Canvas

Algemeen

Begrippen

Canvas	Synoniemen
Cursus / Course	Cursus Blok
Opdrachtengroep / Assignmentgroup	Module (=onderdeel van blok)

Studenten codes

Er is een functie beschikbaar, maar die is om veiligheidsredenen niet gepubliceerd.

Hidden reports

Aantal pogingen per opdracht per student	http://localhost:8080/query/attempts
Productiefste dagen van studenten	http://localhost:8080/query/day-of-week
Productiefste dagdelen van studenten	http://localhost:8080/query/dagdeel
Gemiddelde score per module	http://localhost:8080/query/moeilijk
Snel inleveren	http://localhost:8080/query/rapid
Access Log	http://localhost:8080/query/log

Nieuwe users

1. Gebruik script import_users.py (in repo onder /import) om users uit een Canvas-cursus te importeren.
   Het script maakt SQL-code.

   Let op gebruik van juiste database!

2. Gebruik Eduarte-docent (easy versie) om een CSV van een klas naar Excel te exporteren.
   Voeg deze regel aan het eind toe en voer queries uit.

```
="update user set klas='2B' where student_nr='"&A2&"';"
```

3. Maak unieke access codes voor de studenten.

   Codes zijn afhankelijk van een 'salt' deze kan alleen worden aangepast door de code aan te passen en moet in principe één keer per jaar worden veranderd waarna iedere student een nieuwe code krijgt.

```
Informatie verwijderd in verband met security
```

Windows Dev Omgeving

hosts

`C:\Windows\System32\drivers\etc\hosts`

```
127.0.0.1 c20.cmon.local
127.0.0.1 c21.cmon.local
127.0.0.1 c22.cmon.local
```

vhosts

`C:\xampp\apache\conf\extra\httpd-vhosts.conf`

```
<VirtualHost *:80>
  ServerName cmon.local
  ServerAlias c21.cmon.local c22.cmon.local c20.cmon.local
  DocumentRoot "C:\Users\maxbi\www\yii2\canvas\web"
  RewriteEngine on
  RewriteCond %{SERVER_NAME} =c22.cmon.ovh [OR]
  RewriteCond %{SERVER_NAME} =cmon.ovh [OR]
  RewriteCond %{SERVER_NAME} =c21.cmon.ovh
  RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
</VirtualHost>
```

Create view in DB

(view moet elk jaar worden aangepast)

View is nodig om een overzicht te krijgen van alle nagekeken opdrachten (per docent).

Old

```
drop view all_submissions;

create view all_submissions as
SELECT *, 'c22' as cohort FROM `canvas-c20`.`submission`
union
SELECT *, 'c23' as cohort FROM `canvas-c21`.`submission`
UNION
SELECT *, 'c24' as cohort FROM `canvas-c22`.`submission`;
```

New as from c23

```
DROP VIEW IF EXISTS `all_submissions`;

CREATE view all_submissions as

SELECT
d.id as module_id,
d.naam as module_name,
d.pos as module_pos,
d.generiek as generiek,
u.name as student_name,
u.student_nr as student_nr,
u.grade as grading_enabled,
g.name as grader_name,
c.korte_naam as cursus_short_name,
s.submitted_at as submitted_at,
s.graded_at as graded_at,
s.entered_score as entered_score,
r.minpunten as minpunten,
'c22' as cohort
from `canvas-c22`.submission s
inner join `canvas-c22`.assignment a on a.id=s.assignment_id
inner join `canvas-c22`.module_def d on d.id=a.assignment_group_id
inner join `canvas-c22`.user u on u.id=s.user_id
left outer join `canvas-c22`.user g on g.id=s.grader_id
inner join `canvas-c22`.course c on c.id=s.course_id
```

```
inner join `canvas-c22`.resultaat r on r.module_id=d.id and r.student_nummer = u.student_nr

UNION

SELECT
d.id as module_id,
d.naam as module_name,
d.pos as module_pos,
d.generiek as generiek,
u.name as student_name,
u.student_nr as student_nr,
u.grade as grading_enabled,
g.name as grader_name,
c.korte_naam as cursus_short_name,
s.submitted_at as submitted_at,
s.graded_at as graded_at,
s.entered_score as entered_score,
r.minpunten as minpunten,
'c21' COLLATE utf8mb4_general_ci as cohort
from `canvas-c23`.submission s
inner join `canvas-c23`.assignment a on a.id=s.assignment_id
inner join `canvas-c23`.module_def d on d.id=a.assignment_group_id
inner join `canvas-c23`.user u on u.id=s.user_id
Left outer join `canvas-c23`.user g on g.id=s.grader_id
inner join `canvas-c23`.course c on c.id=s.course_id
inner join `canvas-c23`.resultaat r on r.module_id=d.id and r.student_nummer = u.student_nr

union

SELECT
d.id as module_id,
d.naam as module_name,
d.pos as module_pos,
d.generiek as generiek,
u.name as student_name,
u.student_nr as student_nr,
u.grade as grading_enabled,
g.name as grader_name,
c.korte_naam as cursus_short_name,
```

```
s.submitted_at as submitted_at,
s.graded_at as graded_at,
s.entered_score as entered_score,
r.minpunten as minpunten,
'c24' COLLATE utf8mb4_general_ci as cohort
from `canvas-c24`.submission s
inner join `canvas-c24`.assignment a on a.id=s.assignment_id
inner join `canvas-c24`.module_def d on d.id=a.assignment_group_id
inner join `canvas-c24`.user u on u.id=s.user_id
left outer join `canvas-c24`.user g on g.id=s.grader_id
inner join `canvas-c24`.course c on c.id=s.course_id
inner join `canvas-c24`.resultaat r on r.module_id=d.id and r.student_nummer = u.student_nr
```

