 1, \"courseid\": 2, \"groupid\": 0, \"userid\": 3, \"repeatid\": 0, \"modulename\": \"assign\", \"instance\": 3, \"eventtype\": \"due\", \"timestart\": 1404431700, \"timeduration\": 0, \"visible\": 1, \"uuid\": \"\", \"sequence\": 1, \"timemodified\": 1404320762, \"subscriptionid\": null}, {"id": 3, "name": "Use Cases", "description": "\n<p><\\p>\", \"format\": 1, \"courseid\": 2, \"groupid\": 0, \"userid\": 5, \"repeatid\": 0, \"modulename\": \"assign\", \"instance\": 4, \"eventtype\": \"due\", \"timestart\": 1405036500, \"timeduration\": 0, \"visible\": 1, \"uuid\": \"\", \"sequence\": 1, \"timemodified\": 1404501964, \"subscriptionid\": null}, {"id": 4, "name": "Inleveren klassendiagram", "description": "\n<p><\\p>\", \"format\": 1, \"courseid\": 2, \"groupid\": 0, \"userid\": 5, \"repeatid\": 0, \"modulename\": \"assign\", \"instance\": 5, \"eventtype\": \"due\", \"timestart\": 1406194200, \"timeduration\": 0, \"visible\": 1, \"uuid\": \"\", \"sequence\": 1, \"timemodified\": 1405591393, \"subscriptionid\": null}, {"id": 5, "name": "Inleveren ER-diagram", "description": "\n<p><\\p>\", \"format\": 1, \"courseid\": 2, \"groupid\": 0, \"userid\": 5, \"repeatid\": 0, \"modulename\": \"assign\", \"instance\": 6, \"eventtype\": \"due\", \"timestart\": 1406246100, \"timeduration\": 0, \"visible\": 1, \"uuid\": \"\", \"sequence\": 1, \"timemodified\": 1405591374, \"subscriptionid\": null}, {"id": 6, "name": "Code deel 1", "description": "\n<p>Bouw aan de hand van het klassendiagram \u2225 ER diagram de business-layer, en eventueel al (een deel van) de data-access laag.\n\\u00a0a<\\p>\n\n<p><\\p>\", \"format\": 1, \"courseid\": 2, \"groupid\": 0, \"userid\": 5, \"repeatid\": 0, \"modulename\": \"assign\", \"instance\": 7, \"eventtype\": \"due\", \"timestart\": 1406847300, \"timeduration\": 0, \"visible\": 1, \"uuid\": \"\", \"sequence\": 1, \"timemodified\": 1405591037, \"subscriptionid\": null}], \"warnings\": []}]}]

Json.NET bevat een `JObject` classe met een `Parse` methode.

Deze methode gebruikt een Json string en veranderd deze in een geneste Key-Value structuur van het type Object.

Na de parse ziet de data er zo uit:

```
{  
  "events": [  
    {  
      "id": 1,  
      "name": "Project Requirements deel 1 - Cursisten",  
      "description": "<p>CursistenPortaal: welke functionaliteiten vanuit het standpunt van de cursist, en hoe die best opvraagbaar en  
      gepresenteerd kunnen worden.<br />Structuur je ideeën hierover in een tekst, in te leveren als Word-document (minimaal 1 pagina) tem  
      laatste 25-2, individueel op te laden. Hou ook je logboek bij (cfr. logboek template)!</p>",  
      "format": 1,  
      "courseid": 2,  
      "groupid": 0,  
      "userid": 5,  
      "repeatid": 0,  
      "modulename": "assign",  
      "instance": 2,  
      "eventtype": "due",  
      "timestart": 1403740500,  
      "timeduration": 0,  
      "visible": 1,  
      "uuid": "",  
      "sequence": 1,  
      "timemodified": 1403089640,  
      "subscriptionid": null  
    },  
    {  
      "id": 2,  
      "name": "Insturen Functionele specificaties",  
      "description": "<p>Insturen Functionele specificaties</p>",  
      "format": 1,  
      "courseid": 2,  
      "groupid": 0,  
      "userid": 3,  
      "repeatid": 0,  
      "modulename": "assign",  
      "instance": 3,  
      "eventtype": "due",  
      "timestart": 1404431700,  
      "timeduration": 0,  
      "visible": 1,  
      "uuid": "",  
      "sequence": 1,  
      "timemodified": 1404320762,  
      "subscriptionid": null  
    },  
    {  
      "id": 3,  
      "name": "Use Cases ",  
      "description": "<p>_</p>",  
      "format": 1,  
      "courseid": 2,  
      "groupid": 0,  
      "userid": 5,  
      "repeatid": 0,  
      "modulename": "assign",  
      "instance": 4,  
      "eventtype": "due",  
      "timestart": 1404431700,  
      "timeduration": 0,  
      "visible": 1,  
      "uuid": "",  
      "sequence": 1,  
      "timemodified": 1404320762,  
      "subscriptionid": null  
    }  
  ]  
}
```

```

        "courseid": 2,
        "groupid": 0,
        "userid": 5,
        "repeatid": 0,
        "modulename": "assign",
        "instance": 4,
        "eventtype": "due",
        "timestart": 1405036500,
        "timeduration": 0,
        "visible": 1,
        "uuid": "",
        "sequence": 1,
        "timemodified": 1404501964,
        "subscriptionid": null
    },
    {
        "id": 4,
        "name": "Inleveren klassediagram",
        "description": "<p>_</p>",
        "format": 1,
        "courseid": 2,
        "groupid": 0,
        "userid": 5,
        "repeatid": 0,
        "modulename": "assign",
        "instance": 5,
        "eventtype": "due",
        "timestart": 1406194200,
        "timeduration": 0,
        "visible": 1,
        "uuid": "",
        "sequence": 1,
        "timemodified": 1405591393,
        "subscriptionid": null
    },
    {
        "id": 5,
        "name": "Inleveren ER-diagram",
        "description": "<p>_</p>",
        "format": 1,
        "courseid": 2,
        "groupid": 0,
        "userid": 5,
        "repeatid": 0,
        "modulename": "assign",
        "instance": 6,
        "eventtype": "due",
        "timestart": 1406246100,
        "timeduration": 0,
        "visible": 1,
        "uuid": "",
        "sequence": 1,
        "timemodified": 1405591374,
        "subscriptionid": null
    },
    {
        "id": 6,
        "name": "Code deel 1",
        "description": "<p>Bouw aan de hand van het klassediagram / ER diagram de business-layer, en eventueel al (een deel van) de data-access laag. </p>\n<p></p>",
        "format": 1,
        "courseid": 2,
        "groupid": 0,
        "userid": 5,
        "repeatid": 0,
        "modulename": "assign",
        "instance": 7,
        "eventtype": "due",
        "timestart": 1406847300,
        "timeduration": 0,
        "visible": 1,
        "uuid": "",
        "sequence": 1,
        "timemodified": 1405591037,
        "subscriptionid": null
    }
],
"warnings": []
}

```

Nu kunnen we eenvoudiger informatie van alle kalender events opvragen.

Zo kunnen we met een foreach door alle elementen gaan om specifieke data te tonen.

Met volgende code kan je de namen van alle events afdrukken.

```

foreach (JObject event in assignments["events"])
{
    Console.WriteLine(event["id"] + ": " + event["name"]);
}

```

Output:

```

1: Project Requirements deel 1 - Cursisten
2: Insturen Functionele specificaties
3: Use Cases
4: Inleveren klassediagram
5: Inleveren ER-diagram

```

Sources

<http://www.webthingsconsidered.com/2013/08/09/adventures-in-json-parsing-with-c/>

Auteurs:

Nikos Vanden Broek

Moodle Cursussen

maandag 6 april 2015

16:30

Cursus creëren

Om via web services informatie op te kunnen halen over deadlines, moeten we in onze testomgeving eerst cursussen en taken creëren.

Om een cursus aan te kunnen maken moeten we ingelogd zijn als administrator.

Daarna navigeren we naar *Sitebeheer -> Cursussen -> Beheer cursussen en categorieën*.

Als eerste moeten we een categorie aanmaken waartoe onze cursussen behoren.

We klikken op 'Maak een nieuwe categorie', geven de nieuwe categorie een naam, en maken de categorie aan.

Nu kunnen we een cursus aanmaken voor onze categorie.

We zorgen dat we in de juiste categorie zitten door links op de categorie naam te klikken zodat deze rechts te zien is.

Nu klikken we op 'Maak een nieuwe cursus', vullen de nodige gegevens in, en slaan de cursus op.

Eens opgeslagen komen we op een pagina waar we cursisten kunnen toevoegen aan de cursus.

Meld de nodige cursisten aan en ga terug naar Mijn cursussen, waar we onze nieuwe cursus kunnen zien.

Taken toevoegen

We selecteren de cursus uit de lijst in Mijn cursussen zodat we de cursus aan kunnen passen.

Onder Cursusbeheer zetten we wijzigen aan door er op te klikken.

Om een taak toe te voegen klikken we op een van de 'Activiteit of Bron toevoegen' opties, en selecteren 'Opdracht' uit de lijst.

We vullen de nodige gegevens in, en slaan de opdracht op.

Auteurs:

Nikos Vanden Broek

Algemeen

woensdag 3 juni 2015

17:56

Probleem

In het project moeten we regelmatig data kunnen bijhouden die bij elkaar hoort.

Hiervoor kunnen we klassen maken die waarvan we later objecten kunnen maken om data gestructureerd in bij te houden.

Deze klassen worden aangemaakt in de Business Logic Layer (BLL)

Design

Een BLL klasse aanmaken is redelijk eenvoudig.

We maken een publieke klasse aan en geven deze een duidelijk herkenbare naam.

```
public class Module { }
```

In de klasse kunnen maken we verschillende properties.

Deze dienen als een soort skelet waar later data in opgeslagen kan worden, vergelijkbaar met een rij in een databank.

In het project is al de data redelijk eenvoudig waardoor we me simpele get en set kunnen werken.

De properties kunnen eender welk soort data bevatten zoals int, string, bool, maar ook andere klassen en lijsten.

```
public int Id { get; set; }
public string Code { get; set; }
public bool CursistIsIngeschreven { get; set; }
public bool CursistIsGeslaagd { get; set; }
public bool CursistHeeftVoorkennis { get; set; }
public double PuntenTotaal { get; set; }
public string Naam { get; set; }
public int Lestijden { get; set; }
public List<Module> VoorkeennisModules { get; set; }
```

Als een van de properties een lijst is of een ander object dat geïnitialiseerd moet worden, moet dit in de constructor gebeuren.

Als we dit niet doen kunnen er mogelijk nullpointer errors ontstaan.

```
public Module()
{
    VoorkeennisModules = new List<Module>();
```

De klassen kunnen ook andere methodes bevatten.

Voorbeelden hiervan kun je vinden op de andere pagina's.

Oplossing

```
public class Module
{
    public int Id { get; set; }
    public string Code { get; set; }
    public bool CursistIsIngeschreven { get; set; }
    public bool CursistIsGeslaagd { get; set; }
    public bool CursistHeeftVoorkennis { get; set; }
    public double PuntenTotaal { get; set; }
    public string Naam { get; set; }
    public int Lestijden { get; set; }
    public List<Module> VoorkeennisModules { get; set; }

    public Module()
    {
        VoorkeennisModules = new List<Module>();
    }
}
```

}

Auteurs:

Nikos Vanden Broek

Lessenrooster

woensdag 3 juni 2015

17:57

Auteurs:

Nikos Vanden Broek
Mohamed Amajoutt

Probleem

Op de lessenrooster pagina willen we een optie voor de gebruiker om ook de feestdagen te tonen.
Hiervoor moeten we kunnen bijhouden welke optie de gebruiker heeft gekozen.
Ook moeten we ervoor zorgen dat we objecten van het type KalenderDag kunnen toevoegen aan de lijst van Lesdavinci.

Design

Static property

Om de keuze van de gebruiker bij de houden maken we gebruik van een static property.
Dit zorgt ervoor dat de keuze dezelfde blijft voor al de objecten, en dat we de waarde kunnen opvragen en aan passen zonder een object van de klasse te maken.

```
public static bool FeestdagenTonen { get; set; }
```

Feestdagen

Om de feestdagen toe te voegen moeten we deze dagen omzetten in LesDavinci objecten zodat ze toegevoegd kunnen worden aan de lijst van lesdagen.

Hiervoor hebben we een methode nodig die een lijst van het type KalenderDag gebruikt.

```
public void VoegFeestdagenToe(List<BLL.KalenderDag> feestdagen) { }
```

We gebruiken een foreach lus om de lijst van feestdagen te doorlopen.

In de databank is elke dag in een week van feestdagen (bv winter vakantie) appart opgeslagen.

Het begin en einde van de vakantie zijn benoemd maar de dagen tussen in hebben geen naam.

Voor de lessenrooster willen we deze dagen niet tonen, dus filteren we ze.

```
foreach (BLL.KalenderDag k in feestdagen)
{
    if (!k.Omschrijving.Equals("")) { }
}
```

Nu kunnen we voor elke feestdag een Lesdavinci object maken.

De datum kunnen we overnemen van de feestdag en de omschrijving van de feestdag kunnen we gebruiken als naam van de module.

De andere properties krijgen een default waarde bij het maken van het object, dus moet we deze niet meegeven.

Het object kunnen we nu aan de lijst van les dagen toevoegen.

```
BLL.LesDavinci dag = new BLL.LesDavinci();
dag.Datum = k.Datum;
dag.Module = k.Omschrijving;
Lesdagen.Add(dag);
```

Als laatste moeten we de lijst opnieuw sorteren op datum.

Om zeker te zijn dat lesmomenten op dezelfde dag ook in de juiste volgorde staan, sorteren we ook op aanvangsdatum.

```
Lesdagen = Lesdagen.OrderBy(o => o.Datum).ThenBy(o => o.Aanvangsdatum).ToList();
```

Oplossing

```
public class Lessenrooster
{
```

```

public static bool FeestdagenTonen { get; set; }
public List<BLL.LesDavinci> Lesdagen { get; set; }

public Lessenrooster()
{
    Lesdagen = new List<LesDavinci>();
}

/// <summary>
/// Voegt feestdagen toe aan de lijst van lesdagen
/// </summary>
/// <param name="feestdagen">Lijst met feestdagen</param>
public void VoegFeestdagenToe(List<BLL.KalenderDag> feestdagen)
{
    foreach (BLL.KalenderDag k in feestdagen)
    {
        if (!k.Omschrijving.Equals(""))
        {
            BLL.LesDavinci dag = new BLL.LesDavinci();
            dag.Datum = k.Datum;
            dag.Module = k.Omschrijving;

            Lesdagen.Add(dag);
        }
    }
    Lesdagen = Lesdagen.OrderBy(o => o.Datum).ThenBy(o =>
o.Aanvangsdatum).ToList();
}
}

```

Kalender

woensdag 3 juni 2015

17:58

Auteurs:

Nikos Vanden Broek
Mohamed Amajoutt

Probleem

Voor de kalender willen we altijd een hele maand laten zien en de altijd volledige weken tonen.
Hiervoor worden de begin en eind datums berenkend.

Aan de hand van deze datums moeten we verschillende kalender dagen aanmaken.

Ook moeten het mogelijk zijn om aan de verschillende kalender dagen verschillende lesmomenten, feestdagen en deadlines toe te voegen.

Design

Kalender dagen

Gezien we de begin en eind datums weten, moeten we een methode aanmaken die deze kan gebruiken.

```
public void CreerDagen(DateTime startDatum, DateTime eindDatum) { }
```

We willen altijd dat de methode een nieuwe lijst van dagen begint, dus overschijven we de vorige lijst met een nieuwe lijst.

We houden ook de begin en eind datum bij om later te gebruiken.

```
Dagen = new List<KalenderDag>();  
this.StartDatum = startDatum;  
this.EindDatum = eindDatum;
```

We gebruiken een while loop waarin we voor elke dag tussen begin en einde een KalenderDag object aanmaken, de datum toevoegen, en aan de lijst van dagen toevoegen.

```
while (startDatum.Date <= eindDatum.Date)  
{  
    BLL.KalenderDag dag = new BLL.KalenderDag();  
    dag.Datum = startDatum;  
    Dagen.Add(dag);  
    startDatum = startDatum.AddDays(1);  
}
```

Momenten toevoegen

De methodes voor de verschillende momenten werken hebben dezelfde werkwijze. We gebruiken de feestdagen als voorbeeld.

We beginnen met een methode die een lijst van KalenderDag objecten vraagt.

```
public void VoegFeestenToe(List<BLL.KalenderDag> feestDagen)  
{ }
```

We gebruiken een foreach loop om door de feestdagen te gaan.

Voor elke dag vergelijken we de datum met de startdatum van de kalender.

Zo weten we de hoeveelste dag van de kalender, en dus ook in de lijst van dagen, de feestdag op valt.

```
foreach (BLL.KalenderDag dag in feestDagen)  
{  
    int nr = (dag.Datum - StartDatum).Days;  
}
```

Om zeker te zijn dat da datum van de feestdag wel zeker in de lijst van kalender dagen valt, en dus ook in de range van de lijst, moeten we zeker zijn dat de nr minstens 0 is and niet groter dan de groote van de lijst.

Als dit het geval is voegen we de feestdag toe aan de lijst van feestdagen, voor de kaldenderdag in de lijst.

```
if (nr >= 0 && nr <= Dagen.Count)
{
    Dagen[nr].Feesten.Add(dag);
}
```

Oplossing

```
public class Kalender
{
    public List<BLL.KalenderDag> Dagen { get; set; }
    public DateTime StartDatum { get; set; }
    public DateTime EindDatum { get; set; }

    public Kalender()
    {
        Dagen = new List<KalenderDag>();
    }

    public void CreerDagen(DateTime startDatum, DateTime eindDatum)
    {
        Dagen = new List<KalenderDag>();

        this.StartDatum = startDatum;
        this.EindDatum = eindDatum;

        while (startDatum.Date <= eindDatum.Date)
        {
            BLL.KalenderDag dag = new BLL.KalenderDag();
            dag.Datum = startDatum;
            Dagen.Add(dag);

            startDatum = startDatum.AddDays(1);
        }
    }

    public void VoegFeestenToe(List<BLL.KalenderDag> feestDagen)
    {
        foreach (BLL.KalenderDag dag in feestDagen)
        {
            //tel dagen van begin = dag in lijst
            int nr = (dag.Datum - StartDatum).Days;

            //safeguard
            if (nr >= 0 && nr <= Dagen.Count)
            {
                Dagen[nr].Feesten.Add(dag);
            }
        }
    }

    public void VoegLessenToe(List<BLL.LesDavinci> lesDagen)
    {
        foreach (BLL.LesDavinci les in lesDagen)
        {
            //tel dagen van begin = dag in lijst
            int nr = (les.Datum - StartDatum).Days;

            //safeguard
            if (nr >= 0 && nr <= Dagen.Count)
            {
                Dagen[nr].Lessen.Add(les);
            }
        }
    }
}
```

```
        }

    }

    public void VoegDeadlineToe(List<Moodle.BLL.Deadline> deadlines)
{
    foreach (Moodle.BLL.Deadline deadline in deadlines)
    {
        //tel dagen van begin = dag in lijst
        int nr = (deadline.Date - StartDatum).Days;

        //safeguard
        if (nr >= 0 && nr <= Dagen.Count)
        {
            Dagen[nr].Deadlines.Add(deadline);
        }
    }
}
```

Auteurs:

Nikos Vanden Broek

Trajectoverzicht

woensdag 3 juni 2015

17:59

Probleem

Voor het traject overzicht moeten willen we een lijst van alle modules die gevuld kunnen worden, de status van de modules voor de cursist en de nodige voorkennis.

Design

Module Status

Om de status van een module te weten gebruiken we een methode die een lijst van CursistResultaten gebruikt.

```
public void VoegModuleResultatenToe(List<CursusResultaat> resultaten) { }
```

Om de resultaten aan de correcte module te koppelen, moeten we de ModulevariantId van de module en het resultaat vergelijken.

Dit kunnen we doen door een foreach loop voor de resultaten, in een foreach loop voor de modulen te plaatsen.

```
foreach (BLL.Module module in this.TrajectModules)
{
    foreach (BLL.CursusResultaat resultaat in resultaten) { }
```

Als de ModulevariantId van de twee gelijk zijn weten we al dat de cursist ingeschreven is in de module, en kunnen we het puntentotaal invullen.

