Experiences of Software Development Practicals

# Practice Project

Jeroen Keppens
Software Engineering Group Project (2017/18)

Date: 3–24/10/2017, Place: BH(S) 7.01/2/3

## Introduction

Computer Science is a discipline that spawns new technologies, paradigms and approaches continuously. After you graduate, you will have to continue to learn new technologies. If you become a Software Engineer, you should expect to have to learn at least some new knowledge and skills for many of the projects you undertake. Starting a project fully versed in all of the programming languages, tools, technologies and processes that you could possibly need in your project is a rare luxury rather than the rule, especially when you are starting your career. For that reason, you should not expect the be fully trained in everything you need for your group projects before they start. You will need to so some training yourself. Don't worry about this though: the more experienced you are with learning new Computing technologies, the easier it becomes as you will start to see how the general concepts and principles you have learned apply to many seemingly disparate technologies. This Practice Project is a first opportunity to learn something new as you develop a system.

This practice project will take place during the first 4 weeks of the Software Engineering Group Project Practicals on Tuesdays 9:00-11:00, Tuesdays 11:00-13:00 or Thursdays 16:00-18:00, in October 2017. Your task involves the development of a small webpage using only front-end languages for web development. You will form a team of 3 or 4 people at the start of the first lab, set objectives for your project based on the assignment, organise your training in between lab sessions and develop your webpage during the lab sessions. TAs will be on hand to help you organise your work and point you in the right direction (though they won't solve your technical problems for you!). This project is not assessed but we will keep track of who completed the work.

## Assignment

The objective of this project is to develop a small website that helps third year (level 6) Computer Science and Computer Science with Management students select their module diet. The website must show students the available modules, with a clear indication of how the module fits in the programme (core, compulsory or optional), the assessment structure of module, the module's credit volume, term(s) and lecturer(s), and a short description of the content. The website should enable students to check whether a module diet is valid for their programme. Ideally, it enables students to generate a module diet that is presented in a concise table, which identifies any errors in the module diet. The website could also suggest modules that fit a particular stream or area of expertise students might wish to focus on. Ideally, such suggestions are provided even if the student does not explicitly selects a stream, but has selected one or more modules belonging to a particular stream. The website must **not** rely on a backend and run entirely within a web browser.

## Technologies and Training

The assignment stipulates explicitly that you must develop a web application using only front-end languages. There are no constraints on which languages you use and it is up to each team to decide for itself which languages you are going to use. If you have absolutely no experience with web development whatsoever, there are three languages you should consider to use:

- HTML (hypertext markup language) is a language that marks up documents with tags indicating the roles of elements in a document (e.g. title, paragraph, table, etc.). Although it is possible to use WYSIWYG editors to generate HTML, this project is going to require a little more finesse. If you have never written any HTML before, you will need to learn HTML. Don't worry, the language is easy.

- CSS (cascading style sheets) is a language that specifies how elements in an HTML page are displayed. Normally, you will be producing CSS files alongside your HTML pages so that layout and content are separated. It is a good idea to do that right from the start. You can use CSS files produced by others. However, it is very likely that you will want to tweak the layout of your pages a little so having some knowledge of CSS will be helpful.

- Javascript (or ECMAScript) is a programming language that enables you to make your pages more dynamic. For example, Javascript allows you to get your pages to respond to events that take place within the page. Once you try to implement the "should have" requirements of the assignment, you will need Javascript. Beware, although Javascript looks very similar to Java and has the word "Java" in its name, Javascript is not Java and there are some very significant differences between Java and Javascript.

You do not have to use these languages. For example, you could build a Java Applet that meets all the requirements for this project. However, you will probably learn more if your solution is based on HTML, CSS and Javascript.

You may use any extensions and libraries you wish for these project (as long as you respect any legal restrictions imposed on them). Front-end component libraries, such as Bootstrap, Foundation and Skeleton, enable you to develop webpages with a modern look and feel. These very accessible and require very little technical knowledge. Teams with some background knowledge in web development may wish to use more advanced tools, such as jQuery, AngularJS or React. This is fine as long as every member of your team is happy with that.

Links to training materials are provided on the KEATS page under the Practice Project topic.

## Outcome

You will be deemed to have completed this project successfully if (i) you have produced a webpage using HTML, CSS and Javascript that meets the "must-have" and some of the other requirements, (ii) the webpage was developed by a team of about 3–4 people, with every member of the team making a significant technical (i.e. HTML, CSS and/or Javascript code) contribution to the work, and (iii) you have attended at least three out of four of lab practical sessions for the full 2 hours and worked on the project throughout these lab sessions. Note that it is crucial that you contribute to the code and allow others to contribute to the code as well. A website that meets the requirements but was developed by only one member of the team with others merely watching but not contributing will not be treated as a successful outcome (not even for the person who did all the coding).

If you complete this project successfully, you will be awarded a badge on KEATS. The award of this badge has *no* impact on your module mark or the team you will be assigned to in future projects.

## Etiquette

In this practice project, you are required to work in a team. Therefore, your behaviour will affect the learning experience of others in the project. You should be considerate to others and try ensure everyone has a chance to get the most out of this project. There are a few behaviours that really frustrate teams and yet remain quite common in the years I have taught this module. It is very important that avoid these:

- *Don't be late. Don't leave early.* You cannot participate fully with the project if you're not there to contribute. Worse still, if you miss important discussions, you are creating work for others as they will have to inform you of what was discussed. If you are going to be late or have to leave early due to unavoidable circumstances beyond your control, inform your team as soon as possible, explain your circumstances and make an extra effort to compensate for what you missed.

- *Make an effort.* It's ok to fail at completing tasks and to find some tasks very difficult or impossible. After all, you are here to learn. But if you don't put in sufficient effort, then you are setting yourself up for failure. If you are struggling with work, do not suffer in silence. Tell/show your team what you are doing to try and succeed. Ask for help if you need it. If others help you out, watch closely how they solve problems and learn from them so that you become a better computer scientist.

- *Show some enthusiasm and be friendly.* Good teams tend to create a pleasant, friendly atmosphere where everyone is enjoying what they are doing. Of course, although it is good to have fun, don't exaggerate. Be sensitive that there is work to do and incessant joking around can get in the way of work.

- *Don't be shy, express your views.* Your team will want to know you are on board with the team's decisions and whether you have any questions or concerns. Moreover, you may have interesting ideas and important issues to raise.

- *Listen to others*. People will be more productive if their opinions are valued and the concerns are heard.