Set Collate in all tables

```
ALTER TABLE assignment
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE assignment_group
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE course
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE check_in
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE log
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE login_user
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE module
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE module_def
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE resultaat
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE submission
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
ALTER TABLE `user`
CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
```

Database (main entities)

image-1666113019407.png

Onderstreepte entiteiten komen 1:1 uit de Canvas API.

Resultaat wordt berekend na de import en is een gegenereerde "de-normalized table" ten behoeve van de eenvoud en performance.

Een course (meestal blok) heeft meerdere assignement_groups. De tabellen module, assignement_group en module_def zijn feitelijk één tabel. De reden dat er drie tabellen zijn heeft te maken met de Canvas API, de module en assignement_group komen 1:1 uit de API en de module_def zijn eigen toevoegingen.

In schema

image-1666113613887.png

Server

Backup files (www, home, en apache2)

Backup gaat via Restic naar Object Storage en wordt via crontab dagelijks gedraaid.

```
# backup to eu2.contabostorage.com (s3) and clean once a week
30 23 * * * /home/.../restic/restic-backup.sh > /home/max/log/restic-backup.log 2>&1
# clean up
55 23 * * 0 /home/.../restic/restic-clean.sh  > /home/max/log/restic-clean.log 2>&1
```

Er worden drie backup's gemaakt:

```
restic backup /var/www/       --exclude-file=exclude.txt
restic backup /home/.../      --exclude-file=exclude.txt
restic backup /etc/apache2/   --exclude-file=exclude.txt
```

Voor restoren van een snapshot zoek je eerst *snapshot id* op met `restic-list.sh`. Over het algemeen neem je de onderste uit de list tenzij je verder terug wilt.

Daarna het volgende commando invoeren

```
./restic.sh restore <id> --target /tmp/restore
```

restic.sh

```
#! /bin/bash

export AWS_ACCESS_KEY_ID="xxx-xxx-xxx"
export AWS_SECRET_ACCESS_KEY="xxx-xxx-xxx"
export RESTIC_REPOSITORY="s3:server/bucket"
export RESTIC_PASSWORD="xxx-xxx-xxx"

restic $@
```