Als het puntentotaal maar dan 50 is weten we ook dat de cursist geslaagd is voor de module.

```
if (resultaat.IdModuleVariant == module.Id)
{
    module.CursistIsIngeschreven = true;
    module.PuntenTotaal = resultaat.PuntenTotaal;
    if (module.PuntenTotaal > 50)
    {
        module.CursistIsGeslaagd = true;
    }
}
```

Nadat we alle resultaten doorlopen hebben, sorteren we de modules op naam om de lijst leesbaarder te maken voor de gebruiker.

```
this.TrajectModules = this.TrajectModules.OrderBy(o => o.Naam).ToList();
```

Voorkennis

Om de voorkennis van de modules aan elkaar te kunnen linken, hebben we een lijst van KeyValuePair met Ids.

De key is de ModulevariantId van de module waar de voorkennis voor geldt.

De value is de ModulevariantId van de module die de voorkennis is van de andere module.

```
public void LinkVoorkennisMetKeyValuePair(List<KeyValuePair<int, int>>
voorkennisPairs) { }
```

We doorlopen de lijst van KeyValuePair's met een foreach loop.

Om de correcte module te vinden in de lijst gebruiken we de FindModuleById() methode (zie verder).

Deze methode gebruiken we om het module object en om hetvoorkennis module object op te vragen.

Daarna kunnen we het voorkennis object toevoegen aan de lijst van voorkennis in het module object.

```
foreach (KeyValuePair<int, int> pair in voorkennisPairs)
{
    BLL.Module trajectModule = FindModuleById(pair.Key);
```

```

        BLL.Module voorkennisModule = FindModuleById(pair.Value);
        trajectModule.VoorkennisModules.Add(voorkennisModule);
    }

```

Eens alle voorkennis is gelinkt gaan we nog door de lijst van modules om na te gaan of de cursist de nodige voorkennis heeft om de module te starten.

Hiervoor gebruiken we de HeeftNodigeVoorkennis() die verder nog besproken wordt.

```

foreach (BLL.Module module in this.TrajectModules)
{
    module.CursistHeeftVoorkennis = HeeftNodigeVoorkennis(module);
}

```

FindModuleById

Omdat eenvoudig een module uit de lijst van modules te kunnen halen, maken we een private methode die een int nodig heeft voor de id van de module en het module object terug geeft.

```
private BLL.Module FindModuleById(int id) { }
```

Om de module te vinden gebruiken we een eenvoudige foreach loop en vergelijken we de id met de id van de module in de lijst.

```

BLL.Module module = new BLL.Module();
foreach (BLL.Module m in this.TrajectModules)
{
    if (m.Id == id)
    {
        module = m;
    }
}
return module;

```

HeeftNodigeVoorkennis

Om na te gaan of de cursist de nodige voorkennis heeft voor een module, gebruiken we een methode die een module object vraagt en een bool terug geeft of de cursist al dan niet de voorkennis bezit.

```
private bool HeeftNodigeVoorkennis(BLL.Module module) { }
```

We kunnen de voorkennis na gaan door, van elke voorkennis module, te kijken of de cursist ingeschreven is en geslaagd is.

Als dit van een van de modules niet het geval is weten we meteen dat de cursist niet de voorkennis heeft en geven we false terug.

```

foreach (BLL.Module voorkennis in module.VoorkennisModules)
{
    if (!voorkennis.CursistIsIngeschreven || !voorkennis.CursistIsGeslaagd)
    {
        return false;
    }
}
return true;

```

Oplossing

```

public class TrajectOverzicht
{
    public List<BLL.Module> TrajectModules { get; set; }

    public TrajectOverzicht()
    {
        this.TrajectModules = new List<Module>();
    }

    /// <summary>
    /// Voegt cursist resultaten toe aan de modules die tot het traject behoren
    /// </summary>

```

```

/// <param name="resultaten">Cursust resultaten</param>
public void VoegModuleResultatenToe(List<CursusResultaat> resultaten){
    foreach (BLL.Module module in this.TrajectModules)
    {
        foreach (BLL.CursusResultaat resultaat in resultaten)
        {
            if (resultaat.IdModuleVariant == module.Id)
            {
                module.CursistIsIngeschreven = true;
                module.PuntenTotaal = resultaat.PuntenTotaal;
                if (module.PuntenTotaal > 50)
                {
                    module.CursistIsGeslaagd = true;
                }
            }
        }
    }

    this.TrajectModules = this.TrajectModules.OrderBy(o => o.Naam).ToList();
}

/// <summary>
/// Linkt modules in het traject aan andere modules in het traject als
voorkennis
/// </summary>
/// <param name="voorkennisPairs">Voorkennis links tussen modules</param>
public void LinkVoorkennisMetKeyValuePair(List<KeyValuePair<int, int>>
voorkennisPairs)
{
    foreach (KeyValuePair<int, int> pair in voorkennisPairs)
    {
        BLL.Module trajectModule = FindModuleById(pair.Key);
        BLL.Module voorkennisModule = FindModuleById(pair.Value);
        trajectModule.VoorkennisModules.Add(voorkennisModule);
    }

    foreach (BLL.Module module in this.TrajectModules)
    {
        module.CursistHeeftVoorkennis = HeeftNodigeVoorkennis(module);
    }
}

/// <summary>
/// Kijkt of de cursist de nodige voorkennis heeft voor een module
/// </summary>
/// <param name="module">Module die bekennen moet werken</param>
/// <returns>Of een cursist al dan niet de nodige voorkennis heeft</returns>
private bool HeeftNodigeVoorkennis(BLL.Module module)
{
    foreach (BLL.Module voorkennis in module.VoorkennisModules)
    {
        if (!voorkennis.CursistIsIngeschreven || !
voorkennis.CursistIsGeslaagd)
        {
            return false;
        }
    }
    return true;
}

/// <summary>
/// Zoekt een module in de modules die tot het project behoren aan de hand van
zijn Id
/// </summary>
/// <param name="id">Id van de module</param>
/// <returns>De module met de overeenstemmende Id</returns>
private BLL.Module FindModuleById(int id)
{
    BLL.Module module = new BLL.Module();

```

```
foreach (BLL.Module m in this.TrajectModules)
{
    if (m.Id == id)
    {
        module = m;
    }
}
return module;
```

Auteurs:

Nikos Vanden Broek

BLL Evenement

donderdag 4 juni 2015

1:50

Auteurs:

Mohamed Amajoutt

Probleem:

Op de evenementen pagina staat een lijst van alle evenementen die nog zullen komen. Hierin moet alle informatie zitten over het evenement en de evenementen zijn nog niet verlopen.

Design:

Variabelen

In de BLL klasse voor Evenement zitten enkel de variabelen die we nodig hebben uit de Evenement tabel.

Naam	Bereik	Beschrijving
Id	Public	Primary key
Naam	Public	Naam van het evenement
Datum	Public	
Locatie	Public	
Start	Public	Startuur van evenement
Eind	Public	Einduur van evenement

Methoden

In elke BLL klasse die we hebben gemaakt hebben we voor elke variabele getters en setters gemaakt. Dit hebben we simpel opgelost door achter de variabele "{get; set;}" te plaatsen. Hierdoor geven we rechten om deze op te halen en later aan te passen in de DAL.

Oplossing:

De BLL klasse hebben we in de folder Administratix geplaatst met als naam "BLL.cs". Deze geven we ook de namespace Administratix.BLL omdat we deze gemakkelijker kunnen oproepen en overzichtelijker is. We kozen ervoor om alle entiteiten in verschillende klassen te steken in de klasse "BLL.cs".

```
/// <summary>
/// Class definition Evenement
/// Used in DAL.Evenement
/// </summary>
public class Evenement
{
    public int Id { get; set; }
    public string Naam { get; set; }
    public DateTime Datum { get; set; }
    public string Locatie { get; set; }
    public DateTime Start { get; set; }
    public DateTime Eind { get; set; }

}
```

BLL EvenementInschrijving

zaterdag 6 juni 2015

23:41

Auteurs:

Mohamed Amajoutt

Probleem:

Wanneer de cursist in de Evenementenpagina op de knop "Schrijf me in" drukt dan wordt de cursist ingeschreven voor het evenement. De cursist moet ook een lijst terugkrijgen voor welke evenementen men zijn ingeschreven.

Design:

Variabelen

In de BLL klasse voor EvenementInschrijving zitten de variabelen die we nodig hebben uit de EvenementInschrijving tabel.

Naam	Bereik	Beschrijving
Id	Public	Primary key
ReservatieDatum	Public	Deze datum wordt automatisch ingevuld in de database wanneer de cursist op de knop "schrijf me in" drukt.
IdCursist	Public	Foreign key met de Id van de tabel Cursist
IdEvenement	Public	Foreign key met de Id van de tabel Evenement
Naam	Public	Naam van het evenement
Datum	Public	Datum van het evenement
Locatie	Public	Locatie van het evenement
StartUur	Public	Start van het evenement
EindUur	Public	
Opmerkingen	Public	

Methoden

In elke BLL klasse die we hebben gemaakt hebben we voor elke variabele getters en setters gemaakt. Dit hebben we simpel opgelost door achter de variabele "{get; set;}" te plaatsen. Hierdoor geven we rechten om deze op te halen en later aan te passen in de DAL.

Oplossing:

De BLL klasse hebben we in de folder Administratix geplaatst met als naam "BLL.cs". Deze geven we ook de namespace Administratix.BLL omdat we deze gemakkelijker kunnen oproepen en overzichtelijker is. We kozen ervoor om alle entiteiten in verschillende klassen te steken in de klasse "BLL.cs".

```
/// <summary>
/// Class definition EvenementInschrijving
/// Used in DAL.EvenementInschrijving
/// </summary>
public class EvenementInschrijving
{
```

```
public int Id { get; set; }
public DateTime ReservatieDatum { get; set; }
public int IdCursist { get; set; }
public int IdEvenement { get; set; }
public string Naam { get; set; }
public DateTime Datum { get; set; }
public string Locatie { get; set; }
public DateTime StartUur { get; set; }
public DateTime EindUur { get; set; }
public string Opmerkingen { get; set; }
}
```

ConnectionString

donderdag 4 juni 2015

2:22

Probleem:

Om verbinding te maken met de SQL Server gebruiken we een connectionString in de web.config. Dit vergemakkelijkt om telkens de connectie te maken als we een call moesten doen. De connectionString bevat de server ip adres, database naam, gebruikersnaam, paswoord.

Design:

Sleutelwoorden van connectionString

In de web.config steken we sleutelwoorden, gevolgd door de waarde in de connectionString.

Server	N/A	De naam of netwerkadres van de SQL Server om te connecteren.
Database	N/A	De naam van de database.
UID	N/A	De SQL Server gebruikersnaam om in te loggen.
PWD	N/A	De SQL Server paswoord om in te loggen.

De connectionString geven we ook een benaming om deze gemakkelijk op te roepen, zonder telkens alle sleutelwoorden en waardes mee te geven.

Oplossing:

```
<connectionStrings>
    <remove name="MobileCVO"/>
    ?  <add name="MobileCVO" connectionString="Server=
        92.222.220.213,1501;Database=Administratix_cursist;Uid=sa;Pwd=#####;" />
</connectionStrings>
```

Interface IDal

donderdag 4 juni 2015
2:54

Auteurs:

Mohamed Amajoutt

Probleem:

Alle klassen bevatten een stuk gemeenschappelijke code. In plaats van telkens in elke DAL klasse methodes te herschrijven kiezen we ervoor om een interface te implementeren. Deze zal dan alle methodes die in de interface IDal toevoegen aan de klasse.

Hierdoor hebben we een pak minder werk moeten verrichten en bovendien zeer overzichtelijk.

Design:

Variabelen

Naam	Bereik	Beschrijving
Message	Protected	

Methoden

Methode	Bereik	Parameter	Retour	Beschrijving
Insert	Public	TypeEntity Entity	int	Neemt een object van BLL als parameter en geeft een int terug.
SelectAll	Public		Lijst van een object	Neemt geen parameters en geeft een lijst terug.
SelectAllByCursistNummer	Public	Int	Lijst van een object	Neemt een int cursistNummer als parameter en geeft lijst terug.

Oplossing:

```
interface IDal<TypeEntity>
{
    string Message {get;}
    int Insert(TypeEntity entity);
    List<TypeEntity> SelectAll();
    List<TypeEntity> SelectAllByCursistNummer(int cursistNummer);
}
```

DAL Evenement

donderdag 4 juni 2015
2:51

Auteurs:

Mohamed Amajoutt

Probleem:

De code die zorgt voor het ophalen en manipuleren van de ruwe gegevens opgeslagen in de database. Deze code zal ook de connectieString telkens oproepen. We hebben een lijst van evenementen moeten ophalen uit de database. Deze zal rij per rij worden ingelezen vanuit de database en ingelezen in Evenement uit de BLL klasse. Vervolgens moeten we al deze gegevens in lijst steken.

Design:

Om te beginnen maken we de klasse Evenement aan in DAL.cs en laten we deze de interface IDal implementeren. De methode List<BLL.Evenement> SelectAll gaat vervolgens een lijst van evenementen (BLL.Evenement) teruggeven.

```
public class Evenement : MobileCVO.DAL.IDal<BLL.Evenement>

{
    public List<BLL.Evenement> SelectAll()
    {
        }
}
```

Connectie maken

Vervolgens maken we een lijst aan om de les momenten bij te houden.

```
List<BLL.Evenement> evenementenLijst = new List<BLL.Evenement>();
```

Nu kunnen we beginnen om de connectie met de databank te maken.

Eerst maken we een SqlConnection object aan waarvan we de als ConnectionString property de connection string meegeven die opgeslagen is in de web.config file.

```
SqlConnection connection = new SqlConnection();
connection.ConnectionString =
    System.Configuration.ConfigurationManager.ConnectionStrings["MobileCVO"].ToString();
```

Hierna maken we een SqlCommand object aan die de nodige informatie bevat om de database aan te sturen zodat we de nodig informatie terug krijgen.

Aan dit command geven we mee welk soort type command we gebruiken en de nodige tekst. We gebruiken het type StoredProcedure en in dit voorbeeld de tekst is de naam van de stored procedure: grp2_SelectAllEvenement

```
SqlCommand command = new SqlCommand();
command.CommandType = CommandType.StoredProcedure;
command.CommandText = "grp2_SelectAllEvenement";
```

Als laatste moeten we de connection, die we eerder hebben aangemaakt, toevoegen aan het command zodat deze de nodige connectie kan maken.

```
command.Connection = connection;
```

Data opvangen

Nu we de basis hebben om een connectie te maken, moeten we ervoor zorgen dat we de data die we terug krijgen,

kunnen opvangen en omzetten in data die we in ons programma kunnen gebruiken.

Omdat er Sql fouten kunnen gebeuren werken we in een try catch.

We beginnen met een SqlDataReader waar we het resultaat van de database interactie zullen opvangen, en we openen de connectie.

We eindigen de try catch ook met een finally waarin we de connectie terug sluiten.

Hierdoor zijn we altijd zeker dat de connectie gesloten wordt (ook bij foutmeldingen) zodat we op een ander moment een nieuwe connectie kunnen openen.

Als de connectie geopend is kunnen we het command uitvoeren en de data die we terugkrijgen opslaan in de SqlDataReader die we eerder aan hebben gemaakt.

Vervolgens kijken we of we wel degelijk data hebben teruggekregen.

Nu kunnen we met de .Read() methode van de SqlDataReader rij per rij door de teruggekregen gegevens lezen en omzetten in data die we in het programma kunnen gebruiken

In het voorbeeld van de evenementen lijst maken we telkens een Evenementobject aan.

In dit object kunnen we de properties toekennen door de waarde op te vragen met dezelfde naam als de waarden in de stored procedure.

We moeten deze data telkens converteren naar de juiste type van de properties.

try

```
    {
        connection.Open();
        this.message = "De database is klaar!";

        using (result = command.ExecuteReader())
        {
            if (result.HasRows)
            {
                while (result.Read())
                {
                    BLL.Evenement evenement = new BLL.Evenement();
                    evenement.Id = (int)result["Id"];
                    evenement.Naam = result["Naam"].ToString();
                    evenement.Datum = Convert.ToDateTime(result["Datum"]);
                    evenement.Locatie = result["Locatie"].ToString();
                    evenement.Start = Convert.ToDateTime(result["StartUur"]);
                    evenement.Eind = Convert.ToDateTime(result["EindUur"]);
                    evenementenLijst.Add(evenement);
                }
            }
        }
    catch (SqlException e)
    {
        this.message = e.Message;
    }
    finally
    {
        connection.Close();
    }
```

Oplossing:

```
public class Evenement : MobileCVO.DAL.IDal<BLL.Evenement>
{
    public List<BLL.Evenement> SelectAll()
    {
        List<BLL.Evenement> evenementenLijst = new List<BLL.Evenement>();

        SqlConnection connection = new SqlConnection();
        connection.ConnectionString =
            System.Configuration.ConfigurationManager.
            ConnectionStrings["MobileCVO"].ToString();

        SqlCommand command = new SqlCommand();

        string sqlString = "grp2_SelectAllEvenement";
```

```

        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = sqlString;
        command.Connection = connection;
        this.message = "Niets te melden";
        SqlDataReader result;
        try
        {
            connection.Open();
            this.message = "De database is klaar!";
            using (result = command.ExecuteReader())
            {
                if (result.HasRows)
                {
                    while (result.Read())
                    {
                        BLL.Evenement evenement = new BLL.Evenement();
                        evenement.Id = (int)result["Id"];
                        evenement.Naam = result["Naam"].ToString();
                        evenement.Datum = Convert.ToDateTime(result["Datum"]);
                        evenement.Locatie = result["Locatie"].ToString();
                        evenement.Start = Convert.ToDateTime(result["StartUur"]);
                        evenement.Eind = Convert.ToDateTime(result["EindUur"]);
                        evenementenLijst.Add(evenement);
                    }
                }
            }
            catch (SqlException e)
            {
                this.message = e.Message;
            }
            finally
            {
                connection.Close();
            }
        }
        return evenementenLijst;
    }
}

```

DAL EvenementInschrijving

zaterdag 6 juni 2015
23:51

Auteurs:

Mohamed Amajoutt

Probleem:

De code die zorgt voor het ophalen en manipuleren van de ruwe gegevens opgeslagen in de database. Deze code zal ook de connectieString telkens oproepen. We hebben een lijst van EvenementInschrijving moeten ophalen uit de database. Deze zal rij per rij worden ingelezen vanuit de database en ingelezen in EvenementInschrijving uit de BLL klasse.

We moeten ook een inschrijving voor een evenement kunnen doen adhv IdCursist, IdEvenement. De opmerkingen is optioneel om mee te geven.

Design:

Om te beginnen maken we de klasse EvenementInschrijving aan in DAL.cs en laten we deze de interface IDal implementeren. De methode List<BLL.Evenement> SelectAllByCursistNummer voeren we uit met als parameter CursistNummer omdat we een persoonlijke lijst van EvenementenInschrijving (BLL.EvenementInschrijving) teruggeven.

```
public class EvenementInschrijving : MobileCVO.DAL.IDal<BLL.EvenementInschrijving>
{
    public int Insert(BLL.EvenementInschrijving inschrijvingEvenement)
    {
    }

    public List<BLL.EvenementInschrijving> SelectAllByCursistNummer(int cursistNummer)
    {
    }
}
```

Connectie maken

Vervolgens maken we een lijst aan om de les momenten bij te houden in de methode SelectAllByCursistNummer.

```
List<BLL.EvenementInschrijving> eventInschrijvingsLijst= new List<BLL.EvenementInschrijving>();
```

Nu kunnen we beginnen om de connectie met de databank te maken.

Eerst maken we een SqlConnection object aan waarvan we de als ConnectionString property de connection string meegeven die opgeslagen is in de web.config file.

```
SqlConnection connection = new SqlConnection();
connection.ConnectionString =
    System.Configuration.ConfigurationManager.
ConnectionStrings["MobileCVO"].ToString();
```

Hierna maken we een SqlCommand object aan die de nodige informatie bevat om de database aan te sturen zodat we de nodig informatie terug krijgen.

