Assignment Overview

1DV600 Spring 2016

The purpose of this course is to get an insight into the process of creating computer software. The course will consist of four assignments that will demand quite some effort from you. Each assignment will build on the previous. In this course the focus is on everything around the code in a software project, that is the iterative process itself, documentation and testing. As a bonus you will also learn how a modern web application works.

You will be given a half-finished web application which is the foundation of a library system for books where you can add, modify and delete books. The web application consists of two modules, the first module is a finished front end that you can use to verify that everything works. The second module is only partially finished and it is your job to complete it during this course.

Notice that all assignments are individual!

More information about the web application is available under the sections "Application Stack" and "Installation and Deployment".

Grading

All assignments will be graded on a scale from A to F, where F is failed. You are allowed to improve your work after handing it in via Moodle if you do so before the deadline. DOn't forget to submit your final version. The grade is final, that is -- once it is set, you will not have any opportunities to improve it. We will open up two re-take opportunities where you can re-submit.

We use the **0-4-assessment model** for specific learning goals.
0. Nonexistent
1. Unsatisfactory
2. Sufficient
3. Good
4. Excellent.

Learning goals/criteria is connected to a specific task and relates to one or more learning outcomes in the course syllabus. Examples of learning goals/criteria include:
1. Knowledge and Understanding of a topic.
2. Problem solving Capability
3. Engineering capability
4. Result, Analysis and Interpretation
5. Written and Oral Presentation and Communication.

Submission and Presentation

All text must be handed in PDF format. Send only the code placed in the folders “/src” (Java) or “/app” (Node.js), never build files.

All solutions to the assignments are to be handed in using Moodle. The last assignment, the project, will also be presented orally by you.

Deadline

The deadline for the assignments will always be on Fridays at 23:59. Any assignment handed in after that will be downgraded with 25 percentage points.

Time Log

Each assignment must be accompanied with a time log. This time log should contain the date, time and task to be performed. The reason for doing this is for you to get some experience in estimating your own time -- creating a time log is one of the best ways of doing this.

Time Plan

Each assignment has a date of publication date as well a deadline date.

● Assignment 1 -- publication: w4	deadline: w6
● Assignment 2 -- publication: w6	deadline: w8
● Assignment 3 -- publication: w7	deadline: w9
● Assignment 4 -- publication: w9	deadline: w10

The last assignment can be seen as a project, as it incorporates all the previous parts in one larger assignment.

Application Stack

Because of the complexity inherited in creating software this early in the education, you will be handed a web application with a fully functional front end (client) and a partially implemented back end (server). This application stack is implemented in a modern way and the two modules (the client and the server) will communicate via HTTP requests specified in a RESTful specification that will be given to you.

The application is a book library where you are going to Create, Read, Update and Delete (CRUD) a book. The front end is implemented using a Single Page Application (SPA) architecture, which means that no rendering is done on server side -- all logic for the visual part is done locally and no reload of the page is needed. This also means that the border between back end and front end is very sharp, the only way to know how to communicate is to use the API (the specification is given to you).

The application server is executed in a virtual machine that is managed via Vagrant. More information on installation and deployment is given in the next section.

Installation and Deployment

The following applications are required for the assignment set in this class (**be sure to have the latest versions installed**):

- Vagrant (http://vagrantup.com/) 1.8.1 or greater
- VirtualBox (https://www.virtualbox.org/) 5.0.14 or greater
- Any text editor, for example Atom (https://atom.io/)

Follow the steps below to get started with the application in the assignments:

1. Install VirtualBox and Vagrant from the pages listed above.
2. Open a terminal/command prompt/powershell and navigate to the project folder.
3. Execute **vagrant up** (this will take some time the first time since a Linux distribution is downloaded and installed).
4. Open **http://localhost:9090** in any browser to see the client.
5. In the terminal/command prompt/powershell, press **ctrl-c** twice to stop the process.
6. After doing changes in the code, execute **vagrant reload** (will be faster than initial start).

API Specification

The API specification (in HTML format) is located in the project folder.

Building your own client

It is possible to build another client than the one supplied with the assignments, but this is *not mandatory*. However, if you feel ambitious and curious it is possible to create a new front end. The supplied front end is built using the framework **React.js** and using an architecture called **Flux**. In the project folder you will find a file called **client.zip** which contains instructions on how to build a client in the file **README.md**.