restic manual: https://restic.readthedocs.io/en/latest/index.html

Backup SQL

Gaat via dit script dat via Cron om 23:xx wordt aangeroepen.

```
#!/bin/bash

#####################################################################
##
##   MySQL Database Backup Script
##
#####################################################################

export PATH=/bin:/usr/bin:/usr/local/bin
TODAY=`date +"%d%b%Y"`
TODAY=`date +"%m%d-%H00"`

#####################################################################
################## Update below values  #######################

DB_BACKUP_PATH='/home/.../mysql/Backup-DB'
MYSQL_HOST='localhost'
MYSQL_PORT='3306'
MYSQL_USER='root of wat anders'
MYSQL_PASSWORD='xxxx-xxxx-xxxx'
DATABASE_NAMES='db1 db2 db3 db4 db5'
BACKUP_RETAIN_DAYS=180   ## Number of days to keep local backup copy

#####################################################################
```

```
for DATABASE_NAME in ${DATABASE_NAMES}
do

  echo "Backup started for database - ${DATABASE_NAME}"

  mkdir -p ${DB_BACKUP_PATH}

  mysqldump -h ${MYSQL_HOST} \
     -P ${MYSQL_PORT} \
     -u ${MYSQL_USER} \
     -p${MYSQL_PASSWORD} \
     ${DATABASE_NAME} | gzip > ${DB_BACKUP_PATH}/${DATABASE_NAME}-${TODAY}.sql.gz

  if [ $? -eq 0 ]; then
    echo "Database backup successfully completed"
  else
    echo "Error found during backup"
    exit 1
  fi

done

# remove all files older than 90 days excpet those made on the 1ste of the month
find ${DB_BACKUP_PATH} -type f -not -name '*01-????.sql.gz' -mtime +90 -exec rm {} \;
# remove all files not created at 23:00 and older than 8 days
find ${DB_BACKUP_PATH} -type f -not -name '*300.sql.gz' -mtime +8 -exec rm {} \;

##### Remove backups older than {BACKUP_RETAIN_DAYS} days  #####
find ${DB_BACKUP_PATH}/* -mtime +${BACKUP_RETAIN_DAYS} -exec rm {} \;
```

--

Canvas Resultaten Overzicht

Hoe nieuw blok toe te voegen?

Deze informatie is niet meer relevant

Stap 1

In de import.py module onderaan een blokje code toevoegen.

```
# Blok 3
if (1):
    createBlok(3237)
```

Het nummer is het cursus ID. Deze is te vinden door in canvas het Blok/cursus te openen. Het id staat in de url.

Op regel 2 kan de (1) worden vervangen door (0) als er geen updates meer in het blok te verwachten zijn.

Eventueel kan later script worden aan gepast met cmd line parameters zodat bepaalde blokken vaker worden geupdate.

Stap 2

In de code (sorry) aanpassen:

```
// in view resultaat/index
...
[
 'label' => 'Blok',
 'attribute'=>'course_id',
 'contentOptions' => ['style' => 'width:40px; white-space: normal;'],
  'filter' => ['2101'=>'Blok 1','2110'=>'Blok 2', '3237'=>'Blok 3'],
  'format' => 'raw',
  'value' => function ($data) {
     if ($data->course_id==2101) return "B1";
     if ($data->course_id==2110) return "B2";
     if ($data->course_id==3237) return "B3";
 }
```

```
],
```

Regel 7 uitbreiden en regel na 12 toevoegen.

Canvas module toevoegen aan de montor

Is het blok (in Canvas heet dat cursus) al aan de monitor toegevoegd?

Nee, ga dan naar de cursus in Canvas en voer de volgende stappen uit.

Open de juiste cursus, bijv.

image-1673895542986.png

Cursus ID

Het nummer bovenaan in het internetadres (in dit voorbeeld 7720) is het cursus ID. In het plaatje wordt dit door de bovenste rode pijl aangegeven.

People

Onder people (blauwe pijl in plaatje) kun je studenten en docenten aan de cursus toevoegen. Klik op people. En voeg een persoon toe (knop rechts boven).

image-1673895797933.png

1. Selecteer SIS ID.
2. Vul dan bij SIS ID M122790 in.
3. Selecteer bij Role "Teacher".
4. Bij section moet je de volledige sectie keizen.

Assignments

Op het eerste plaatje (bovenaan de pagina) bij de onderste rode pijl staat *Assignments*, ga daar naar toe.

Je ziet nu bijvoorbeeld he volgende:

image-1673896034970.png

Dit zijn de assignement groups waarin alle opdrachten van de deze cursus staan. Een module in de Canvas Monitor komt overeen met een assignment group in Canvas.

In dit voorbeeld zijn er drie assignment groups. De derde begint met een ! en dat betekent dat de Canvas Montor deze groep "niet ziet".

De eerste twee Assignement groups komen in Canvas.

Koppeling (aanvragen)

Weet je zelf hoe dit werkt dan kan je met het beheer account onder beheer - modules de assignment group in de monior activeren.

Let op de "SQL voldaan regel", wanneer die onjuist is, zal de Canvas monitor niet werken.

Kun of wil je de assignement group niet zelf activeren in de monitor, stuur dan een bericht (via Teams) naar de Canvas Monitor beheerder met het Cursus ID (zie paragraaf hierboven). De cursus wordt toegevoegd en ná de import (die draait 3x per dag) is de cursus beschikbaar in de monitor.

Stuur verder de naam van de assignment groep en geef aan bij hoeveel punten de student een groen vinkje moet krijgen.