Aan dit command geven we mee welk soort type command we gebruiken en de nodige tekst. We gebruiken het type StoredProcedure en in dit voorbeeld de tekst is de naam van de stored procedure: grp2_SelectAllEvenementInschrijvingByCursistNummer of grp2_InsertEvenementInschrijving

```
SqlCommand command = new SqlCommand();
```

```

    command.CommandType = CommandType.StoredProcedure;
    command.CommandText = "grp2_SelectAllEvenementInschrijvingByCursistNummer";
Of voor te inserten
    command.CommandText = "grp2_InsertEvenementInschrijving";

```

Parameters meegeven

- SelectAllByCursistNummer(int cursistNummer)


```
// shortcut to add parameter
command.Parameters.Add(new SqlParameter("@CursistNummer",
    SqlDbType.Int)).Value = cursistNummer;
```
- Insert(BLL.EvenementInschrijving inschrijvingEvenement)


```
// shortcut to add parameter
command.Parameters.Add(new SqlParameter("@CursistNummer",
    SqlDbType.Int)).Value = inschrijvingEvenement.IdCursist;
command.Parameters.Add(new SqlParameter("@IdEvenement",
    SqlDbType.Int)).Value = inschrijvingEvenement.IdEvenement;
command.Parameters.Add(new SqlParameter("@Opmerkingen",
    SqlDbType.NVarChar, 255)).Value =
inschrijvingEvenement.Opmerkingen;
command.Parameters.Add(new SqlParameter("@ReservatieDatum",
    SqlDbType.DateTime)).Value = DateTime.Now;
SqlParameter id = new SqlParameter("@Id", SqlDbType.Int);
id.Direction = ParameterDirection.Output;
command.Parameters.Add(id);
```

Als laatste moeten we de connection, die we eerder hebben aangemaakt, toevoegen aan het command zodat deze de nodige connectie kan maken.

```
command.Connection = connection;
```

Data opvangen

Nu we de basis hebben om een connectie te maken, moeten we ervoor zorgen dat we de data die we terug krijgen, kunnen opvangen en omzetten in data die we in ons programma kunnen gebruiken.

Omdat er Sql fouten kunnen gebeuren werken we in een try catch.

We beginnen met een SqlDataReader waar we het resultaat van de database interactie zullen opvangen, en we openen de connectie.

We eindigen de try catch ook met een finally waarin we de connectie terug sluiten.

Hierdoor zijn we altijd zeker dat de connectie gesloten wordt (ook bij foutmeldingen) zodat we op een ander moment een nieuwe connectie kunnen openen.

Als de connectie geopend is kunnen we het command uitvoeren en de data die we terugkrijgen opslaan in de SqlDataReader die we eerder aan hebben gemaakt.

Vervolgens kijken we of we wel degelijk data hebben teruggekregen.

Nu kunnen we met de .Read() methode van de SqlDataReader rij per rij door de teruggekregen gegevens lezen en omzetten in data die we in het programma kunnen gebruiken

In het voorbeeld van de evenementeninschrijving lijst maken we telkens een EvenementInschrijvingobject aan.

In dit object kunnen we de properties toekennen door de waarde op te vragen met dezelfde naam als de waarden in de stored procedure.

We moeten deze data telkens converteren naar de juiste type van de properties.

try

```

{
    connection.Open();
    this.message = "De database is klaar!";

    using (result = command.ExecuteReader())
    {
        if (result.HasRows)
        {
            while (result.Read())
            {
                BLL.EvenementInschrijving eventInschrijving = new BLL.EvenementInschrijving();
                eventInschrijving.ReservatieDatum =
Convert.ToDateTime(result["Reservatiedatum"]);
                eventInschrijving.Naam = result["Naam"].ToString();
                eventInschrijving.Datum = Convert.ToDateTime(result["Datum"]);
                eventInschrijving.Locatie = result["Locatie"].ToString();
                eventInschrijving.StartUur = Convert.ToDateTime(result["StartUur"]);
                eventInschrijving.EindUur = Convert.ToDateTime(result["EindUur"]);
                eventInschrijving.Opmerkingen = result["Opmerkingen"].ToString();
                eventInschrijvingsLijst.Add(eventInschrijving);
            }
        }
    }
    catch (SqlException e)
    {
        this.message = e.Message;
    }
    finally
    {
        connection.Close();
    }
}

```

Oplossing:

```

public class EvenementInschrijving : MobileCVO.DAL.IDal< BLL.EvenementInschrijving>
{
    private string message;
    public string Message
    {
        get
        {
            return message;
        }
    }

    public EvenementInschrijving()
    {
        this.message = "";
    }

    public int Insert(BLL.EvenementInschrijving inschrijvingEvenement)
    {
        SqlConnection connection = new SqlConnection();

```

```

connection.ConnectionString =
    System.Configuration.ConfigurationManager.
    ConnectionStrings["MobileCVO"].ToString();
// SqlCommand object
SqlCommand command = new SqlCommand();
// in de CommandText eigenschap stoppen de naam
// van de stored procedure
string sqlString = "grp2_InsertEvenementInschrijving";
// shortcut to add parameter
command.Parameters.Add(new SqlParameter("@CursistNummer",
    SqlDbType.Int)).Value = inschrijvingEvenement.IdCursist;
command.Parameters.Add(new SqlParameter("@IdEvenement",
    SqlDbType.Int)).Value = inschrijvingEvenement.IdEvenement;
command.Parameters.Add(new SqlParameter("@Opmerkingen",
    SqlDbType.NVarChar, 255)).Value = inschrijvingEvenement.Opmerkingen;
command.Parameters.Add(new SqlParameter("@ReservatieDatum",
    SqlDbType.DateTime)).Value = DateTime.Now;
SqlParameter id = new SqlParameter("@Id", SqlDbType.Int);
id.Direction = ParameterDirection.Output;
command.Parameters.Add(id);
// zeg aan het command object dat het een tored procedure
// zal krijgen en geen SQL Statement
command.CommandType = CommandType.StoredProcedure;
// stop het sql statement in het command object
command.CommandText = sqlString;
// geeft het connection object door aan het command object
command.Connection = connection;
this.message = "Niets te melden";
// we gaan ervan uit dat het mislukt
int result = 0;
try
{
    connection.Open();
    // retourneert het aantal rijen dat geïnsered werd
    this.message = "De database is klaar!";
    result = command.ExecuteNonQuery();
    // we moeten kijken naar de waarde van out parameter
    // van Insert stored procedure. Als de naam van de
    // category al bestaat, retourneert de out parameter van
    // de stored procedure
    // -1
    if ((int)id.Value == -100)
    {
        this.message = String.Format("U bent reeds ingeschreven voor dit evenement.");
        result = -100;
    }
    else if (result <= 0)
    {
        this.message = String.Format("Uw inschrijving voor dit evenement is niet geregistreerd,
probeer het nog een keer.");
    }
    else
    {
        message = String.Format("Uw inschrijving voor dit evenement is succesvol afgerond!");
        result = (int)id.Value;
    }
}

```

```

        catch (SqlException e)
    {
        this.message = e.Message;
    }
    finally
    {
        connection.Close();
    }
    return result; // 0 of de Id van de nieuwe rij
}

public List<BLL.EvenementInschrijving> SelectAll()
{
    throw new NotImplementedException();
}

public List<BLL.EvenementInschrijving> SelectAllByCursistNummer(int cursistNummer)
{
    List<BLL.EvenementInschrijving> eventInschrijvingsLijst= new
List<BLL.EvenementInschrijving>();

    SqlConnection connection = new SqlConnection();
    connection.ConnectionString =
        System.Configuration.ConfigurationManager.
        ConnectionStrings["MobileCVO"].ToString();

    SqlCommand command = new SqlCommand();

    string sqlString = "grp2_SelectAllEvenementInschrijvingByCursistNummer";

    command.CommandType = CommandType.StoredProcedure;

    // shortcut to add parameter
    command.Parameters.Add(new SqlParameter("@CursistNummer",
        SqlDbType.Int)).Value = cursistNummer;
    command.CommandText = sqlString;

    command.Connection = connection;
    this.message = "Niets te melden";

    SqlDataReader result;
    try
    {
        connection.Open();
        this.message = "De database is klaar!";

        using (result = command.ExecuteReader())
        {
            if (result.HasRows)
            {
                while (result.Read())
                {
                    BLL.EvenementInschrijving eventInschrijving = new BLL.EvenementInschrijving();
                    eventInschrijving.ReservatieDatum =
                        Convert.ToDateTime(result["Reservatiedatum"]);
                    eventInschrijving.Naam = result["Naam"].ToString();
                }
            }
        }
    }
}

```

```

        eventInschrijving.Datum = Convert.ToDateTime(result["Datum"]);
        eventInschrijving.Locatie = result["Locatie"].ToString();
        eventInschrijving.StartUur = Convert.ToDateTime(result["StartUur"]);
        eventInschrijving.EindUur = Convert.ToDateTime(result["EindUur"]);
        eventInschrijving.Opmerkingen = result["Opmerkingen"].ToString();
        eventInschrijvingsLijst.Add(eventInschrijving);
    }

}

}

catch (SqlException e)
{
    this.message = e.Message;
}
finally
{
    connection.Close();
}

return eventInschrijvingsLijst;
}
}

```

Administratrix Select voorbeeld

maandag 1 juni 2015

16:12

Auteurs:

Nikos Vanden Broek
Mohamed Amajoutt

Probleem

Voor het grootste deel van het project moeten we data uit de database halen, die daarna getoond kan worden aan de gebruiker.

Hiervoor moeten we in de DAL verschillende select methode schrijven die de connectie met de database maken, en de data omzetten zodat we ze later eenvoudig kunnen tonen.

Als voorbeeld gebruiken we de methode om les momenten op te vragen.

Deze code moet aan de hand van een cursistnummer, alle lessen tussen een begin- en einddatum teruggeven.

Design

Om te beginnen moeten we ervoor zorgen dat de methode statisch is zodat we deze kunnen aanroepen zonder een object aan te maken. De methode moet een lijst van lesmomenten (BLL.LesDavinci) teruggeven, en heeft 3 parameters nodig: de cursistnummer, een begindatum en een einddatum.

```
public static List<BLL.LesDavinci> SelectAllByCursistNummerAndDates(int
cursistNummer, DateTime begin,
DateTime einde){ }
```

Connectie aanmaken

Vervolgens maken we een lijst aan om de les momenten bij te houden.

```
List<BLL.LesDavinci> lessenrooster = new List<BLL.LesDavinci>();
```

Nu kunnen we beginnen om de connectie met de databank te maken.

Eerst maken we een SqlConnection object aan waarvan we de als ConnectionString property de connection string meegeven die opgeslagen is in de Web.config file.

```
SqlConnection connection = new SqlConnection();
connection.ConnectionString =
System.Configuration.ConfigurationManager.
ConnectionStrings["MobileCVO"].ToString();
```

Hierna maken we een SqlCommand object aan die de nodige informatie bevat om de database aan te sturen zodat we de nodig informatie terug krijgen.

Aan dit command geven we mee welk soort type command we gebruiken en de nodige tekst. We gebruiken het type StoredProcedure en in dit voorbeeld de tekst is de naam van de stored procedure: grp2_SelectLestDavinci.

```
SqlCommand command = new SqlCommand();
command.CommandType = CommandType.StoredProcedure;
command.CommandText = "grp2_SelectLesDavinci";
```

Omdat de stored procedures parameters nodig heeft voor cursistnummer, begin datum en eind datum.

Dit kunnen we doen door nieuwe parameters de maken met de naam en databasetype. Deze toe te voegen aan de lijst van parameters, en er de nodige waarde aan te geven.

```
command.Parameters.Add(new SqlParameter("@CursistNummer", SqlDbType.Int)).Value =
cursistNummer;
command.Parameters.Add(new SqlParameter("@DateBegin", SqlDbType.DateTime)).Value =
begin;
command.Parameters.Add(new SqlParameter("@DateEinde", SqlDbType.DateTime)).Value
```

```
= einde;
```

Als laatste moeten we de connection, die we eerder hebben aangemaakt, toevoegen aan het command zodat deze de nodige connectie kan maken.

```
command.Connection = connection;
```

Data opvangen

Nu we de basis hebben om een connectie te maken, moeten we ervoor zorgen dat we de data die we terug krijgen, kunnen opvangen en omzetten in data die we in ons programma kunnen gebruiken.

Omdat er Sql fouten kunnen gebeuren werken we in een try catch.

We beginnen met een SqlDataReader waar we het resultaat van de database interactie zullen opvangen, en we openen de connectie.

We eindigen de try catch ook met een finally waarin we de connectie terug sluiten.

Hierdoor zijn we altijd zeker dat de connectie gesloten wordt (ook bij foutmeldingen) zodat we op een ander moment een nieuwe connectie kunnen openen.

```
try
{
    SqlDataReader result;
    connection.Open();
}
catch (SqlException e)
{
    throw e;
}
finally
{
    connection.Close();
}
```

Nu de connectie geopend is kunnen we het command uitvoeren en de data die we terugkrijgen opslaan in de SqlDataReader die we eerder aan hebben gemaakt.

Vervolgens kijken we of we wel degelijk data hebben teruggekregen.

```
using (result = command.ExecuteReader())
{
    if (result.HasRows)
    {
    }
}
```

Nu kunnen we met de .Read() methode van de SqlDataReader rij per rij door de teruggekregen gegevens lezen en omzetten in data die we in het programma kunnen gebruiken

In het voorbeeld van de lessen rooster maken we telkens een LesDavinci object aan.

In dit object kunnen we de properties toekennen door de waarde op te vragen met dezelfde naam als de waarden in de stored procedure.

We moeten deze data telkens converteren naar de juiste data van de properties.

En we voegen het les moment toe aan de lijst van les momenten die we eerder hebben aangemaakt

```
while (result.Read())
{
    BLL.LesDavinci les = new BLL.LesDavinci();
    les.Cursusnummer = result["Cursusnummer"].ToString();
    les.Datum = Convert.ToDateTime(result["Datum"]);
    les.Docent = result["Docent"].ToString();
    les.Campus = result["Campus"].ToString();
    les.Lokaal = result["Lokaal"].ToString();
    les.Module = result["Module"].ToString();
    les.Aanvangsdatum = Convert.ToDateTime(result["Van"]);
    les.Einddatum = Convert.ToDateTime(result["Tot"]);
    lessenrooster.Add(les);
}
```

Als laatste moeten we uiteraard de lijst van lesmomenten teruggeven.

```
        return lessenrooster;
```

Oplossing

```
public static List<BLL.LesDavinci> SelectAllByCursistNummerAndDates(int cursistNummer,
DateTime begin, DateTime einde)
{
    List<BLL.LesDavinci> lessenrooster = new List<BLL.LesDavinci>();

    SqlConnection connection = new SqlConnection();
    connection.ConnectionString =
        System.Configuration.ConfigurationManager.
        ConnectionStrings["MobileCV0"].ToString();

    SqlCommand command = new SqlCommand();

    command.CommandType = CommandType.StoredProcedure;
    command.CommandText = "grp2_SelectLesDavinci";

    command.Parameters.Add(new SqlParameter("@CursistNummer", SqlDbType.Int)).Value =
        cursistNummer;
    command.Parameters.Add(new SqlParameter("@DateBegin", SqlDbType.DateTime)).Value =
        begin;
    command.Parameters.Add(new SqlParameter("@DateEinde", SqlDbType.DateTime)).Value =
        einde;

    command.Connection = connection;

    try
    {
        SqlDataReader result;
        connection.Open();

        using (result = command.ExecuteReader())
        {
            if (result.HasRows)
            {
                while (result.Read())
                {
                    BLL.LesDavinci les = new BLL.LesDavinci();
                    les.Cursusnummer = result["Cursusnummer"].ToString();
                    les.Datum = Convert.ToDateTime(result["Datum"]);
                    les.Docent = result["Docent"].ToString();
                    les.Campus = result["Campus"].ToString();
                    les.Lokaal = result["Lokaal"].ToString();
                    les.Module = result["Module"].ToString();
                    les.Aanvangsdatum = Convert.ToDateTime(result["Van"]);
                    les.Einddatum = Convert.ToDateTime(result["Tot"]);
                    lessenrooster.Add(les);
                }
            }
        }
    catch (SqlException e)
    {
        throw e;
    }
    finally
    {
        connection.Close();
    }
}

return lessenrooster;
}
```

Auteurs:

Nikos Vanden Broek

Moodle MoodlePackage

dinsdag 26 mei 2015

14:45

Probleem

Om een communicatie te kunnen maken tussen het programma en Moodle is een klasse nodig die dit voor ons in orde brengt.

Deze klasse is origineel gemaakt door Abouchehatan Fouad.

Om niet opnieuw het wiel uit te vinden hebben we deze code overgenomen en is aangepast door Vanden Broek Nikos.

Design

De klasse bevat 3 velden:

Naam	Functie
MoodleURL	Een statische string die de basis URL bijhoud tussen alle objecten van de klasse zodat we deze maar 1 keer moeten meegeven
Service	Hier wordt de naam van de externe service in bewaard
Parameters	Een lijst van KeyValueCollection die toegevoegd moeten worden aan de basis URL.

En 3 methodes:

Naam	Functie
Constructor	Er wordt verplicht om een service naam mee te geven die bewaard wordt, en de lijst van KeyValueCollection wordt geïnitialiseerd
AddParameter	Voegt een KeyValueCollection toe aan de lijst
Send	Deze methode creëert de volledige URL uit de basis URL, stuurt een request naar de URL en geeft de Json reply terug als string

Oplossing

```
public class MoodlePackage
{
    public static string MoodleURL { set; get; }
    string service = "";
    List<KeyValuePair<string, string>> parameters;

    // Stel service in
    public MoodlePackage(String service)
    {
        this.service = service;
        parameters = new List<KeyValuePair<string, string>>();
    }

    // Voeg Input Parameters toe
    public void AddParameter(string key, string value)
    {
        parameters.Add(new KeyValuePair<string, string>(key, value));
    }

    /// <summary>
    /// Send package.
    /// </summary>
    /// <returns>JSON string.</returns>
    public String Send()
    {
        string output = "";
```

```

        string param_string = "";
        foreach (KeyValuePair<string, string> dx in parameters)
            param_string += dx.Key + "=" + dx.Value + "&";

        byte[] buffer = Encoding.ASCII.GetBytes(param_string);
        string lex = "http://" + MoodleURL + service + "?" + param_string;

        HttpWebRequest WebReq = (HttpWebRequest)WebRequest.Create(lex);
        WebReq.Method = WebRequestMethods.Http.Post;
        WebReq.ContentType = "application/x-www-form-urlencoded";

        WebReq.ContentLength = buffer.Length;
        using (Stream PostData = WebReq.GetRequestStream())
            PostData.Write(buffer, 0, buffer.Length);

        HttpWebResponse WebResp = (HttpWebResponse)WebReq.GetResponse();
        using (StreamReader reader = new StreamReader(WebResp.GetResponseStream()))
            output = reader.ReadToEnd();

        return output;
    }
}

```

Auteurs:

Nikos Vanden Broek

Moodle Token

dinsdag 26 mei 2015

16:13

Probleem

Om onze code toegang te geven om de webservices te kunnen gebruiken hebben we een token nodig.

Dit token kunnen we verkrijgen door een request naar Moodle te sturen met de inloggegeven van een specifieke gebruiker, en de naam van de service waar we toegang tot willen krijgen. (meer uitleg zie Webservices activeren)

Design

Eerst maken we een MoodleObject met en geven als service de locatie van de tokensevice mee.

```
MoodlePackage pToken = new Moodle.DAL.MoodlePackage("/login/token.php");
```

Vervolgens geven we de gebruikersnaam en wachtwoord van onze gebruiker mee voor de parameters username en password.

```
pToken.AddParameter("username", accountName);
pToken.AddParameter("password", password);
```

Als laatste geven we nog de naam van onze externe service mee voor de parameter service

```
pToken.AddParameter("service", service);
```

Nu ons MoodlePacket klaar is verzenden we het en vangen we de reply up on een JObject.

De reply bevat enkel de tokenstring dus kunnen we deze eenvoudig uit het object halen en terug geven met een return.

```
jToken = JObject.Parse(pToken.Send());
token = (string)jToken["token"];
return token;
```

Oplossing

```
public class Token
{
    /// <summary>
    /// Request a web service token from Moodle.
    /// </summary>
    /// <param name="accountName">Acount name of service account.</param>
    /// <param name="password">Password of service account.</param>
    /// <param name="service">Name of service.</param>
    /// <returns>Token string.</returns>
    public static string RequestTokenForService(string accountName, string password, string service)
    {
        string token = "";

        MoodlePackage pToken = new Moodle.DAL.MoodlePackage("/login/token.php");
        pToken.AddParameter("username", accountName);
        pToken.AddParameter("password", password);
        pToken.AddParameter("service", service);

        JObject jToken = new JObject();
        try
        {
            jToken = JObject.Parse(pToken.Send());
            token = (string)jToken["token"];
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
        return token;
    }
}
```

Auteurs:

Nikos Vanden Broek

Moodle Grade voorbeeld

dinsdag 26 mei 2015

15:27

Probleemstelling

Het doel van deze methode is om de scores van een cursus op te vragen van een specifieke opdracht.

