Mandatory Assignment 3
Courses: Software Development
Class: DAT18C
Semester: 2019

This is one of the compulsory assignments you need to hand in, and pass, to qualify for 3rd semester exam.

The final product of these mandatory assignments must be presented in front of your teachers by the deadline.

You are allowed to work in groups of 3-4 persons (minimum 3). I do not recommend you to work alone as this assignment is large!

Dat18C hand in date:
Hand in date:

- **29 November 2019, latest at 23:55, on Fronter.**

Hand in info:

- **You must submit a link to your git repo.**
- **In the git repo you will include your source code. Furthermore on the root of the repo you will have a "readme.md" file where you will describe your solution. This description will be brief and will focus on the "Deploy part".**

The assignment:
At our university, each semester students are assigned to courses (mandatory or electives). At the moment the process of presenting and assigning to these courses is not digital. You are to create a web-app that will digitalize this system.

The users of the system:
There are 3 types of users for this system:

- Student. This user will login with the school credentials and can do the following tasks:
  - Search for courses
  - Sign up for a course
  - See a list with the courses he/she signed up for
- Teacher. This user will login with the school credentials and can do the following tasks:
  - Create a new course with all the course information (see Course info section)
  - Edit a course they created or they are teaching with other teachers.

- Administration worker. This user will login with the school credentials and can do the following tasks:
    - See a list of the students that signed up for a specific course. This list will be sorted according to the timestamp of the signed up moment.
    - Approve or reject a signed up request from the student.
- Legacy system. This is an old system from which you can retrieve a list of Teachers and Courses, but note that not all data describing those is enough, your program should handle that. This system exposes all its endpoints, both for what they assume as parameters, what it returns and on which format it responded through swagger.

Let's build it
You are to build this web-app using the Spring Framework. The data produced by this system will be stored on a Database (MySQL or other - this is up to you).

Course info
Each course will contain the following informations. This data is provided by the teacher and the students can only view this data.

- Name of the course in Danish
- Name of the course in English
- Study Program. One or more of the following options:
    - Computer Science
    - Web Development
    - Software development
    - IT-Security
- Mandatory or Elective
- ECTS
- Course language
- Minimum number of students
- Expected number of students
- Maximum number of students
- Prerequisites
- Learning outcome
- Content
- Learning activities
- Exam form
- Teachers

Here is an example of this info for a specific course:

Name of the course (Danish)	Full Stack NodeJs
Name of the course (English)	Full Stack NodeJs
Semester	2
Class code	WD-2020-F-NODEJS
Study Programme	Web Development
Mandatory or Elective	Elective
ECTS	10
Course language	English
Minimum # of students	15
Expected # of students	35
Maximum # of students	50
Prerequisites	Students must know HTML, CSS, JS, PHP and MySQL.
Learning outcome	Students will be able to code a full stack web based application, set-up a NODEJS server in the cloud and decide the best possible use of MongoDB
Content	NodeJS, Flexbox, Grid, CSS, MongoDB, AJAX, Websockets, JSON objects, Setting up a server in Amazon Web Servers and locally. Use of the terminal and FTP. Also, the setup and use of HTTPS.
Learning activities	Individual work and exam. Communication takes place via our Ryver channel WD-2020-F-NODEJS

Exam form	Internal oral exam based on hand in product. Graded based on the 7-scale.
Teachers	Santiago Donoso (sand@kea.dk) Constantin Alexandru Gheorghiasa (coag@kea.dk)

Swagger documentation

The legacy system is deployed to an EC2 instance at AWS, use the following IP-address to see the complete list of endpoints.

IP-address:http://35.159.46.191/swagger-ui.html

Must-have requirements

1. CRUD for course. This includes also a list with the students assigned to this course.
2. Login for each type of the user
3. All the requirements for the different type of user
4. Auto update of the content on pages when need it (jQuery)
5. A user-friendly design.
6. Call against the provided Web-Service to fetch some of the content (Teachers & Courses) use the swagger documentation for the details.
7. When you add/update a new Teacher or Course to your backend, the newly added/updated resource should also be reflected in the legacy system.