Stel je hebt 10 opdrachten waarvoor je elk 10 punten kan krijgen dan is het aan te raden om de grens op minimaal 91 punten te zetten. Zet je de grens lager dan gaan studenten (proberen) om een opdracht over te slaan.

Voorbeeld bericht.

> Gaarne cursus **7720** in de Canvas Monitor zetten.
>
> De volgende twee modules toevoegen
>
> **Nederlands OP1 - C22**, groen vinkje bij 91 punten of meer.
>
> **Nederlands OP2 - C22**, groen vinkje bij 91 punten of meer.

Extra opties

Voor wat betreft de groene vinkjes zijn er iets meer mogelijkheden. Zo kan je ook instellen dat een student bijvoorbeeld voor elke opdracht minimaal x punten moet halen of dat studenten minimaal x opdrachten moet inleveren.

Een gemiddelde is hetzelfde als een minimaal aantal punten; stel je hebt 6 opdrachten waarvoor je 0-10 kan krijgen. Je wilt een gemiddelde van een 5 of hoger, dan moet je minimaal 30 punten halen (5 x 6).

Module niet zichtbaar in CM

- De gebruiker van de API key heeft geen toegang tot de module?
- Geen cursisten aan module toegevoegd?
-

--

Nieuw Cohort

Deze versie staat online op: https://www.roc.ovh/books/canvas/page/nieuw-cohort

Aanpassingen op Server

- Pas de file config/**subdomain.php** aan:

```
<?php

function subDomain() {
    $subDomain = "www";

    if ( isset($_SERVER['SERVER_NAME'])) {
        $subDomain = explode('.', $_SERVER['SERVER_NAME'])[0];
    }



    if ( $subDomain == '...') {
        $DB='......';
    } elseif ( $subDomain == '...') {
        $DB='......';
    } elseif ( $subDomain == '...') {
        $DB='......';
    } else {
        $DB='...';
    }

    return $DB;
}

?>
```

- pas de file `config/params.php` aan:

```
# for searching students in home page
$databases = ['canvas-c24', 'canvas-c23', 'canvas-c22', 'canvas-c21', 'canvas-c20'];
```

- pas de controller *ResultaatController*, function *actionRotate* aan.
- Maak de nieuwe database aan.
- Pas de SQL views aan.
- `/etc/apache2/sites-enabled/....conf` server alias toevoegen

  vergeet de **dev-omving** niet,
  `C:\xampp\apache\conf\extra\httpd-vhosts.conf`
  en eventueel de hosts file in
  `C:\Windows\System32\drivers\etc`
- Maak gebruikers/docenten in database aan (export/import uit vorig cohort)
- Voeg *cursussen* toe via de GUI (menu beheer - cursus)
- Draai een update (op de sever `python3 update.py -c 12345`).
- Voeg *modules* toe via de GUI (menu beheer - module).
- Draai weer een update.
- Pas de *rotate* functie aan (als je op het log links boven klikt).
  `ResultaatController.php,  function actionRotate()`
- Pas de vakanties aan in de *import.py* ten behoeve van de *predictions.*
- Gebruik de
  `public function actionGenerate($code = 0)`
  in de `StudetController` om de unieke codes te genereren voor de studenten.

  Note 1: Kijk even hoe dit aan te roepen (ivm security wordt dat verder hier niet uitgelegd).
  Note 2: indien een student via de CMON wordt ingevoerd (dus niet rechtstreeks in de database wordt geïmporteerd) dan wordt deze code automatisch gegenereerd.

Importeren Studenten

Zorg dat alle studenten in een cursus staat door deze cursus te koppelen met de juiste Eduarte groepen.

Draai dan dit script dat in de folder python_scripts in de Canvasfolder staat.

1. Pas op regel 21 het cursus id aan zodat dit overeenkomt met de cursus waaruit de studenten moeten worden gehaald.
2. Draai het script.

3. Copy/paste de output in phpmyadmin.
4. Run de import op de module waar de studenten namen in staan, vb:
   `python3 import.py -c 18660`
5. Pas handmatig (via de CMON GUI) de klassen aan.

```
# Get all users created in a particular year (note that these must be assigned to a course)
# then create inserts to insert these users in the database
# use this script at the beginning of the year to add students to the CM (klas still needs to
be edited by hand)

from canvasapi import Canvas
import configparser
import sys

config = configparser.ConfigParser()
if ( not (config.read("../import/canvas.ini") or config.read("canvas.ini"))):
    print()
    dd('Error: canvas.ini not found')

# Canvas API URL
API_URL = config.get('main', 'host')
# Canvas API key
API_KEY = config.get('main', 'api_key')

canvas = Canvas(API_URL, API_KEY)

course_id = 18660

course = canvas.get_course(course_id)
all_users = course.get_users(enrollment_type=['student'])

print(f"Course Name: {course.name}")

# Iterate through the list of users
i=1
for user in all_users:
    student_nr = user.sis_user_id[1:]
    email = student_nr + 'talnet.nl'
    # print(f"{i} id: {user.id}, name: {user.name}, login_id: {email} student_nr:
```

```
{student_nr}")
    print(f"insert into user (id, name, login_id, student_nr, klas) values ('{user.id}',
'{user.name}', '{email}', '{student_nr}', '4x');")
    i=i+1
```