Dit gebeurt aan de hand van een token, het Id van de cursist en het Id van de opdracht.

We maken gebruik van de package Json.NET. Meer uitleg hierover kan je vinden onder Webservices -> Json.NET

Design

MoodlePackage

We maken eerst een MoodlePackage object aan en geven de locatie van de service mee.

```
MoodlePackage pGrade = new MoodlePackage("/webservice/rest/server.php");
```

Als eerste parameter geven we het token mee voor de wstoken parameter.

```
pGrade.AddParameter("wstoken", token);
```

De volgende parameter is de naam van de functie die we willen gebruiken voor de parameter wsfunction, in dit geval gebruiken we mod_assing_get_grades.

```
pGrade.AddParameter("wsfunction", "mod_assing_get_grades");
```

De derde parameter is het formaat waaring we onze reply willen terugkrijgen, voor ons is dit json voor parameter moodlewsrestformat

```
pGrade.AddParameter("moodlewsrestformat", "json");
```

Als laatste moeten we de nodige parameters voor onze functie meegeven. Hier de Id van de opdracht waarvan we de score willen weten voor de parameter met de naam assignmentids[0].

```
pGrade.AddParameter("assignmentids[0]", "" + assignmentId);
```

Reply ontcijferen

De volgende stap is het package te verzenden en de JSON string die we terug krijgen te ontmantelen zodat we de nodige data kunnen gebruiken.

Omdat er errors kunnen ontstaan met de webservices werken we uiteraard in een try catch.

We maken een JObject aan waar we met de .Parse() methode de JSON string in steken.

Deze string bekomen we door de .Send() methode van de MoodlePackage te gebruiken.

De .Parse() methode zet de string om in een soort KeyValuePair die het eenvoudiger maar om de data die we nodig hebben op te vragen.

```
JObject jAssignments = new JObject();
jAssignments = JObject.Parse(pGrade.Send());
```

Om errors in de JSON strings op te vangen kijken we met de .HasValues propperty of we de nodige waarde terug krijgen.

Is dit niet zo dan maken gooien we een error met de .ToString() van het warnings deel.

We kunnen eenvoudig kijken of het de warnings waarden bevat omdat de JSON string geen warnings mee geeft als alles correct verloopt.

Hierdoor zou een nullpointer onstain.

```
if (!jAssignments["assignments"].HasValues)
{
    throw new Exception(jAssignments["warnings"].ToString());
}
```

Vervolgens kijken we de scores na van de opdracht.

De webservice functie geeft de scores van al de cursisten terug dus moeten we de cursistId van elke score vergelijken met de Id van de cursist die we nodig hebben.

Omdat de score die we terug krijgen foutief word geconvert naar een double moeten we zelf nog een omweg maken.

De teruggekregen score in string vorm is bv: 12.34000. Als we deze converteren naar een double wordt deze 1234000.

Dus om de correcte score te krijgen moeten we het getal eerst nog delen door 100.000.

We geven meteen de score terug zodat we niet blijven verder zoeken.

```
JObject jGrades = (JObject)jAssignments["assignments"][0];
foreach (JObject g in jGrades["grades"])
{
    if (Convert.ToInt32((string)g["userid"]) == studentId)
    {
        string score = (string)g["grade"];
        grade = Convert.ToDouble(score);
```

```

        grade /= 100000;
        return grade;
    }

}


```

Oplossing

```

public class Grade
{
    /// <summary>
    /// Get the grade of a student with id for an assignment with id.
    /// </summary>
    /// <param name="token">Moodle token.</param>
    /// <param name="assignmentId">Id of assignment.</param>
    /// <param name="studentId">Id of student.</param>
    /// <returns>Decimal grade for assignment.</returns>
    public static double GetGradeOfStudentForAssignment(string token, int studentId, int assignmentId)
    {
        double grade = -1;

        MoodlePackage pGrade = new MoodlePackage("/webservice/rest/server.php");
        pGrade.AddParameter("wstoken", token);
        pGrade.AddParameter("wsfunction", "mod_assign_get_grades");
        pGrade.AddParameter("moodlewsrestformat", "json");
        pGrade.AddParameter("assignmentids[0]", "" + assignmentId);

        JObject jAssignments = new JObject();

        try
        {
            jAssignments = JObject.Parse(pGrade.Send());

            if (!jAssignments["assignments"].HasValues)
            {
                throw new Exception(jAssignments["warnings"].ToString());
            }

            JObject jGrades = (JObject)jAssignments["assignments"][0];

            foreach (JObject g in jGrades["grades"])
            {
                if (Convert.ToInt32((string)g["userid"]) == studentId)
                {
                    string score = (string)g["grade"];
                    // Turn database string into double (12.34000 => 1234000)
                    grade = Convert.ToDouble(score);
                    // Devide to get actual grade (1234000 => 12.34)
                    grade /= 100000;

                    return grade;
                }
            }
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }

    return grade;
}

```

Auteurs:

Nikos Vanden Broek

Model Evenement

donderdag 4 juni 2015

3:46

Auteurs:

Mohamed Amajoutt

Probleem:

Het model bevat de datastructuren en methoden die we nodig hebben om de gegevens van de UI te prepareren of te verwerken. De model klasse is de scharnier tussen de UI en de gegevens van de organisatie. De organisatiegegevens bestaan uit business logica en dataopslag.

We hebben een lijst van evenementen moeten ophalen uit de database. Deze zal rij per rij worden ingelezen vanuit de database en ingelezen in Evenement uit de BLL klasse. Vervolgens moeten we al deze gegevens in lijst steken.

Design:

In de klasse Evenement van Model hebben we dus een EvenementSelectAll methode die een lijst retourneert van het object Evenement.

Methoden

Methode	Bereik	Parameter	Retour	Beschrijving
SelectAll	Public		Dal.SelectAll()	Neemt geen parameters en schiet de SelectAll methode van DAL.Evenement in gang.

Deze zal vervolgens een lijst terug geven en deze via de controller in de view tonen.

Oplossing:

```
public class Evenement
{
    public static List<BLL.Evenement> EvenementSelectAll()
    {
        Administratix.DAL.Evenement dal = new DAL.Evenement();
        return dal.SelectAll();
    }
}
```

Model EvenementInschrijving

zondag 7 juni 2015
0:10

Auteurs:
Mohamed Amajoult

Probleem:

Het model bevat de datastructuren en methoden die we nodig hebben om de gegevens van de UI te prepareren of te verwerken. De model klasse is de scharnier tussen de UI en de gegevens van de organisatie. De organisatiegegevens bestaan uit business logica en dataopslag.

We hebben een lijst van EvenementInschrijving moeten ophalen uit de database. Deze zal rij per rij worden ingelezen vanuit de database en ingelezen in EvenementInschrijving uit de BLL klasse.

We moeten ook een inschrijving voor een evenement kunnen doen adhv IdCursist, IdEvenement. De opmerkingen is optioneel om mee te geven.

Design:

In de klasse EvenementInschrijving van Model hebben we dus een EvenementInschrijvingSelectAllByCursistNummer methode die een lijst retourneert van het object EvenementInschrijving met als parameter een integer cursistnummer. We hebben ook de methode EvenementInschrijvingInsert die een boodschap teruggeeft vanuit de DAL. Deze methode geven we samen met de parameters idCursist, idEvenement, opmerkingen.

Methoden

Methode	Bereik	Parameter	Retour	Beschrijving
EvenementInschrijvingSelectAllByCursistNummer	Public	Int cursistNummer	Dal.SelectAllByCursistNummer(cursistNummer);	Neemt geen parameters en schiet de SelectAllByCursistNummer methode van DAL.EvenementInschrijving in gang.
EvenementInschrijvingInsert	Public	Int idCursist, int idEvenement, string opmerkingen	Dal.message;	Retourneert de boodschap op de insert is gelukt of mislukt.

Deze zal vervolgens een lijst terug geven en deze via de controller in de view tonen.

Oplossing:

```
public class EvenementInschrijving
{
    public static string EvenementInschrijvingInsert(int idCursist, string idEvenement, string opmerkingen)
    {
        Administratix.BLL.EvenementInschrijving evenementInschrijving = new BLL.EvenementInschrijving();
        evenementInschrijving.IdCursist = Convert.ToInt32(idCursist);
        evenementInschrijving.IdEvenement = Convert.ToInt32(idEvenement);
        evenementInschrijving.Opmerkingen = opmerkingen;
        Administratix.DAL.EvenementInschrijving dal = new DAL.EvenementInschrijving();
        dal.Insert(evenementInschrijving);
        return dal.Message;
    }

    public static List<Administratix.BLL.EvenementInschrijving> evenementInschrijvingSelectAllByCursistNummer(int cursistNummer)
    {
        Administratix.DAL.EvenementInschrijving dal = new DAL.EvenementInschrijving();
        return dal.SelectAllByCursistNummer(cursistNummer);
    }
}
```

View/Controller Evenement

donderdag 4 juni 2015
4:02

Auteurs:

Mohamed Amajoutt

Probleem:

De view houdt zich bezig met de voorstelling van informatie die getoond moet worden op de pagina. In dit geval is het een lijst van evenementen. Verder moet er nog niets gebeuren. De view moet ook 2 knoppen weergeven om de ingeschreven 2de zit van een cursist of alle evenementen te tonen.

Design:

View

Eerst hebben we in de View de knoppen die bovenaan weergegeven moeten worden. Deze zijn van het type "Submit" en hebben als name "Action" en value een zelfgekozen waarde. Deze 2 waardes moeten verschillend zijn omdat we de waarde van de "Action" gaan veranderen om de juiste handelingen te verrichten. Dit zal de controller moeten regelen.

```
<form id="evenementen-display" action="" method="post" class="containerBig containerOptions">
    <div class="containerOption">
        <div class="containerTitle"></div>
        <button type="submit" name="action" class="optionButton" value="ingeschreven-tonen">Ingeschreven</button></a>
        <button type="submit" name="action" class="optionButton" value="alleEvenementen-tonen">Alle</button></a>
    </div>
</form>
```

Controller

De controller zal nadien valideren wat de action bevat en de juiste View en data genereren door de methode SelectAll van de Model klasse aan te roepen. Deze zal vervolgens de lijst met evenementen afgaan en door middel van een for-lus elke rij afgaan. De informatie die uit elke rij komt wordt getoond voor de gebruiker en onder elkaar gezet.

```
if (action.Equals("ingeschreven-tonen"))
{
    evenementInschrijvingLijst =
Administratix.Model.EvenementInschrijving.evenementInschrijvingSelectAllByCursistNummer(Convert.ToInt32(cursist.CursistNummer));
}
```

In de view wordt er nog eens gevalideerd op de action en vervolgens het resultaat onder elkaar gezet.

```
@if (action.Equals("ingeschreven-tonen"))
{
    for (int i = 0; i < evenementInschrijvingLijst.Count; i++)
    {
        Administratix.BLL.EvenementInschrijving inschrijving = evenementInschrijvingLijst[i];
        <div class="itemBox textBox" id="@i">
            <a class="itemTitle" href="#">
                <div class="tableTitle1">
                    Datum: @inschrijving.Datum.ToString("dd-MM-yyyy", culture)
                </div>
                <div class="tableTitle2">
                    Tijdspip: @inschrijving.StartUur.ToString("HH:mm", culture) - @inschrijving.EindUur.ToString("HH:mm", culture)
                </div>
                <div class="tableTitle3">
                    Naam: @inschrijving.Naam
                </div>
            </a>
        </div>
    }
}
```

Oplossing:

```
@{
    Layout = "~/Shared/_Layout.cshtml";
    Page.Title = "Evenementen";

    Session["Page"] = "3";
    string action = "";
    Session["loginError"] = "";
    if (IsPost)
    {
```

```

        action = Request["action"];
        if (action != null)
        {
            switch (action)
            {
                case "login":
                    if (Session["cursist"] == null)
                    {
                        try
                        {

                            Session["cursist"] = Administratix.Model.Cursist.Login(Request["Login-CursistNummer"]);
                        }
                        catch (Exception e)
                        {

                            Session["loginError"] = e.Message;
                        }
                    }
                    break;

                case "logout":
                    Session["cursist"] = null;
                    break;
            }
        }
    }

    if (Session["cursist"] == null)
    {
        Response.Redirect("~/Deelpagina's/Default.cshtml");
    }

    //Dit is om nederlandse weekdagen/maanden te krijgen
    System.Globalization.CultureInfo culture = new System.Globalization.CultureInfo("nl-BE");

    string message = "Alles gaat goed";
    List<Administratix.BLL.Evenement> evenementenLijst = new List<Administratix.BLL.Evenement>();
    List<Administratix.BLL.EvenementInschrijving> evenementInschrijvingLijst = new
    List<Administratix.BLL.EvenementInschrijving>();

    evenementenLijst = Administratix.Model.Evenement.EvenementSelectAll();

    if (Session["cursist"] != null)
    {
        Administratix.BLL.Cursist cursist = (Administratix.BLL.Cursist)Session["cursist"];

        evenementenLijst = Administratix.Model.Evenement.EvenementSelectAll();

        if (action != null)
        {
            if (action.Equals("inschrijven"))
            {
                message =
Administratix.Model.EvenementInschrijving.EvenementInschrijvingInsert(Convert.ToInt32(cursist.CursistNummer),
Request["evenementId-inschrijving"], Request["opmerkingen-inschrijving"]);
            }
            if (action.Equals("ingeschreven-tonen"))
            {
                evenementInschrijvingLijst =
Administratix.Model.EvenementInschrijving.evenementInschrijvingSelectAllByCursistNummer(Convert.ToInt32(cursist.CursistNummer));
            }
        }
    }
}

<h1>Evenementen</h1>
<form id="evenementen-display" action="" method="post" class="containerBig containerOptions">
    <div class="containerOption">
        <div class="containerTitle"></div>
        <button type="submit" name="action" class="optionButton" value="ingeschreven-tonen">Ingeschreven</button></a>
        <button type="submit" name="action" class="optionButton" value="alleEvenementen-tonen">Alle</button></a>
    </div>
</form>

@if (action.Equals("ingeschreven-tonen"))
{
    for (int i = 0; i < evenementInschrijvingLijst.Count; i++)

```

```

    {
        Administratix.BLL.EvenementInschrijving inschrijving = evenementInschrijvingLijst[i];
        <div class="itemBox tableBox" id="@i">
            <a class="itemTitle" href="#@i">
                <div class="tableTitle1">
                    Datum: @inschrijving.Datum.ToString("dd-MM-yyyy", culture)
                </div>
                <div class="tableTitle2">
                    Tijdstip: @inschrijving.StartUur.ToString("HH:mm", culture) - @inschrijving.EindUur.ToString("HH:mm", culture)
                </div>
                <div class="tableTitle3">
                    Naam: @inschrijving.Naam
                </div>
            </a>
            <div class="itemContent table">
                <div class="itemContentText table">
                    Reservatiедatum: @inschrijving.ReservatieDatum.ToString("dd-MM-yyyy", culture)
                </div>
                <div class="itemContentText table">
                    Locatie: @inschrijving.Locatie
                </div>
                <div class="itemContentText table">
                    Opmerkingen: @inschrijving.Opmerkingen
                </div>
            </div>
        </div>
    }
}

else
{
    for (int i = 0; i < evenementenLijst.Count; i++)
    {
        Administratix.BLL.Evenement evenement = evenementenLijst[i];
        <div class="itemBox tableBox" id="@i">
            <a class="itemTitle" href="#@i">
                <div class="tableTitle1">
                    Datum: @evenement.Datum.ToString("dd-MM-yyyy", culture)
                </div>
                <div class="tableTitle2">
                    Tijdstip: @evenement.Start.ToString("HH:mm", culture) - @evenement.Eind.ToString("HH:mm", culture)
                </div>
                <div class="tableTitle3">
                    Naam: @evenement.Naam
                </div>
            </a>
            <div class="itemContent table">
                <div class="itemContentText table">
                    Locatie: @evenement.Locatie
                </div>
                @if (Session["cursist"] != null)
                {
                    <form id="evenementen-inschrijven" action="" method="post">
                        <div class="itemContentText table">
                            <input type="text" name="opmerkingen-inschrijving" id="opmerkingen-inschrijving" />
                            <input type="hidden" name="evenementId-inschrijving" id="evenementId-inschrijving" value="@evenement.Id" />
                        </div>
                        <div class="itemContentText table">
                            <button type="submit" name="action" value="inschrijven">Schrijf me in!</button>
                        </div>
                    </form>
                }
            </div>
        </div>
    }
}

```

@message

Auteurs:

Deadlines

donderdag 4 juni 2015
12:22

Probleem

In het project willen we een pagina waar de gebruiker zijn of haar deadlines kan bekijken.
Deze deadlines zijn de datums van taken op Moodle waarop de taak ingediend moet worden.

Design

Coordinating controller

Om te beginnen moeten we voor MoodlePackage de static property voor de moodle URL instellen zodat deze niet elke keer opnieuw meegegeven moet worden.

```
Moodle.DAL.MoodlePackage.MoodleURL = "moodle-cvomobile.rhcloud.com";
```

Vervolgens maken we een lijst van Course objecten om bij te houden voor welke cursussen de gebruiken is ingeschreven, en een lijst van Deadline objecten om bij te houden welke deadlines de gebruiker heeft in een komende tijdsperiode.

```
List<Moodle.BLL.Course> _enrolledCourses = new List<Moodle.BLL.Course>();  
List<Moodle.BLL.Deadline> _upcommingDeadlines = new List<Moodle.BLL.Deadline>();
```

Nu kunnen we een tokenstring vragen aan Moodle die we nodig hebben om data op te vragen.

```
string token = Moodle.DAL.Token.RequestTokenForService("cvomobile", "Boerderijm1#s", "mobile");
```

Om te weten te komen welke taken van toepassing zijn om de gebruiker moeten we weten wat de cursist id van de gebruiker is op Moodle.
Dit doen we door eerst het ingelogde cursist object op te vragen uit de sessie.

Hiermee kunnen we dan, met behulp van het token en het e-mail adres van de gebruiker, de Id opvragen.

```
Administratix.BLL.Cursist cursist = (Administratix.BLL.Cursist)Session["cursist"];  
int cursistId = Moodle.DAL.User.GetUserByIdByEmail(token, cursist.Email);
```

Nu we de Id van de gebruiker weten kunnen we van moodle al de cursussen opvragen waar de gebruiker toe behoort.
Dit laten we enkel toe als de Id van de gebruiker niet -1 is, wat betekend dat de gebruiker niet gevonden is op Moodle.

```
if (cursistId != -1)  
{  
    _enrolledCourses = Moodle.DAL.Course.GetUserEnrolledCourses(token, cursistId);  
}
```

Hierna willen we van elke cursus al de bestaande taken opvragen.

```
foreach (Moodle.BLL.Course _course in _enrolledCourses)  
{  
    _course.Assignments = Moodle.DAL.Assignment.GetAllAssingmentInCourse(token, _course.Id);  
}
```

Als laatste vragen we ook eventuele scores op van de taken die van toepassing zijn op de gebruiker.

```
foreach (Moodle.BLL.Assignment _assing in _course.Assignments)  
{  
    _assing.CourseName = _course.FullName;  
    double score = Moodle.Model.GetGradeOfStudentForAssignment(token, _assing.Id, cursistId);  
    _assing.UserScore = "" + score;  
}
```

Nu kunnen we met de data de we hebben gekregen ook een lijst maken van al de taken die de gebruiker heeft in een komende tijdsspanne.
Ook sorteren we al de cursussen op naam voor een duidelijker overzicht.

```
_upcommingDeadlines = Moodle.Model.GetDeadlinesInTimespan(_enrolledCourses, DateTime.Now, 14);  
_enrolledCourses = _enrolledCourses.OrderBy(o => o.FullName).ToList();
```

View Controller

Als eerste gaan we na of er een fout is gebeurd bij het opvragen van de cursist Id.