Email versturen naar studenten

Elke student heeft een unieke link naar zijn Canvas monitor.

- Maak een export van alle studenten uit de canvas monitor (menu - beheer - studenten (export).
- Voeg een kolom toe dir de volledige URL weergeeft.

```
="https://c20.cmon.ovh/public?code="&G28
```

- Voeg een kolom toe die (later) via een Python script een mail via MS Outlook verstuurd.

```
="Emailer('Beste "&D2&",<br><br>Jouw personal Canvas Monitor link
is:<br>"&I2&"<br><br>Succes!','Personal Link for Canvas Monitor','"&C2&"@talnet.nl')"
```

- Zet outlook "*offline*".
- Maak een *emailer.py* python file:

```
import win32com.client as win32

def Emailer(text, subject, recipient):
    outlook = win32.Dispatch('outlook.application')
    mail = outlook.CreateItem(0)
    mail.To = recipient
    mail.Subject = subject
    mail.HtmlBody = text
    mail.send

# plak hier de emailer regels uit Excel
```

- Plaats de 'e-mailregels' uit Excel in het script en draai het script.
- Controleer de e-mail in Outlook. Indien juist, zet Outlook "*online*" en de e-mail wordt verstuurd.

Database Back-up

Zet de nieuwe Canvas Database in de (SQL) backup: `/home/max/mysql/backupDB.sh`

Crontab (op server)

```
# evaluate prioritization for updates
23 04 * * * cd /home/max/canvas/import/c24 ; /usr/bin/python3 /home/max/canvas/import/bin/set-prio.py > /home/max/canvas/import/c23/set-prio.log 2>&1
22 04 * * * cd /home/max/canvas/import/c23 ; /usr/bin/python3 /home/max/canvas/import/bin/set-prio.py > /home/max/canvas/import/c23/set-prio.log 2>&1
21 04 * * * cd /home/max/canvas/import/c22 ; /usr/bin/python3 /home/max/canvas/import/bin/set-prio.py > /home/max/canvas/import/c22/set-prio.log 2>&1
20 04 * * * cd /home/max/canvas/import/c21 ; /usr/bin/python3 /home/max/canvas/import/bin/set-prio.py > /home/max/canvas/import/c21/set-prio.log 2>&1

# Prio 1 updates (only update - no new assignments are processed)
56 11,14,17 * * * /home/max/canvas/import/fast-import.cron c24 > /home/max/canvas/import/import.cron.c24.log
55 11,14,17 * * * /home/max/canvas/import/fast-import.cron c23 > /home/max/canvas/import/import.cron.c23.log
54 11,14,17 * * * /home/max/canvas/import/fast-import.cron c22 > /home/max/canvas/import/import.cron.c22.log
53 11,14,17 * * * /home/max/canvas/import/fast-import.cron c21 > /home/max/canvas/import/import.cron.c21.log

# Full prio 1,2,3 (all) import (delete/insert) during night
50 04    * * * /home/max/canvas/import/import.cron c24 3 > /home/max/canvas/import/import.cron.c24.log
50 04    * * * /home/max/canvas/import/import.cron c23 3 > /home/max/canvas/import/import.cron.c23.log
40 04    * * * /home/max/canvas/import/import.cron c22 3 > /home/max/canvas/import/import.cron.c22.log
30 04    * * * /home/max/canvas/import/import.cron c21 3 > /home/max/canvas/import/import.cron.c21.log
20 04    * * * /home/max/canvas/import/import.cron c20 3 > /home/max/canvas/import/import.cron.c20.log
```

--

Canvas API

Example API URLS:

Show all items (pages, quizes, assignemnts, ...) from a module.

> https://.../api/v1/courses/3237/modules/19631/items

Show assignment **110611** from course **7760**
Include user- en submission information *see parameters)

> https://.../api/v1/courses/**7760**/assignments/**110611**
> /submissions?include[]=submission_comments&include[]=user

Zelfde maar nu alleen van user **61196**

> https://.../api/v1/courses/7760/assignments/131589/submissions/**61196**
> ?include[]=submission_comments&include[]=user

Boiler Plate for rating a submission with Python

```
from canvasapi import Canvas


course = canvas.get_course(course_id)
assignment = course.get_assignment(assignment_id)


submissions = assignment.get_submissions(include=["user", "submission_comments"])


for submission in submissions:
  # create code to search for the right submission (don't know how to select one directly...?
  if ( submission.id == ....):
    this_attempt = submission.attempt
    submission.edit(submission={"posted_grade": str(10)})
    submission.edit(comment={"text_comment": 'Well done!', "attempt": this_attempt})
```

xxx

Koppeling beheren

In Canvas heb je een cursus met daarin modules.

Het koppelen van een module aan de Canvas Monitor verloopt in twee stappen:

1. Eerst moet de cursus met Canvas zijn gekoppeld
2. Daarna kan je modules koppelen.

Stap 1 wordt pas effectief na een volledige update (meestal 's nachts).

Stap 2 is direct effectief en de module is direct zichtbaar in de CM.

Cursus koppelen

Je hebt het cursus ID nodig uit Canvas, deze staat in de URL als je de cursus opent.

Bijv: site.server.com/courses/10755 dan is 10755 het Canvas cursus ID.

Ga naar de CM, selecteer Cursus (Blok)

image-1693568528734.png

Druk op de Create Course (groen knop).

image-1693568570480.png

Vul de velden in:

- Id is het eerder genoemde Canvas ID
- Naam is bijvoorbeeld "Blok 3"
- Korte naam is bijvoorbeeld B3
- Positie is niet heel belangrijk omdat de modules later allemaal een eigen positie krijgen.
- Prioriteit, heeft te maken met de hoeveelheid updates. Vul hier standaard 3 in.

Bewaar de cursus en wacht tot de volgende dag naar de totale update.

CMON 2024

Canvas

In het LMS Canvas, bestaan cursussen. Een cursus bestaat uit één of meer module(s). Een module bevat pagina's opdrachten, quizjes, etc.

Cursus (Courses) bestaan uiit:	Module (Modules) bestaan uit:	Opdrachten (assignments)

In ons onderwijssysteem wordt bijna exclusief gebruik gemaakt van opdrachten. Dit maakt het eenvoudiger en een opdracht kan uitleg bevatten. Indien er alleen theorie wordt gepresenteerd is het nog steeds handig om een controleopdracht te geven, zodat de stof beter wordt verwerkt door de studenten.

Blokken en tracks

Een blok is een verzameling modules die een bepaalde onderwijsperiode dient te worden afgerond. De definitie van welke modules in welk blok vallen wordt in de Canvas Monitor gedefinieerd.

Een track is een verzameling modules dit logischerwijs bij elkaar horen en waarin studenten in toenemende mate zelfstandig in kunnen werken.

Voor het cohort 2024 was een cursus en een blok gelijk.

Cancas Cursus	=	Blok

Na cohort 2024 is een cursus gelijk aan een track.

Canvas Cursus	=	Track

Toevoegen Cursus in de Canvas Monitor (CMON).

Als je de beheerprivileges hebt, dan vind je een menu met de naam *beheer*.

Kies onder dit menu de optie *Cursus.* Druk vervolgens op de groene knop bovenaan '*Create Course* '.

Vul de volgende velden in:

Cursus ID, dit is het ID uit Canvas (ga naar de module in Canvas en neem het nummer over uit de url.
Voorbeeld: in https://.....com/courses/12345, is 12345 het *cursus ID*.

Cursusnaam, is de naam zoals die in CMON wordt gebruikt in verschillende overzichten. In principe zou deze (min of meer) moeten overeenkomen met de naam van de cursusnaam in Canvas.

Sort order, geeft de volgorde van tonen van cursussen in bepaalde overzicht aan. Dit gaat alleen om overzichten waarin cursussen worden getoond. Wanneer modules worden getoond dan prevaleert de sorteervolgorde van de module.

Bloknaam, C23 en eerder: deze geeft aan in welk blok deze cursus komt. C24: dit is de cursusnaam (tracknaam) zoals die in een paar overzichten wordt getoond (de indeling per blok wordt bij de module gedaan).

Prio, 1,2,3 of 9: 1 geeft aan dat de module in beginsel zo vaak mogelijk wordt geüpdate, 2 wordt iets minder vaak geüpdate en 3 alleen 's nachts. Een 9 wordt alleen handmatig (op verzoek) geüpdate.