Is dit het geval kijken we of de gebruiker een e-mail adres mist, of dat de gebruiker niet bestaat in Moodle

```
if (@cursistId == -1)  
{  
    <div>  
        @if (cursist.Email.Equals(""))  
        {  
            @:Er is geen e-mail adres gevonden voor deze cursist. Gelieve contact op te nemen met de administratie.  
        }  
        else  
        {  
            @:Geen cursist gevonden met het volgende email adres: @cursist.Email  
        }  
    </div>  
}
```

Als de Id in orde is laten we een lijst zien van al de deadlines in de komende 2 weken.

Indien de lijst van komende deadlines leeg is laten we de gebruiker weten dat er geen zijn.

```
<div class="containerBig">  
    <div class="containerTitle">
```

```

        Komende 2 weken
    </div>
    @if (_upcommingDeadlines.Count == 0)
    {
        <div class="itemContentText">
            Je hebt momenteel geen deadlines in de komende 2 weken.
        </div>
    }
</div>

```

Zijn er wel deadlines dan laten we per datum zien welke taak en voor welke cursus er zijn.

```

else
{
    foreach (Moodle.BLL.Deadline deadline in _upcommingDeadlines)
    {
        <div class="moodleAssignment">
            <div class="itemContentTitle">
                @String.Format("{0:d/M/yyyy}", deadline.Date.Date)
            </div>
            @foreach (Moodle.BLL.Assignment assignment in deadline.Assignments)
            {
                <div class="moodleGrade">
                    <b>@assignment.CourseName</b> : @assignment.Name<br />
                </div>
            }
        </div>
    }
}

```

Hierna laten we ook elke cursus zien aan de gebruiker.

Als de cursus taken bevat laten we ook een ▼ icoon zien zodat de gebruiker weet dat hij/zij deze kan openen voor meer informatie.

```

<div class="itemContainer">
    @for (int i = 0; i < _enrolledCourses.Count; i++)
    {
        Moodle.BLL.Course course = _enrolledCourses[i];
        <div class="itemBox" id="@course.Id">
            <a class="itemTitle" href="#@course.Id">
                @course.FullName
                @if (course.Assignments.Count != 0)
                {
                    <div class="iconExpand">▼</div>
                }
            </a>
        </div>
    }
</div>

```

Voor elke cursus laten we de namen van de taken zien.

Als de gebruiker al een score heeft gekregen wordt deze getoond.

Indien dit niet het geval is wordt de datum van de deadline getoond als deze van toepassing is, anders wordt NVT getoond.

```

<div class="itemContent">
    @for (int j = 0; j < course.Assignments.Count; j++)
    {
        Moodle.BLL.Assignment assign = course.Assignments[j];
        <div class="itemContentTitle">
            @assign.Name
        </div>
        <div class="itemContentText">
            @if (assign.UserScore.Equals("") + -1)
            {
                @:Deadline:
                @if (assign.DueDate.Date == new DateTime(1970, 1, 1).Date)
                {
                    @:NVT
                }
                else
                {
                    @String.Format("{0:d/M/yyyy HH:mm}", assign.DueDate);
                }
            }
            else
            {
                @:Score: @assign.UserScore / @assign.MaxGrade
            }
        </div>
    }
</div>

```

Oplossing

```

@{
    Layout = "~/Shared/_Layout.cshtml";
}

```

```

Page.Title = "Deadlines";
string action = "";
Session["Page"] = "2";
Session["loginError"] = "";
if (IsPost)
{
    action = Request["action"];
    switch (action)
    {
        case "login":
            if (Session["cursist"] == null)
            {
                try
                {
                    Session["cursist"] = Administratix.Model.Cursist.Login(Request["Login-CursistNummer"]);
                }
                catch (Exception e)
                {
                    Session["loginError"] = e.Message;
                }
            }
            break;

        case "logout":
            Session["cursist"] = null;
            break;
    }
}
if (Session["cursist"] == null)
{
    Response.Redirect("~/Deelpagina's/Default.cshtml");
}

Moodle.DAL.MoodlePackage.MoodleURL = "moodle-cvomobile.rhcloud.com";
List<Moodle.BLL.Course> _enrolledCourses = new List<Moodle.BLL.Course>();
List<Moodle.BLL.Deadline> _upcommingDeadlines = new List<Moodle.BLL.Deadline>();

string token = Moodle.DAL.Token.RequestTokenForService("cvomobile", "Boerderijm1n#s", "mobile");
Administratix.BLL.Cursist cursist = (Administratix.BLL.Cursist)Session["cursist"];
int cursistId = Moodle.DAL.User.GetUserIdByEmail(token, cursist.Email);

if (cursistId != -1)
{
    // Get all courses the student is enrolled in
    _enrolledCourses = Moodle.DAL.Course.GetUserEnrolledCourses(token, cursistId);

    foreach (Moodle.BLL.Course _course in _enrolledCourses)
    {
        // Get all assignments of a course
        _course.Assignments = Moodle.DAL.Assignment.GetAllAssingmentInCourse(token, _course.Id);

        foreach (Moodle.BLL.Assignment _assing in _course.Assignments)
        {
            // Add name of course for dealines
            _assing.CourseName = _course.FullName;

            // Get the score of the student for the assignment
            double score = Moodle.Model.GetGradeOfStudentForAssignment(token, _assing.Id, cursistId);
            _assing.UserScore = "" + score;
        }
    }

    _upcommingDeadlines = Moodle.Model.GetDeadlinesInTimespan(_enrolledCourses, DateTime.Now, 14);
    _enrolledCourses = _enrolledCourses.OrderBy(o => o.FullName).ToList();
}

if (@cursistId == -1)

```

```

{
    <div>
        @if (cursist.Email.Equals(""))
        {
            #:Er is geen e-mail adres gevonden voor deze cursist. Gelieve contact op te nemen met de administratie.
        }
        else
        {
            #:Geen cursist gevonden met het volgende email adres: @cursist.Email
        }
    </div>
}
else
{
    <div class="containerBig">
        <div class="containerTitle">
            Komende 2 weken
        </div>

        @if (_upcommingDeadlines.Count == 0)
        {
            <div class="itemContentText">
                Je hebt momenteel geen deadlines in de komende 2 weken.
            </div>
        }
        else
        {

            foreach (Moodle.BLL.Deadline deadline in _upcommingDeadlines)
            {
                <div class="moodleAssignment">
                    <div class="itemContentTitle">
                        @String.Format("{0:d/M/yyyy}", deadline.Date.Date)
                    </div>
                    @foreach (Moodle.BLL.Assignment assignment in deadline.Assignments)
                    {
                        <div class="moodleGrade">
                            <b>@assignment.CourseName</b> : @assignment.Name<br />
                        </div>
                    }
                </div>
            }
        }
    </div>

    <div class="itemContainer">
        @for (int i = 0; i < _enrolledCourses.Count; i++)
        {
            Moodle.BLL.Course course = _enrolledCourses[i];

            <div class="itemBox" id="@course.Id">
                <a class="itemTitle" href="#@course.Id">
                    @course.FullName
                    @if (course.Assignments.Count != 0)
                    {
                        <div class="iconExpand">▼</div>
                    }
                </a>

                <div class="itemContent">

                    @for (int j = 0; j < course.Assignments.Count; j++)
                    {
                        Moodle.BLL.Assignment assign = course.Assignments[j];
                        <div class="itemContentTitle">
                            @assign.Name
                        </div>
                        <div class="itemContentText">
                            @if (assign.UserScore.Equals("") + -1)
                            {
                                #:Deadline: @if (assign.DueDate.Date == new DateTime(1970, 1, 1).Date)
                                {
                                    #:NVT
                                }
                                else
                                {
                                    @String.Format("{0:d/M/yyyy HH:mm}", assign.DueDate);
                                }
                            }

                            @else
                            {
                                #:Score: @assign.UserScore / @assign.MaxGrade
                            }
                        </div>
                    }
                </div>
            </div>
        }
    </div>
}

```

```
        </div>
    }
</div>

}

}
```

Auteurs:

Nikos Vanden Broek

Login systeem

donderdag 4 juni 2015

12:31

Probleem

Voor het project hebben we een systeem nodig waar de gebruiker 1 maal zijn of haar cursist nummer moet ingeven zodat we deze kunnen gebruiken op de verschillende pagina's. Omdat de school zelf aan een nieuw systeem werkt houden we het eenvoudig waar enkel de cursist nummer nodig is en geen wachtwoord.

Design

Om data te bewaren tussen pagina's maken we gebruik van Session waar we met verschillende namen, data in kunnen opslaan.

Deze sessie wordt per default bijgehouden voor 20 minuten.

Voor het project gebruiken we een session cursist om de gegevens van de gebruikt bij te houden, en een session loginError waar we foutmeldingen bijhouden bij het inloggen.

Op het begin van elke pagina maken we de loginError terug leeg.

```
Session["loginError"] = "";
```

Vervolgens kijken we het laden van de pagina gebeurt via Post.

Als dit het geval is vragen we de value van de action afhankelijk waarvan we weten of de gebruiker in of uit logt.

```
if (IsPost)
{
    action = Request["action"];
    switch (action)
    {
        case "login":
            break;
        case "logout":
            break;
    }
}
```

Indien de gebruiker in logt gebruiken we en try catch omdat de methode om in te loggen een exception kan geven.

We gebruiken de login methode waar we de ingegeven cursist nummer aan mee geven.

Het Cursist object dat we terug krijgen slaan we op in de cursist session.

Als we een exception is wordt deze opgevangen en slaan we de error message op in de loginError session.

```
case "login":
    if (Session["cursist"] == null)
    {
        try
        {
            Session["cursist"] =
                Administratix.Model.Cursist.Login(Request["Login-
                CursistNummer"]);
        }
        catch (Exception e)
        {
            Session["loginError"] = e.Message;
        }
    }
    break;
```

Indien de gebruiker uit logt zetter we simpelweg de cursist session op null.

```
case "logout":
    Session["cursist"] = null;
```

```
break;
```

Na de IsPost kijken we altijd of de cursist session null is.

Indien dit zo is, redirecten we de gebruiker naar de default pagina, die op zich redirect naar de homepage waar de gebruiker opnieuw gevraagd wordt in te loggen.

```
if (Session["cursist"] == null)
{
    Response.Redirect("~/Deelpagina's/Default.cshtml");
```

Oplossing

```
Session["loginError"] = "";
if (IsPost)
{
    action = Request["action"];
    switch (action)
    {
        case "login":
            if (Session["cursist"] == null)
            {
                try
                {
                    Session["cursist"] =
                        Administratix.Model.Cursist.Login(Request["Login-
CursistNummer"]);
                }
                catch (Exception e)
                {
                    Session["loginError"] = e.Message;
                }
            }
            break;
        case "logout":
            Session["cursist"] = null;
            break;
    }
}
if (Session["cursist"] == null)
{
    Response.Redirect("~/Deelpagina's/Default.cshtml");
}
```

Auteurs:

Nikos Vanden Broek

CSS

donderdag 4 juni 2015

3:52

Probleem

Om ervoor te zorgen dat onze pagina's niet simpelweg zwarte tekst op een witte achtergrond is maken we gebruik van een Cascading Style Sheet (CSS).

Hiermee kunnen we aan de hand van properties de elementen schikken en veranderen van uiterlijk zoals we dat willen.

Design

CSS maakt gebruik van een **selector** die, zoals bij c#, **properties** heeft waar we **values** aan kunnen geven.

De selector kan de naam zijn van een element, een class of een id zijn.

Voor elementen gebruik je gewoon hun type.

Voor een class gebruik je een punt (.) gevuld door de naam.

Voor een id gebruik je een hekje (#) gevuld door de naam.

```
body {  
    background: #DEDEDE url('../Images/bg01.png') no-repeat fixed center 300px;  
    color: #676767;  
}
```

Het is mogelijk om meerdere selectoren te gebruiken als en dezelfde properties aan te passen door ze te onderscheiden met een komma (,).

```
table, td {  
    padding: 5px;  
    vertical-align: top;  
    border-collapse: collapse;  
}
```

Om enkel elementen te selecteren met een specifieke parent element gebruik je het groter dan (>) teken.

In dit voorbeeld selecteren we elke div waarvan de parent een class .itemTitle heeft, die op zich de parent .roosterBox heeft.

```
.roosterBox > .itemTitle > div {  
    display: inline-block;  
}
```

Je kan ook elementen selecteren afhankelijk van verschillende zaken zoals:

:link	Een link die nog niet bezocht is
:visited	Een link die eerder bezocht is
:hover	Een element waar de gebruiker de muis over houdt
:target	Een element geselecteerd aan de hand van een anchor (#) in de URL

```
a:link,a:visited,a:hover,a:active {  
    text-decoration: none;  
}
```

Auteurs:

Nikos Vanden Broek

Flexbox

donderdag 4 juni 2015

12:27

Probleem

Omdat het project op zowel desktop als mobile bruikbaar moet zijn, moet de lay-out van de pagina's zich kunnen aanpassen aan de verschillende schermresoluties.

Een eenvoudige en effectieve oplossing hiervoor is het gebruik van Flexbox.

Als voorbeeld gebruiken we de trajectoverzicht pagina.

Design

Om de verschillende objecten te ordenen moeten ze allemaal in een gezamenlijke container zitten zoals een div.

```
<div class="container">
  <div class="item">
    Multimedia
  </div>
  <div class="item">
    Netwerkbeheer
  </div>
  <div class="item">
    Programmeren
  </div>
</div>
```

Nu moeten we het display type van de container veranderen naar flex zodat de elementen verplaatst kunnen worden.

```
.container{
  display: flex;
}
```

Nu zorgen we ervoor dat de grootte van de items bepaald wordt.

In het voorbeeld hebben we op de desktop lay-out 2 items naast elkaar te zien zijn.

Dit kunnen we doen door de breedte van de items in te stellen op bv: 45%.

Omdat we op de mobiele lay-out alle items onder elkaar willen geven we de items ook een minimum breedte.

```
.item{
  width: 45%;
  min-width: 250px;
}
```

Om ervoor te zorgen al de items in rijen onder elkaar kunnen staan moeten we de flex container de flex-wrap property op wrap zetten.

Dit zorgt ervoor dat, als er meerdere elementen zijn dan er naast elkaar kunnen staan, ze in rijen onder elkaar verdeeld worden.

```
.container{
  display: flex;
  flex-wrap: wrap;
}
```

Omdat de totale grootte van de 2 items naast elkaar kleiner is dan de totale grote (90%) moeten we de items nog centreren.

Dit kan door het justify-content property van de container in te stellen op center, space-between of space-around in te stellen.

- Center zal items in het midden plaatsen met ruimte van de kanten (5% links en 5% rechts).
- Space-between zal de items verspreiden met gelijke ruimtes tussen de items, maar geen ruimte van de kanten (10% tussen de items)
- Space-around zal ook gelijke ruimtes tussen de items plaatsen maar houd ook ruimte van de kanten (2.5% links, 5% tussen de items en 2.5% rechts)

In ons voorbeeld gebruiken we Space-around.

```
.container{
```

```
display: flex;  
flex-wrap :wrap;  
justify-content:space-around;  
}
```

Oplossing

Desktop

☒ Multimedia 1 TV

☒ Netwerkbeheer 2 TV ▼

nodige voorkennis

☒ Datacommunicatie en netwerken TV

☒ Netwerkbeheer 1 TV

☒ Netwerkbeheer 1 TV

☒ Programmeren 1 TV

☒ Programmeren 2 TV

☒ Programmeren 3 TV

Mobile

☒ Multimedia 1 TV

☒ Netwerkbeheer 1 TV ▼

☒ Netwerkbeheer 2 TV ▼

nodige voorkennis

☒ Datacommunicatie en netwerken
TV

☒ Netwerkbeheer 1 TV

☒ Programmeren 1 TV

☒ Programmeren 2 TV

☒ Programmeren 3 TV ▼

Auteurs:

Nikos Vanden Broek

Masterpage lay-out

donderdag 4 juni 2015

12:29

Probleem

Om alle pagina's dezelfde layout te geven was er een masterpage nodig, De masterpage dicteert de layout van alle pagina's die het implementeren.

Design

Om de masterpage aan te maken moest er eerst een basis worden ingevoegd die alles dicteert, hiervoor werd de _Layout pagina gemaakt, de '_' zorgt er ook voor dat de pagina wordt genegeerd en kan dus niet bereikt worden in een webbrowser.

Om de navigatie , header en footer te scheiden van de layout werden aparte pagina's geschreven, om die pagina's te implementeren moest dit stukje in de layout pagina gestoken worden.

```
@RenderPage("~/Shared/_Nav.cshtml")
```

Deze code zorgt ervoor dat de data van '_nav' wordt ingeladen bij het implementeren van de masterpage, in '_nav' wordt ook code gezet die de navigatie bestuurd, die is hierdoor gescheiden van de rest van het programma.

In het '<head>' gedeelte van de layout wordt de CSS aangeroepen die wordt gebruikt op alle pagina's, deze bepaalt hoe ze er uit gaan zien, en zorgt voor een uniforme look van de site

```
<link href "~/Styles/Site.css" rel="stylesheet" type="text/css" />
```

Ook in de head vinden we de code '<title>@Page.Title</title>', dit stukje code zorgt ervoor dat de titel van de pagina's kan worden bepaald.

De plaats waar al de gegevens komen wordt bepaalt door de tag @RenderBody(), deze bepaalt waar alle HTML code van de pagina's komen die de Masterpage implementeren, hierdoor voorkom je dat de header, footer of navigatie nog worden aangepast

Om de pagina op te roepen moet je maar twee lijntjes code schrijven:

- Layout = "~/Shared/_Layout.cshtml";
- Page.Title = "Evenementen";

De layout tag zorgt ervoor dat de masterpage wordt ingeladen met al zijn componenten.

De page.Title tag zorgt ervoor dat je de titel van de pagina kan veranderen

Oplossing

```
@{  
    Layout = null;  
}  
  
<!DOCTYPE html>  
  
<html>  
<head>  
    <meta name="viewport" content="width=device-width" />  
    <title>@Page.Title</title>  
    <link href "~/Styles/Site.css" rel="stylesheet" type="text/css" />  
</head>  
<body>  
    <main>  
        <div id="centerContainer">  
            <div id="contentContainer">  
                @RenderPage("~/Shared/_Nav.cshtml")  
                <div id="watermark">  
                </div>
```

```
        @RenderBody()
    </div>
</main>
</body>
</html>
```

Auteurs:

Jo Ronsse
Sonja Van Rompaey

Navigatie

donderdag 4 juni 2015

12:31

Auteur: Jo Ronsse

Probleem

Om een aangename ervaring te maken voor een android gsm hebben we de navigatie speciaal aangepast.

Design

We beginnen met een lijst te maken van alle pagina's , hier kan makkelijk worden aan toegevoegd en dit word gebruikt om de volgende en vorige pagina's af te halen.

Deze is geplaatst in DAL.

```
public class paginaTest
{
    public static List<string> Makelist()
    {
        List<string> test = new List<string>();
        test.Add("../Default.cshtml");
        test.Add("/Deelpagina's/Kalender.cshtml");
        test.Add("/Deelpagina's/Deadlines.cshtml");
        test.Add("/Deelpagina's/Evenementen.cshtml");
        test.Add("/Deelpagina's/ExamenData.cshtml");
        test.Add("/Deelpagina's/PuntenOverzicht.cshtml");
        test.Add("/Deelpagina's/Lessenrooster.cshtml");
        test.Add("/Deelpagina's/TrajectOverzicht.cshtml");
        return test;
    }
}
```

Om deze data af te halen staat er in de 'Model' 2 aparte aanroep methodes , 1 voor de vorige pagina en een voor de volgende.

```
public class pagina
{
    public static string volgende(int i)
    {
        try
        {
            List<string> pages = DAL.paginaTest.Makelist();
            return pages[i + 1];
        }
        catch
        {
            return "Error";
        }
    }
}
```

Zoals u ziet is het omgeven met een 'try catch' om te voorkomen dat er een 'out of bounds' gebeurt, dit zou normaal in de '_Nav' moeten zijn opgevangen maar is een extra veiligheid.

In de pagina _Nav die word aangeroepen in de Masterpage checkt de website of de gebruiker is ingelogd.

```
@if (Session["cursist"] != null)
```

Dit voorkomt dat gebruikers naar andere pagina's kunnen gaan zonder in te loggen, omdat elke pagina, behalve de 'default' page, de data van "cursist" nodig heeft.

Is deze voorwaarde voldaan word het volgende stuk code aangeroepen.

```
<a href="/"></a>
    if (Session["Page"] != "0")
    {
```

```

        <a
    href="@Administratix.Model.pagina.vorige(Convert.ToInt32(Session["Page"]))"><img
    src "~/Images/nav_left.png" alt="Nav Links" /></a>
    }
    if (Session["Page"] != "7")
    {
        <a
    href="@Administratix.Model.pagina.volgende(Convert.ToInt32(Session["Page"]))"><img
    src "~/Images/nav_right.png" alt="Nav Rechts" /></a>
    }

```

Het eerste logo is een terug knop, dit is om terug naar het menu met alle pagina's te gaan.

Het volgende logo brengt je terug naar de vorige pagina, als deze pagina de 'Default' pagina is dan word deze niet weer gegeven.

```

if (Session["Page"] != "0")
{
    <a
    href="@Administratix.Model.pagina.vorige(Convert.ToInt32(Session["Page"]))"><img
    src "~/Images/nav_left.png" alt="Nav Links" /></a>
}

```

Het laatste logo brengt je naar de volgende pagina in de lijst, dit logo word niet weergegeven als je op de laatste pagina bent.

```

if (Session["Page"] != "7")
{
    <a
    href="@Administratix.Model.pagina.volgende(Convert.ToInt32(Session["Page"]))"><img
    src "~/Images/nav_right.png" alt="Nav Rechts" /></a>
}

```

Als er pagina's bijkomen moet het getal "7" met het nieuwe nummer worden aangepast, om te zorgen dat de volgende pagina's kunnen worden bereikt.

De navigatie ziet er als het volgt uit als alle voorwaarden zijn voldaan: (ingelogd en niet op de eerste of laatste pagina)



Oplossing

DAL

```

public class paginaTest
{
    public static List<string> Makelist()
    {
        List<string> test = new List<string>();
        test.Add("../Default.cshtml");
        test.Add("/Deelpagina's/Kalender.cshtml");
        test.Add("/Deelpagina's/Deadlines.cshtml");
        test.Add("/Deelpagina's/Evenementen.cshtml");
        test.Add("/Deelpagina's/ExamenData.cshtml");
        test.Add("/Deelpagina's/PuntenOverzicht.cshtml");
        test.Add("/Deelpagina's/Lessenrooster.cshtml");
        test.Add("/Deelpagina's/TrajectOverzicht.cshtml");
        return test;
    }
}

```

Model

```

public class pagina
{

```

```

public static string volgende(int i)
{
    try
    {
        List<string> pages = DAL.paginaTest.Makelist();
        return pages[i + 1];
    }
    catch
    {
        return "Error";
    }
}
public static string vorige(int i)
{
    try
    {
        List<string> pages = DAL.paginaTest.Makelist();
        return pages[i - 1];
    }
    catch
    {
        return "Error";
    }
}
}

_Nav
<div class="navLogo">
    <img src("~/Images/Logo_mobileCVO.jpg" alt="Logo MobileCVO" />
    @if (Session["cursist"] != null)
    {
        <a href="/"><img src "~/Images/1431366103_text_align_full-128.png" alt="Logo Sandwich" /></a>
        if (Session["Page"] != "0")
        {
            <a
                href="@Administratix.Model.pagina.vorige(Convert.ToInt32(Session["Page"]))"><img
                src "~/Images/nav_left.png" alt="Nav Links" /></a>
        }
        if (Session["Page"] != "7")
        {
            <a
                href="@Administratix.Model.pagina.volgende(Convert.ToInt32(Session["Page"]))"><img
                src "~/Images/nav_right.png" alt="Nav Rechts" /></a>
        }
    }
</div>

```

Gebruik van de server

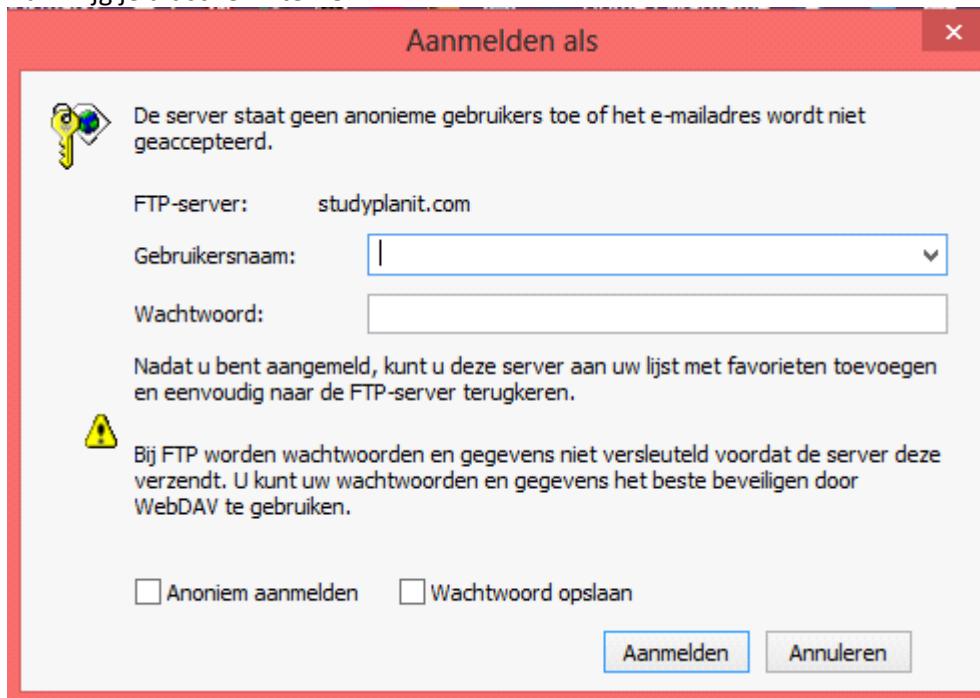
zaterdag 6 juni 2015

17:03

Aanmelden

Eerst ga je naar dit adres: <ftp://studyplanit.com/>

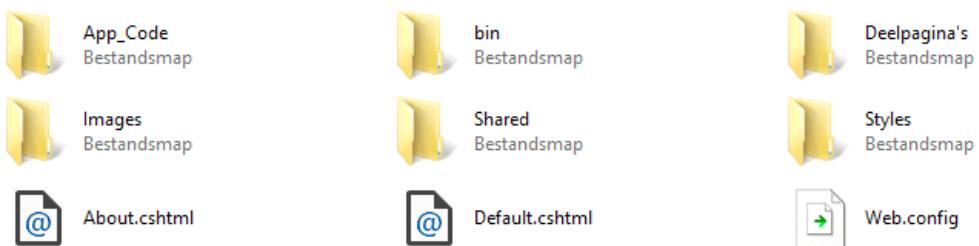
Dan krijg je dit scherm te zien.



Hier log je in om naar de server map te gaan.

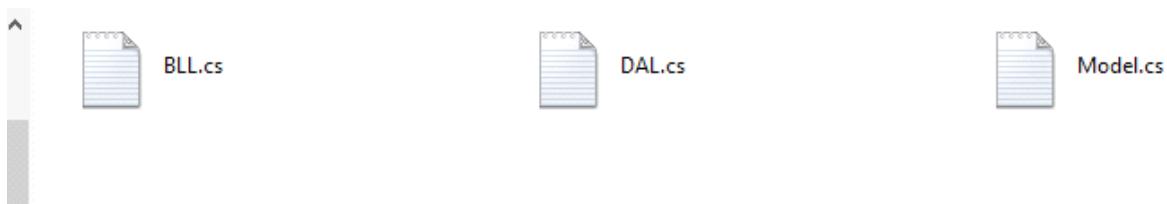
Server uploaden

Eens je in de map bent moet je de webpagina's toevoegen, de pagina's zet je in de root van de server map

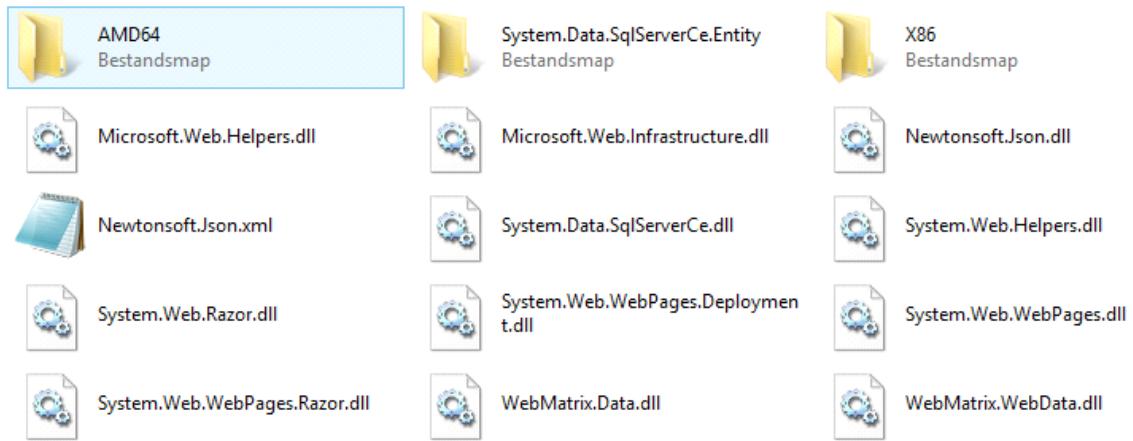


De code gaat in App_Code

Internet > studyplanit.com > App_Code > Administratix



In de folder Bin gaan alle DLL's om de code te laten werken.



Testen

Als alles op de juiste plek staat kun je de pagina bezien op www.studyplanit.com/mobilecvo



Handleiding Mobile-CVO

vrijdag 5 juni 2015

15:24

Auteur: Jo Ronsse

Inloggen

Om de website te kunnen gebruiken moet je inloggen, daarna word alle navigatie zichtbaar.

Inloggen kan met een van deze cursistnummers:

- 26544 : Sonja Van Rompaey;
- 46371 : Nikos VandenBroek;
- 46408 : Jo Ronsse;
- 26946 : Benjamin Peeters;
- 43703 : Mohamed Amajoutt;

Navigatie

Navigatie word gedaan met deze 3 knoppen:



De knop linkt terug naar de Homepage.

De knop linkt naar de vorige pagina.

De knop linkt naar de volgende pagina.

Pagina's

home:

Bevat alle links naar de andere pagina's.

Kalender:

Kalender laat de komende evenementen , taken, etc. zien per maand.

Als je op een dag klikt krijg je een gedetailleerd overzicht van die dag.

do 11/6



Lessen

Projectwerk informatica (TV) 09:00 - 10:30

Projectwerk informatica (TV) 10:45 - 12:15

Deadlines:

De pagina deadlines laat de deadlines voor de komende 2 weken zien, als je op een les klikt zie je alle opgaves voor die les.

mobileCVO

Nikos Vanden Broek (46371) [Log uit](#)

Komende 2 weken

8-6-2015

Netwerkbeheer 1 : Opdracht 3

10-6-2015

Netwerkbeheer 1 : Opdracht 2

Programmeren 5 - 11628 : Taak 2: Survey customers

Netwerkbeheer 1

Opdracht 1

Deadline: 10-5-2015 00:00

Opdracht 2

Deadline: 10-6-2015 07:30

Opdracht 3

Deadline: 8-6-2015 00:00

Programmeren 5 - 11628

Projectwerk informatica CVO-Mobile

Evenementen:

De pagina evenementen laat alle evenementen zien, je kunt ook filteren op ingeschreven evenementen.

Om je in te schrijven voor een evenement klik je op het gewenste evenement en vul je eventuele opmerkingen in het vak.

mobileCVO

Nikos Vanden Broek (46371) [Log uit](#)

Evenementen

Ingeschreven

Alle

Datum: 30-06-2015

Tijdstip: 09:00 - 17:00

Naam: Infodagen

Locatie: CVO-Antwerpen

[Schrijf me in!](#)

Alles gaat goed

Examendata:

In examendata vind je alle datums in verband met examens.

mobileCVO

Nikos Vanden Broek (46371) [Log uit](#)

Examendata

Cursusnummer: 11571

Moduledatum: 01-09-2014 tot 08-10-2014 Module: Basiskennis TV

1ste zitdatum: niet bekend

Deliberatiestdatum: 01-01-0001 00:00 2de zitdatum: 25-02-2015 00:00

Cursusnummer: 11578

Moduledatum: 01-09-2014 tot 08-10-2014 Module: Datacommunicatie en netwerken TV

Cursusnummer: 11672

Moduledatum: 03-09-2014 tot 09-10-2014 Module: Databanken TV

Cursusnummer: 11627

Moduledatum: 17-11-2014 tot 28-01-2015 Module: Programmeren 4 TV

Cursusnummer: 11583

Moduledatum: 02-09-2014 tot 30-01-2015 Module: Programmeren 1 TV

Puntenoverzicht:

In puntenoverzicht kun je alle punten zien per vak.

Puntenoverzicht

Mijn 2de zit	Alle resultaten
Cursusnummer: 11571 Module: Basiskennis TV	
PE: niet bekend	1ste zit: niet bekend
2de zit: niet bekend	Totaal: niet bekend
2de zit: niet bekend	Deliberatie 2de zit:
Inschrijven	
Cursusnummer: 11578 Module: Datacommunicatie en netwerken TV	
Cursusnummer: 11672 Module: Databanken TV	

Lessenrooster:

laat alle vakantiedagen en lesmomenten zien voor het hele jaar, je kunt ook kiezen om de feestdagen te verbergen.

mobileCVO 

Nikos Vanden Broek (46371) [Log uit](#)

Feestdagen			
Verbergen		Tonen	
03-01-2015	* einde kerstvakantie		
05-01-2015	09:00 - 12:45	Programmeren 5 TV	
Cursusnummer: 11628	Lokaal: B10	Docent: Balbeart Ivo	Campus: Hoboken
05-01-2015	13:15 - 17:00	Programmeren 4 TV	

Trajectoverzicht:

Laat zien welke vakken je voor geslaagd bent, welke vakken je nog voor moet inschrijven, en welke vakken je moet voltooien om een vak te mogen doen.

mobileCVO 

Nikos Vanden Broek (46371) [Log uit](#)

Legende			
<input type="radio"/> Geen punten	<input checked="" type="radio"/> Geslaagd.	<input checked="" type="radio"/> Niet geslaagd.	
<input checked="" type="checkbox"/> Onvoldoende voorkennis.	<input checked="" type="checkbox"/> Voldoende voorkennis.		
Analyse TV		Basiskennis TV	
nodige voorkennis		O Antwerpen in voor volwassenenonderwijs	
<input checked="" type="checkbox"/> Programmeren 1 TV		<input checked="" type="radio"/> Basiskennis TV	
<input checked="" type="checkbox"/> Programmeren 2 TV			
<input checked="" type="radio"/> Databanken TV			
Beheer van databanken TV		Besturingssystemen TV	

Sonja Van Rompaey

maandag 1 juni 2015

18:29



logboek

Nikos Vanden Broek

maandag 1 juni 2015

18:50



Logboek ...

Ronsse Jo

dinsdag 2 juni 2015
21:07



logboek

Mohamed Amajoutt

donderdag 4 juni 2015
12:26



logboekM...