Rapporten

Inleiding

Om rapporten te maken voor studenten is een klein Python script gemaakt.

Zorg ervoor dat Python3 is geïnstalleerd, en dat via PIP de juiste library's zijn geïnstalleerd.

```
pip install pandas
```

Benodigde bestanden

studenten.csv

```
Nr;Klas;Canvas Id;Student nr;Naam
1;3B;20576;2033543;Leonard Cranary
2;3C;26328;2140079;Walid Sallek
...
```

Dit bestand is een export van de Canvasmonitor waarbij niet relevante kolommen zijn weggehaald.

Dit bestand is leidend en de Id's worden gebruikt om een koppeling te maken met andere Excel lijsten.

De kolomnamen kunnen worden gebruikt om teksten in de template te vervangen. Zo kun je wordt _Klas_ in een template vervangen door de klas in dit overzicht.

template.docx

Dit is een Word-template (template.docx) waarbij de in-te-vullen velden worden gekenmerkt door een pre-fix en post fix, bijvoorbeeld:

```
_studentNaam_
```

Ander bestanden

Ander bestanden, presentie, blokken, kennis-check,.....

Afhankelijk van de gegevens die je nodig hebt kan je meerdere bestanden inlezen. Elk bestand dient een ID-kolom te hebben. Dit id moet het studentennummer bevatten dat overeenkomt met

het studentnummer in de *stundenten.csv*.

De bestanden worden ingelezen in het codeblok vanaf regel 51.

Vanaf regel 90 worden de velden uit de template vervangen.

Voorbeeld:

```
replace_text_in_doc(doc, "_kennis_", str(kennis.get(student_number, '???')))
```

In een van de ingelezen bestanden staat een kolom *kennis.* Deze wordt gematched uit kennis.csv via het student_number veld (uit studenten.csv).

Aanpassen in de code

Plaats de docenten en de klassen in de dictionary (regel 12 v/d code).

Code

```
import os, sys
# import chardet
import pandas as pd
from shutil import copyfile
from docx import Document



# Load the Excel file
basis_csv = './studenten.csv'
word_template = './template.docx'
presentie_csv = './presentie.csv'
kennis_csv = './kennis.csv'
canvas_csv = './canvas.csv'
base_output_folder = './'
docent = { '3A':'Mina Ulas - Bicer', '3B':'Muhammed Çagli / Anton Boutkam', '3C':'Max
Bisschop'}




# Define the function to replace text in the document
def replace_text_in_doc(doc, old_text, new_text):
```

```
    for paragraph in doc.paragraphs:
        if old_text in paragraph.text:
            paragraph.text = paragraph.text.replace(old_text, new_text)
    for table in doc.tables:
        for row in table.rows:
            for cell in row.cells:
                replace_text_in_doc(cell, old_text, new_text)



def readExcel(excel_path):
    # with open(excel_path, 'rb') as file:
    #     result = chardet.detect(file.read())
    # print(f'Encoding {excel_path}: '+result['encoding'])
    try:
        excel_data = pd.read_csv(excel_path, index_col=None, sep=';', encoding='utf-8-sig')
        # print("Column names:", [col for col in excel_data.columns])
    except FileNotFoundError:
        sys.exit(f"The file {excel_path} was not found.")
    except pd.errors.EmptyDataError:
        sys.exit(f"The file {excel_path} is empty.")
    except pd.errors.ParserError:
        sys.exit(f"The file {excel_path} is probably not a CSV file or it is badly
formatted.")
    except Exception as e:
        sys.exit(f"An unexpected error occurred in readExcel {excel_path}: {e}")
    return (excel_data)

def getDataFromExcel(file_name, column_name):
    print(f"Reading {column_name} from {file_name}")
    if not os.path.exists(file_name):
        sys.exit(f"File {file_name} not found")

    data = readExcel(file_name)
    if column_name not in data.columns:
        sys.exit(f"Error: Column '{column_name}' does not exist in the file.")
    if 'ID' not in data.columns:
        sys.exit(f"Error: Column ID does not exist in the file.")
```

```
    dict = {}
    for index, row in data.iterrows():
        dict[row['ID']] = row[column_name]
    return(dict)

presentie = getDataFromExcel(presentie_csv, 'presentie')
kennis = getDataFromExcel(kennis_csv, 'kennis')
# praktijk = getDataFromExcel('./canvas.csv', 'praktijk')
blok1 = getDataFromExcel(canvas_csv, 'blok1')
blok2 = getDataFromExcel(canvas_csv, 'blok2')
# studieduur = getDataFromExcel(canvas_csv, 'studieduur')

cmon = readExcel(basis_csv)

for index, row in cmon.iterrows():
    print(f"Processing {row['Naam']}")
    person_name = row['Naam']  # Adjust the column name as necessary
    group_name = row['Klas']  # Adjust the column name as necessary
    student_number = row['Student nr']

    # Create a new file name with prefix "voortgang"
    new_file_name = f"voortgang {student_number} {person_name} {group_name}.docx"

    # Create a path for the new group folder if it doesn't exist
    group_folder_path = os.path.join(base_output_folder, group_name)
    if not os.path.exists(group_folder_path):
        os.makedirs(group_folder_path)

    # Define the full path for the new file within the group folder
    new_file_path = os.path.join(group_folder_path, new_file_name)

    # Copy the template to the new file name in the group folder

    doc = Document(word_template)
    replace_text_in_doc(doc, "_studentnaam_", person_name)
    replace_text_in_doc(doc, "_docentnaam_", str(docent.get(group_name, '???')))
    replace_text_in_doc(doc, "_studentnummer_", str(student_number))
    replace_text_in_doc(doc, "_presentie_", str(presentie.get(student_number, '???')))
    replace_text_in_doc(doc, "_kennis_", str(kennis.get(student_number, '???')))
```

```
    # replace_text_in_doc(doc, "_praktijk_", str(praktijk.get(student_number, '???')))
    replace_text_in_doc(doc, "_blok1_", str(blok1.get(student_number, '???')))
    replace_text_in_doc(doc, "_blok2_", str(blok2.get(student_number, '???')))
    # replace_text_in_doc(doc, "_studieduur_", str(studieduur.get(student_number, '???')))
    doc.save(new_file_path)
    print(f"File created for: {person_name} in group: {group_name}")
    # sys.exit("Test")

print("All files have been created.")
```