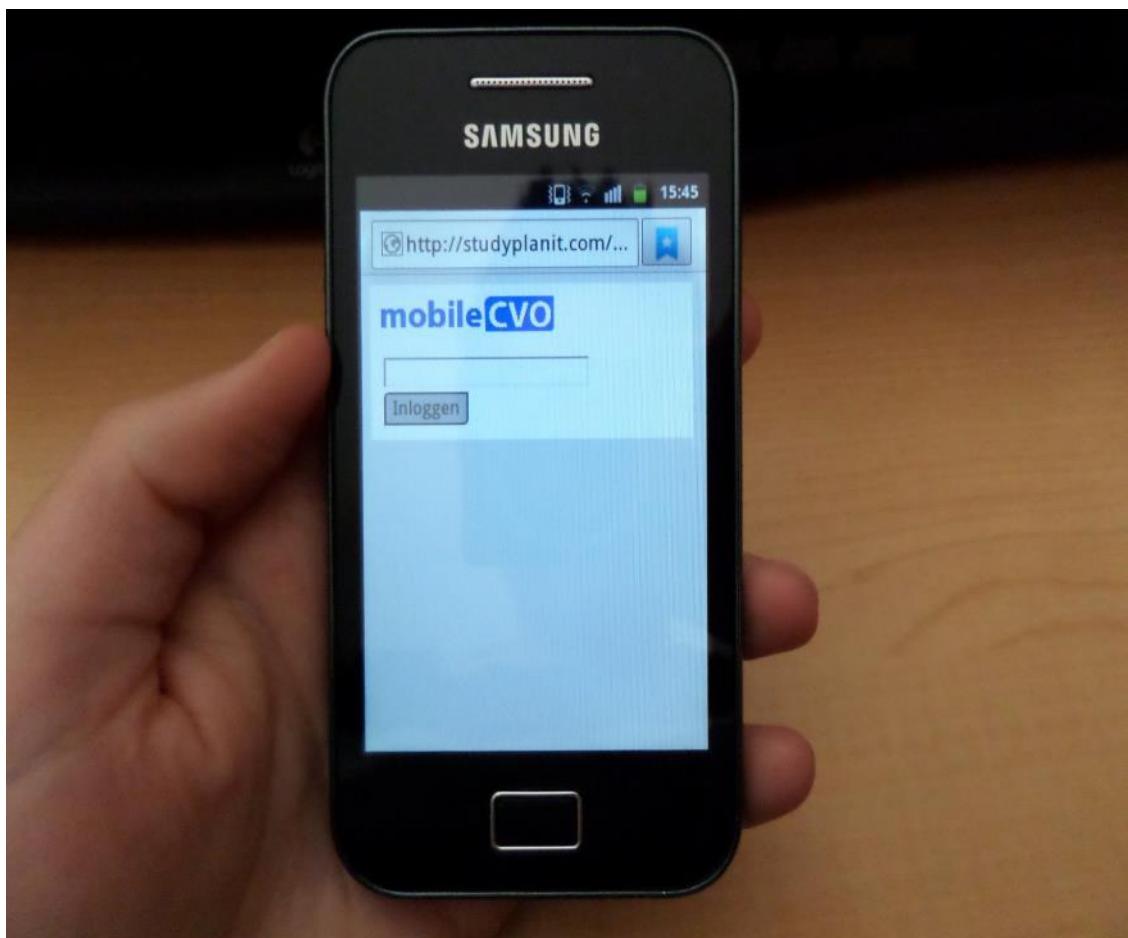
Resultaat

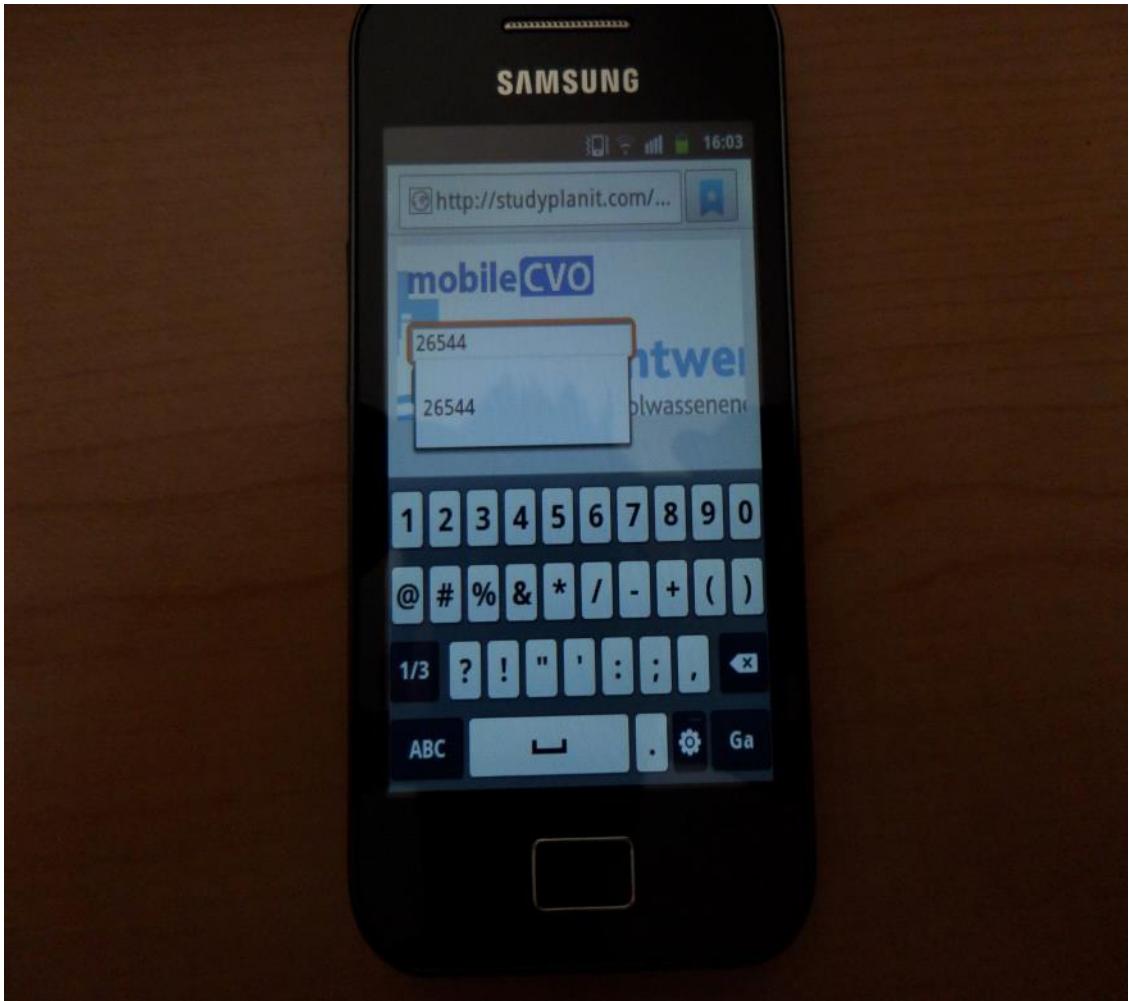
zaterdag 6 juni 2015

16:18

Inloggen

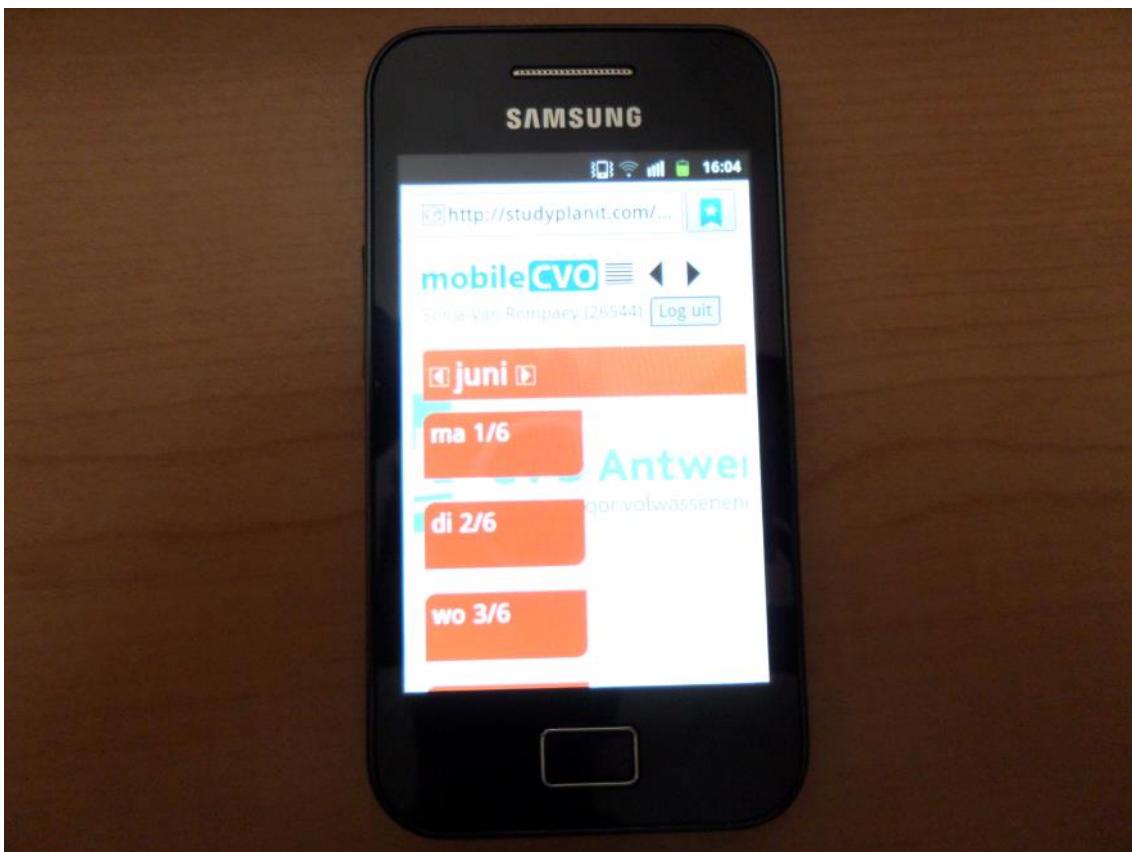
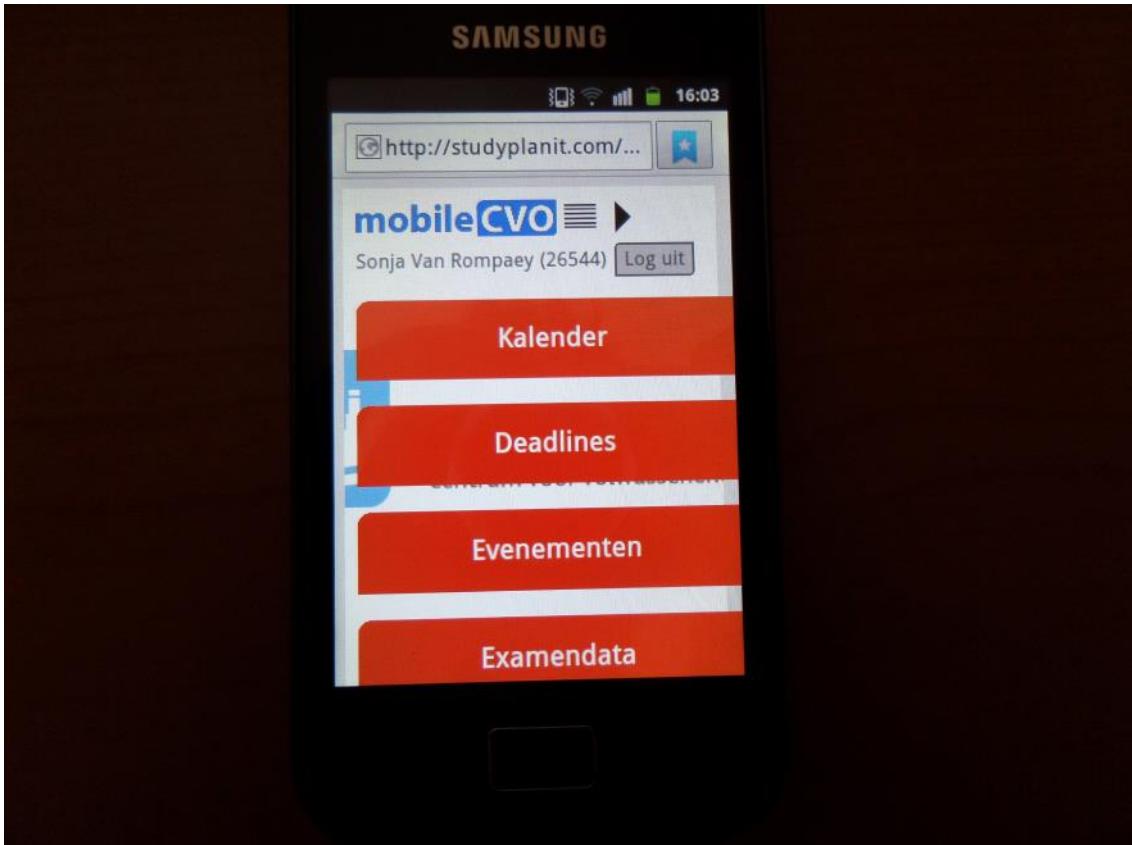
Door naar volgend adres te surfen op je mobile device kan je met een geldig cursistnummer inloggen op de app.
<http://studyplanit.com/mobilecvo/>

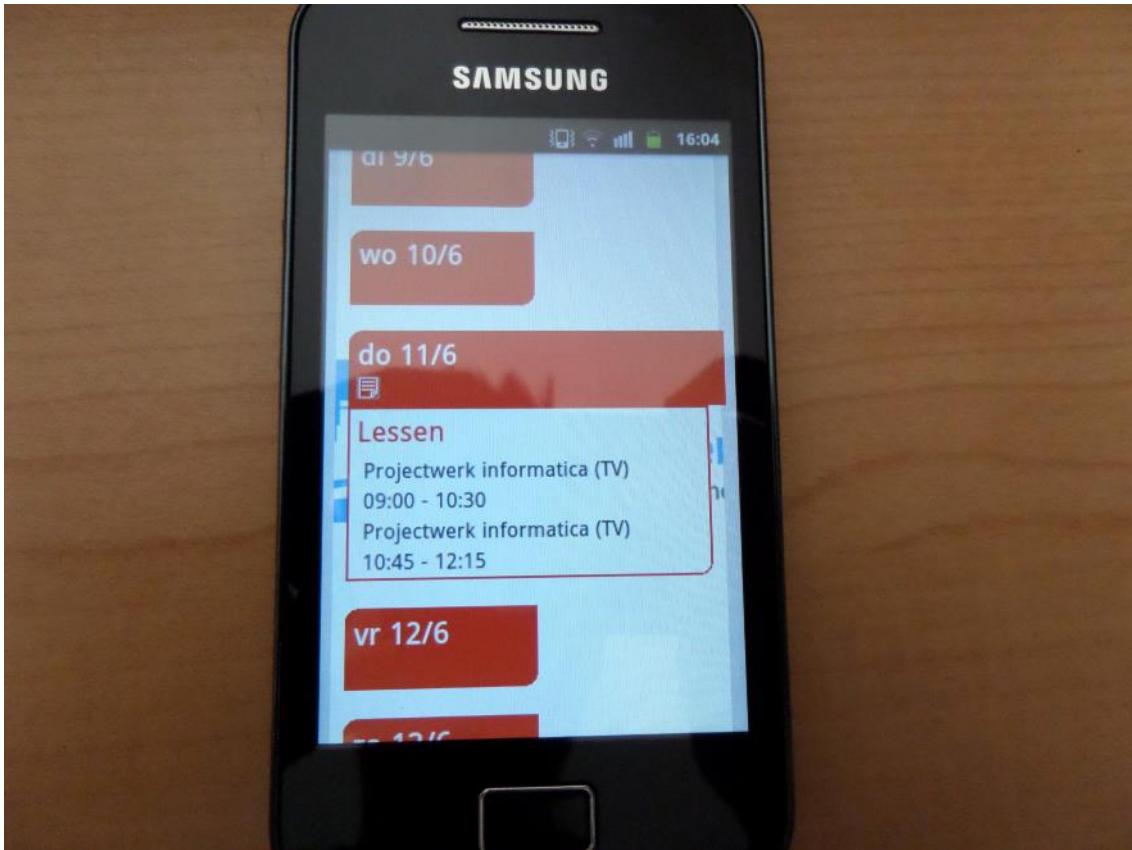




Kalender

De kalender geeft een maandelijkse weergave van alle dagen met aanduiding van:
lesmomenten, deadlines en feestdagen.





Deadlines

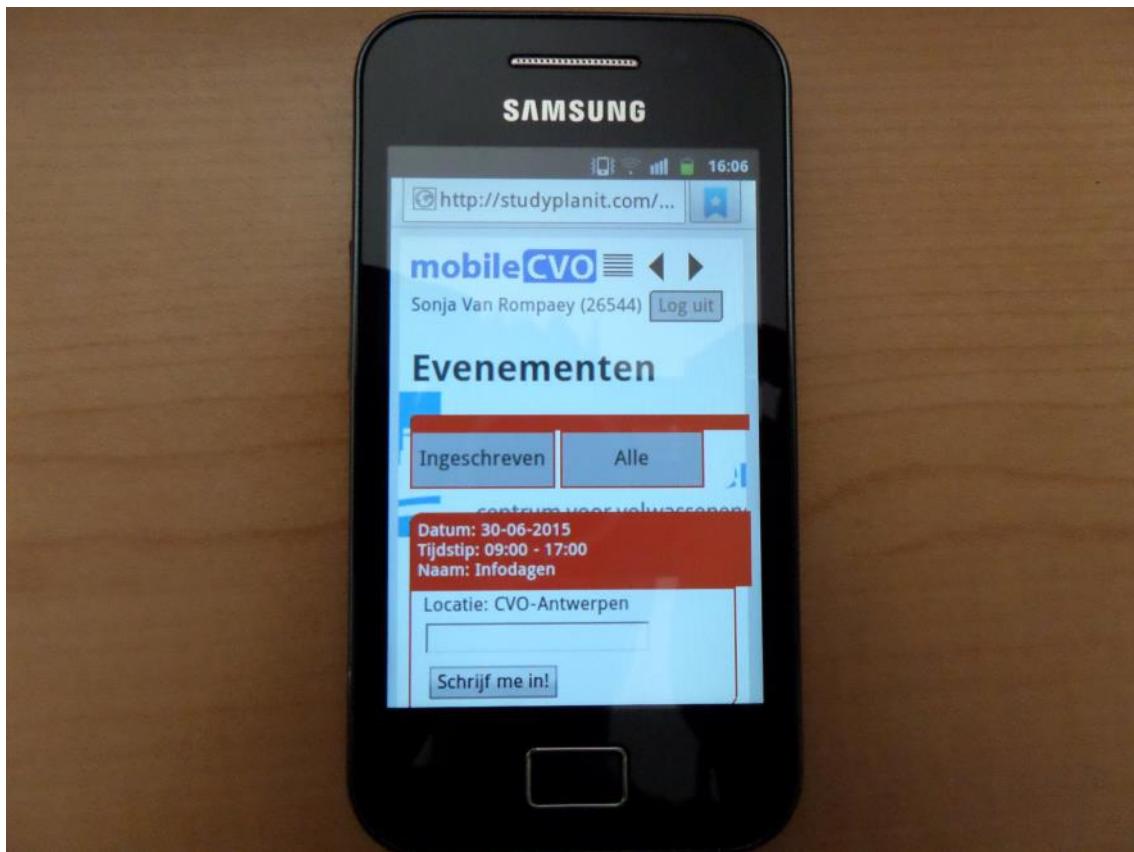
Bij deadlines krijg je eerst bovenaan een melding of er deadlines zijn voor de komende 2 weken. Daaronder kan je verder in de toekomst kijken of er nog deadlines zijn.



Evenementen

Op de evenementen page heb je een overzicht van alle evenementen en de mogelijkheid om

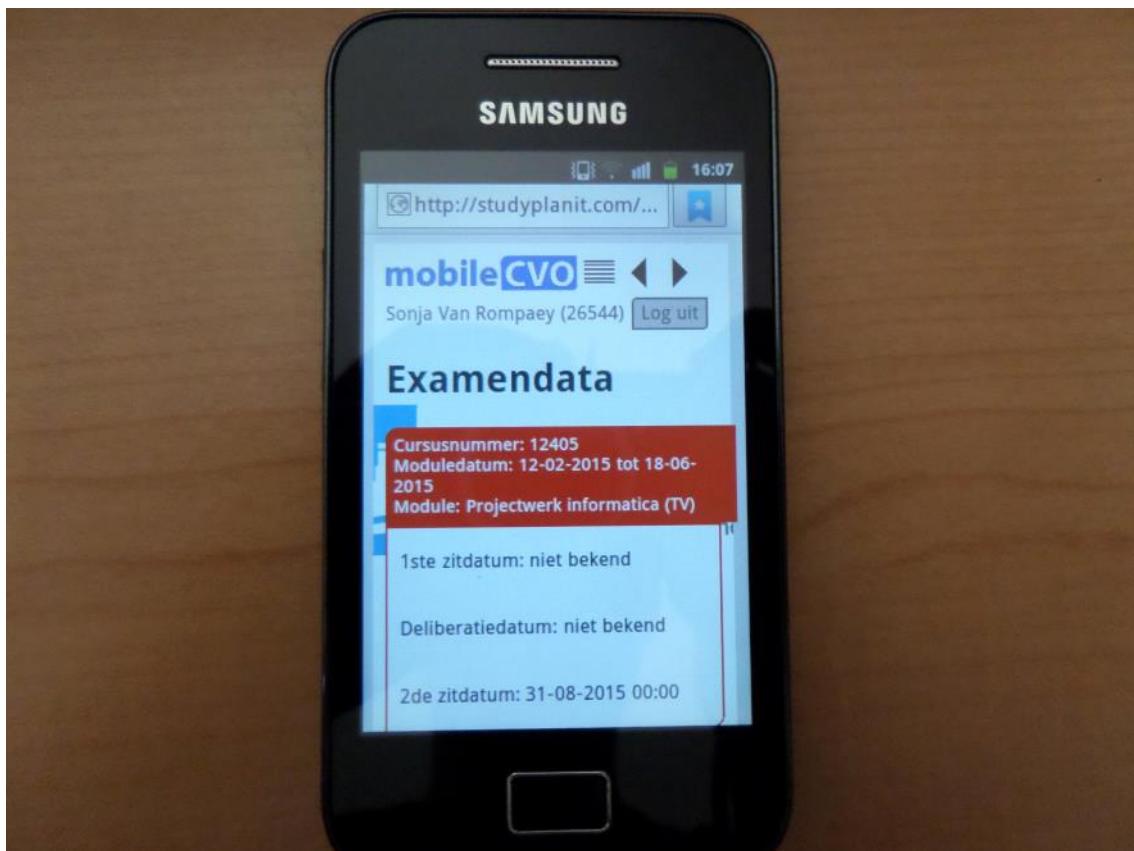
je hiervoor in te schrijven.



Examendata

Vervolgens heeft de examendate een weergave per cursus van de examendatum, deliberatiedatum en de 2de zitdatum.

Ook de cursusnummer en de begin en einddatum van een module worden weergegeven.



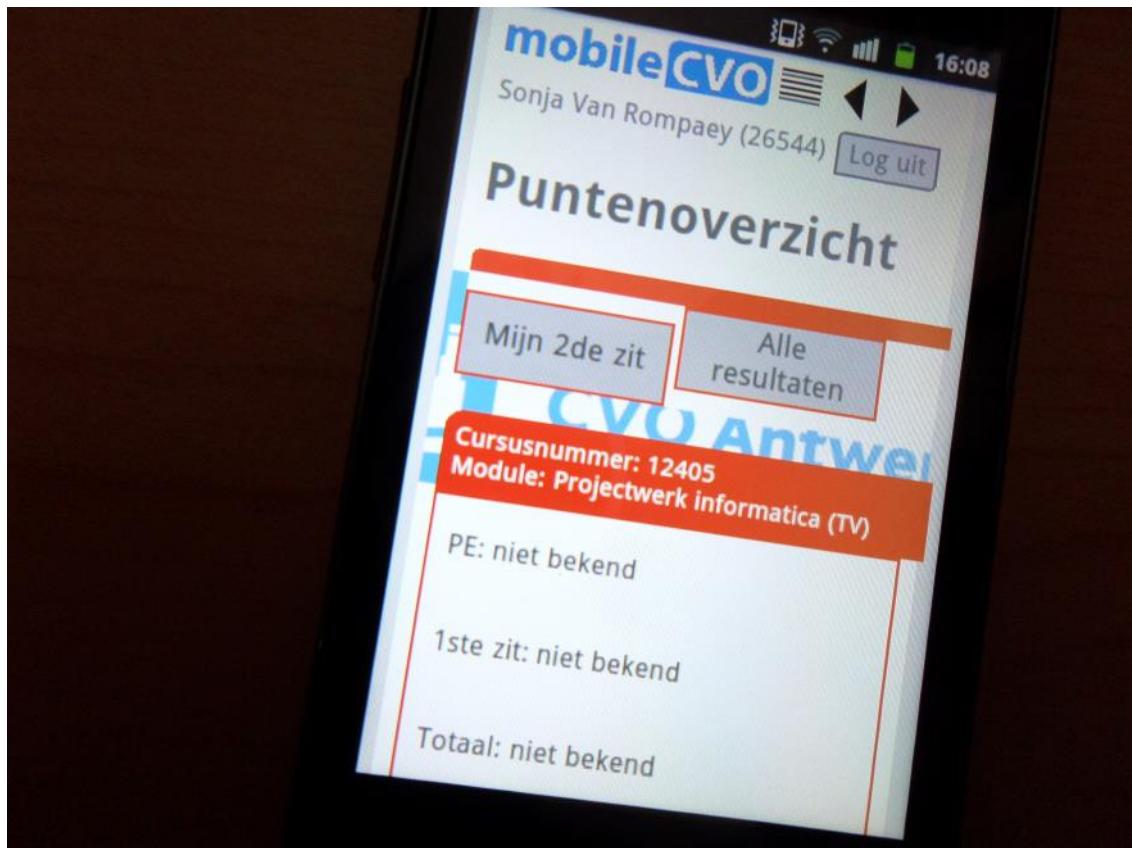
Puntenoverzicht

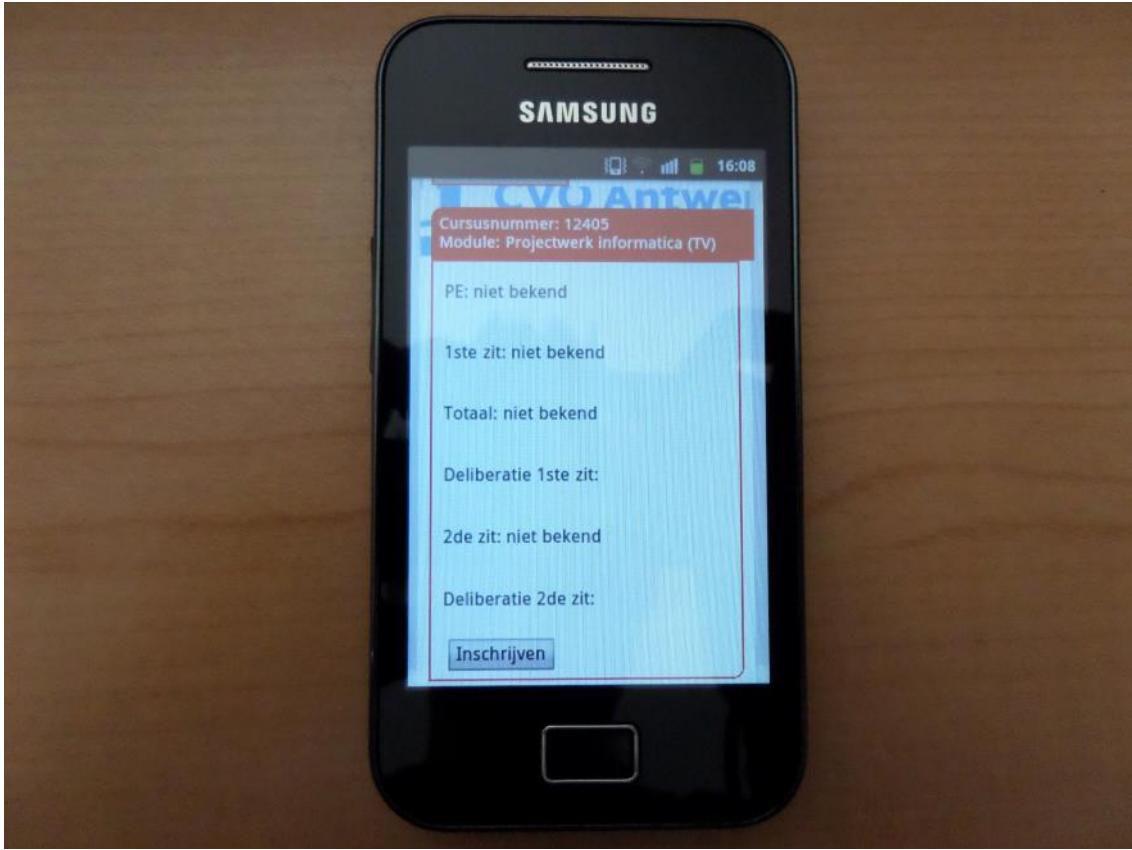
De volgende page is die van puntenoverzicht.

Hier kan je bekijken of je al bent ingeschreven voor een 2de zit, door op de button "Mijn 2de zit" te drukken.

Of alle resultaten opvragen door op de button "Alle resultaten" te drukken.

Nadat je resultaten hebt bekeken kan je onderaan inschrijven voor deelname aan een 2de zit.





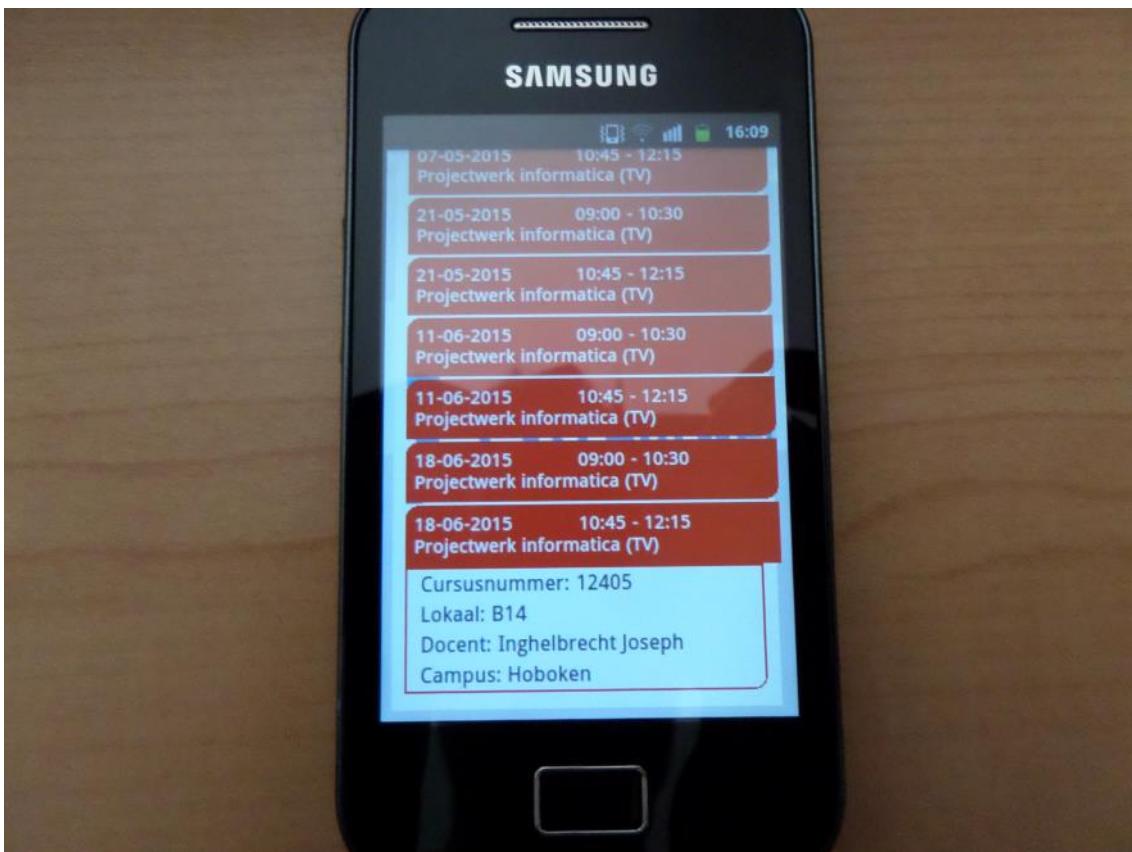
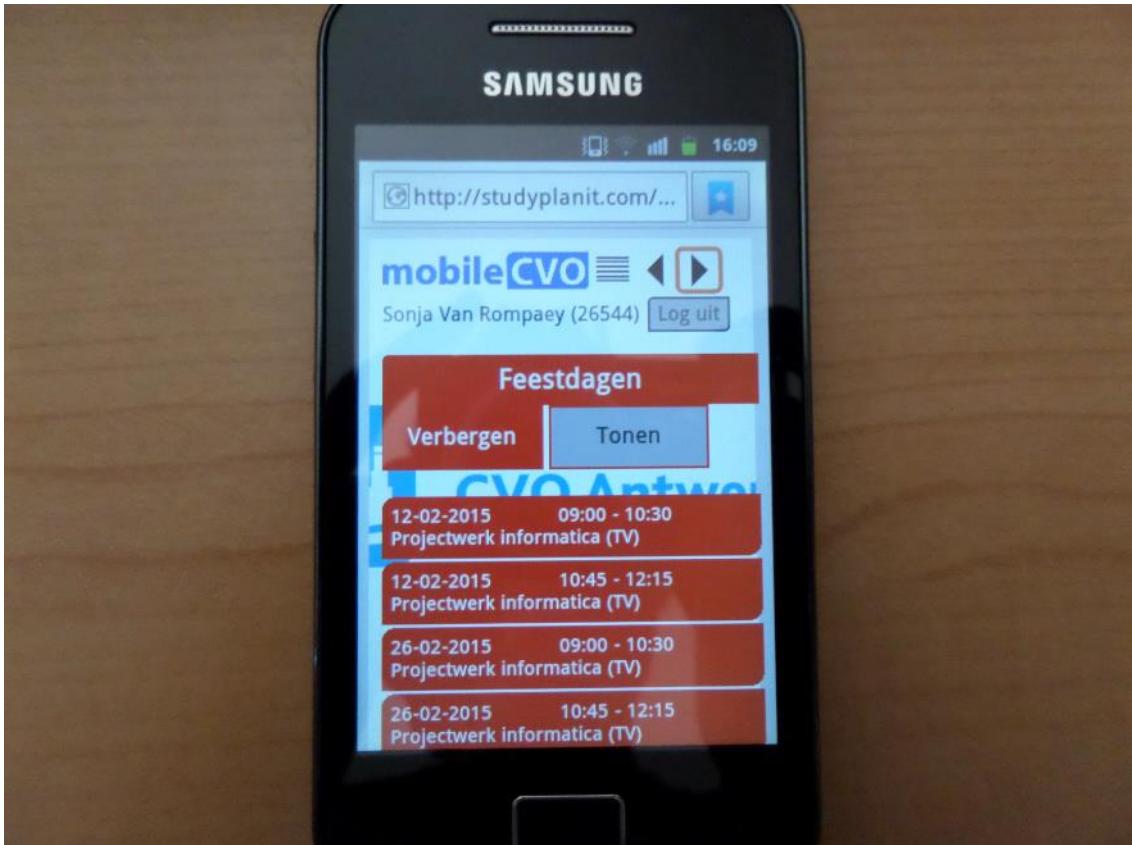
Lesrooster

Op de lesrooster page krijg je een overzicht van alle lesmomenten.

Je ziet de datum, start en einduur en de naam van de module.

Wanneer je een specifiek lesmoment aanduid krijg je nog een extra weergave met de cursusnummer, het lokaal, de docent en de campus.

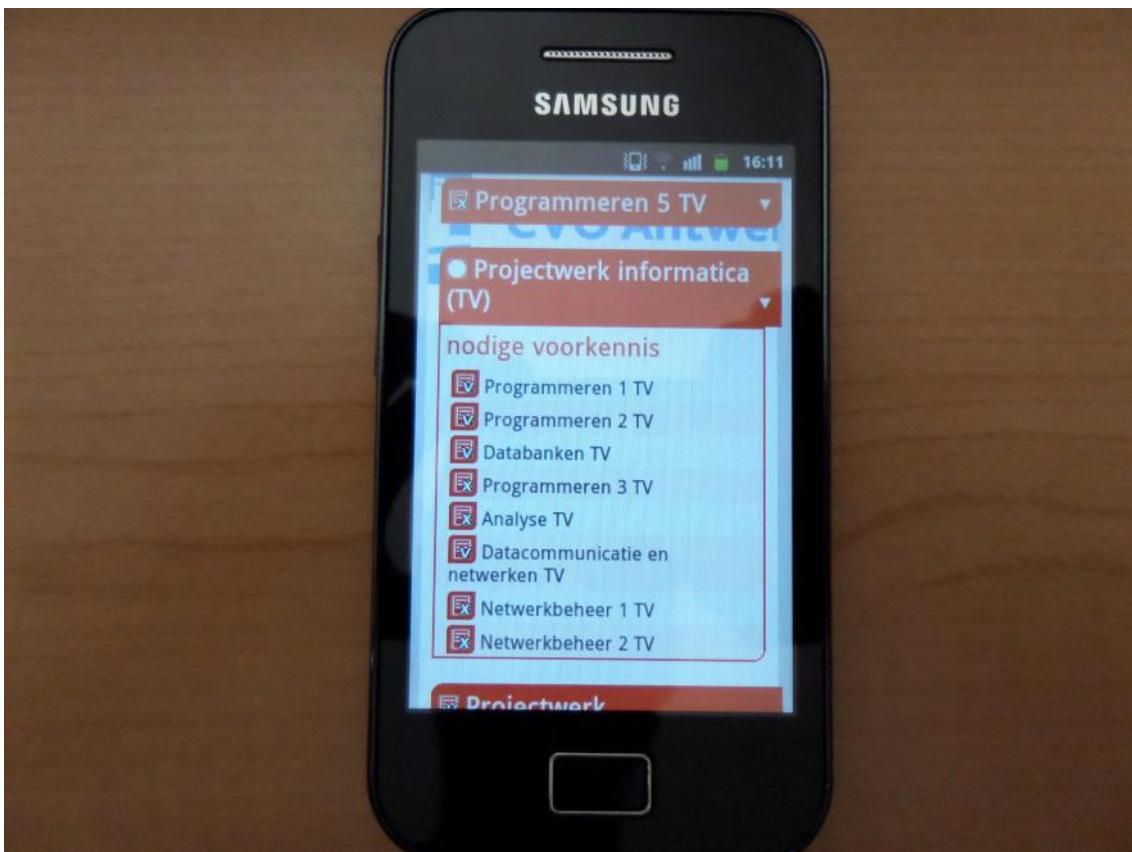
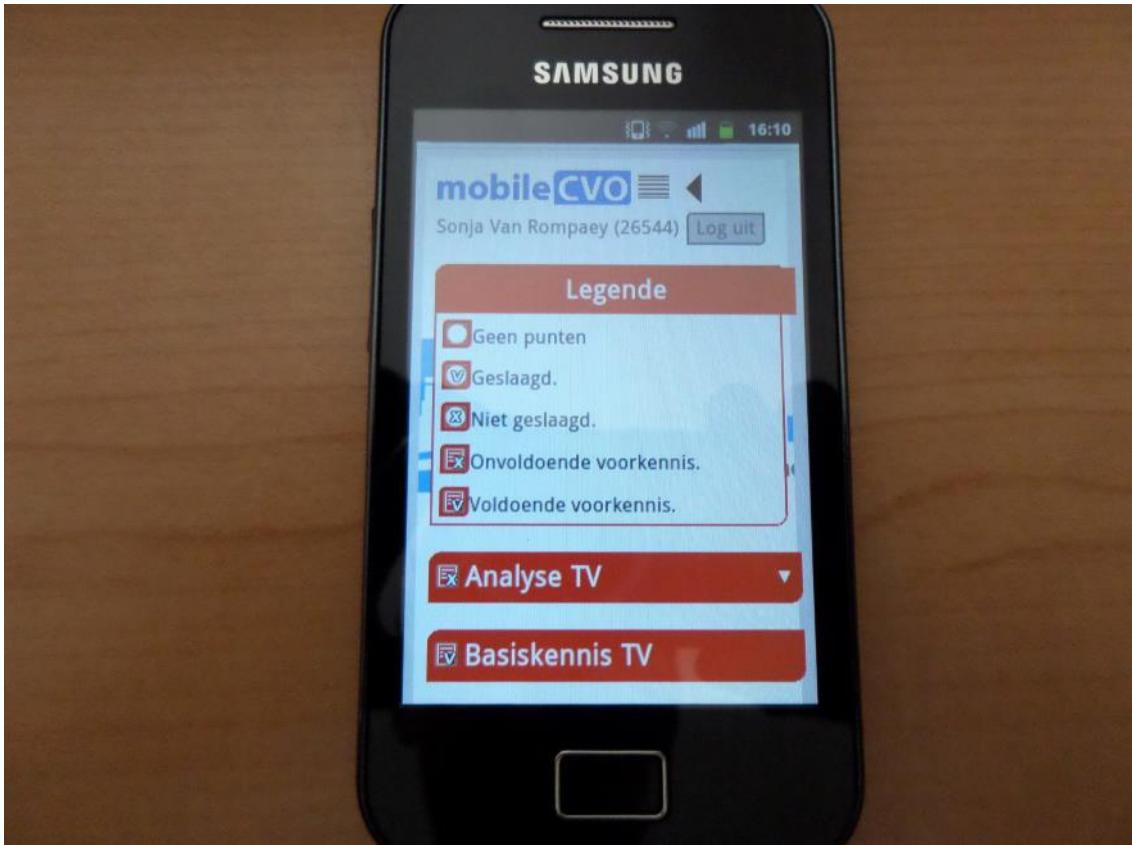
Het is ook mogelijk om feestdagen te tonen in de lesrooster, dit doe je door bovenaan op de knop "Tonen" of "Verbergen" van Feestdagen te drukken.



Trajectoverzicht

Het trajectoverzicht laat je zien welke modules al zijn afgerond en welke een cursist nog moet volgen.

Door de weergave van de voorkennis weet een cursist of hij/zij zich al kan gaan inschrijven of welke voorgaande modules er eerst moeten voldaan worden.



Moodle

Door op de tegel moodle te drukken ga je rechtstreeks naar de moodle website.
Daar kunnen taken worden ingediend.

CVO Antwerpen

De tegel cvo antwerpen verbindt de gebruiker door naar de website van de school.
Hier kan onder andere contact info worden opgezocht.

Uitloggen

Terug uitloggen kan altijd door op de knop "Log uit" te drukken.
Deze bevind zicht bovenaan op elke page.

Auteurs:

Sonja Van Rompaey

Besluit

donderdag 4 juni 2015
12:34

Bevindingen

Het was een zeer leuk project om als eindwerk te mogen doen.
Alles wat we gedurende onze opleiding hebben geleerd is van toepassing geweest en hebben we in praktijk kunnen omzetten.
Dit project is een prototype waarin we volgens mij geslaagd zijn om te maken.
Alle basis functionaliteit zit erin en voor de onderdelen die te complex waren zijn creatieve oplossingen gezocht, er is tot het bittere einde fanatiek doorgewerkt.

Groepsleden

Ik heb veel bijgeleerd door het kunnen werken in teamverband.
Samen coderen en andere personen hun code lezen en testen biedt perspectief en motivatie om mijn eigen manier van werken aan te passen en te verbeteren.
Dit werkt zonder twijfel ook in beide richtingen, want op vele vlakken zijn we als groep aan elkaar gewaagt.

Bedankt voor deze leerrijke ervaring:

Nikos Vanden Broek

Mohamed Amajoult

Jo Ronsse

Benjamin Peeters

Andreas De Bresser

Dankwoord

Uiteraard bedank ik ook de docenten om last minute zich op te geven om deze module te willen geven, initieel was deze module volzet en niet ingepland.

Bedankt om ons de kans te geven om dit schooljaar nog af te studeren:

Jef Inghelbrecht
en
Ivo Balbaert

Auteurs:

Sonja Van Rompaey

Bronvermelding

donderdag 4 juni 2015

16:35

Cursussen:

Programmeren 3

Programmeren 4

Programmeren 5

Informatie:

Google

www.google.be

Tutorials:

ASP.NET Razor

[http://www.asp.net/web-pages/overview/ui,-layouts,-and-themes/3-creating-a-c_consistent-look](http://www.asp.net/web-pages/overview/ui,-layouts,-and-themes/3-creating-a-consistent-look)

W3Schools

<http://www.w3schools.com/>

Elementen:

IconFinder

<https://www.iconfinder.com/>

Tools:

Github

<https://github.com/mamajoutt/ProjectMobileCVO>

Facebook

www.facebook.com

Software

www.academicdownload.com

Youtube

www.youtube.com

Screencast-O-Matic

<http://www.screencast-o-matic.com/>

Auteurs:

Sonja Van Rompaey