Voorbeeld extra databestanden

(bovenstaande code werkt met deze bestanden).template.docx

presentie.csv (uit Eduarte)

```
Deelnemer;% Aanwezig;Nr;ID;presentie
2186011 - Yusuf Aksari;96;1;2186011;96
2132925 - Ibrahim Azarfane;13;2;2132925;13
...
```

canvas.csv (voor voortgang blokken, uit de Canvas monitor)

```
Rank;ID;Klas;Student;Studentnaam;Tot;blok7;blok8;Voortgang;studieduur;1;2;3;4;5;6;7;8;9;10;11;
12;13;14;15;16;17;18;19;20;21;22;23;24;;praktijk;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;
Naam
7350;2184477;3D;Student;Giovanni Deniz;36;Ja;Ja;3653;3-
;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;100;1
00;;0;100;100;100;100;100;0;100;100;13;100;100;100;100;100;40;GiovanniD|/public/index|code|935
315184868c08dc18c67ff3f4d3645
.....
```

kennis.csv (voor kennis-check)

```
ID;Naam;kennis
2202160;Alie Abduka;73,0
2197436;André Romburg;76,5
...
```

Stappen

Let op: Bewaar alle CSV-bestanden in het formaat: *CSV - UTF-8*

studenten.csv

Download uit de Canvas-monitor alles studenten. Menu (beheer - studenten en dan export).
Verander de naam naar *studenten.csv*.
Dit is het basisbestand.

Kolomnamen hoeven niet te worden aangepast.

canvas.scv

Download de voortgang uit de Canvas-monitor.
Noem de kolom met het studentennummer 'ID' en voeg kolommen toe blok1, blok 2, ...Zet in deze kolommen "ja" of "nee" afhankelijk of het blok is gehaald. Pas de code aan afhankelijk welke blokken je wilt afdrukken.

Formule: `=IF(SUM(<verwijzing naar blokken>)=300;"Ja";"Nee")`

presentie.csv

Voeg de presentie bestanden samen tot één CSV en bereken het percentage aanwezig.
Percentage aanwezig is: `=round(aanwezig/(aanwezig+ongeoorloofd+ziek)*100;0)`

kennis.csv

Maak een csv-bestand met ID (studentennummer) en zet het (gemiddelde) cijfer voor de kennis-checks onder de kop *kennis*.

Praktijk test

In 2024/25 nog niet getest.

Test run

Om te testen haal het commentaar uit regel 104. De code stopt na één rapport. Controleer het rapport en indien correct draai alle rapporten door regel 104 weer in commentaar te zetten.

--