Web Information Systems: Project Assignment (2013 - 2014)

Prof. Dr. Beat Signer
Assistant: Reinout Roels (rroels@vub.ac.be)

1. Goal

The goal of this project is to develop a Web 2.0 application centered around a topic of your choice. The ultimate goal of your website is to allowing users to view, search, create, manage and share information. This information is created by users themselves but it should also be retrieved from existing Web Services. Furthermore, the user-generated data should be published, either via a Web Service or by injecting a machine-readable version into web pages. Aside from data, functionality offered by Web Services can be reused as well. Social media networks such as Facebook and Twitter can be plugged in and RIA-style interfaces (using interactive maps, AJAX and HTML5) can be developed to interactively work with information and enable user collaboration. Note that users will often visit the web application using their mobile device. Therefore, care should be taken that the web application is usable on a mobile device. All the required concepts and technologies will be mentioned and explained during this semester's lectures and exercise sessions.

The assignment is carried out in teams of 3 people. You are free to use any language or framework for web development. However, you are not allowed to use content management systems (CMS) where you can graphically click a site together (e.g. Wordpress, Drupal or Joomla). Please consult the rest of this document for further details.

2. Choosing a Topic

You are free to choose any topic for your application. However, we expect you to implement a minimum of functionality and require the usage of certain technologies. These will be detailed in Sections 3 and 4 respectively. We suggest that you read these requirements carefully before deciding on a topic. To give you an idea of what we are expecting we will briefly describe the main idea behind two projects from previous years. Note that there are just examples and we prefer that you come up with your own topic.

- MovieMaps: This project was centered around movies and their filming locations. The system allows users to search, view or create profiles for movies. This information is augmented with information provided from Web Services such as IMDb or Rotten Tomatoes (e.g. scores, genres, images or trailers). When a user recognises a filming location from the movie they can go to its profile and enter that location via coordinates or by marking it on a Google Map or Streetview. Users could also graphically connect locations to form tours. These tours could then in turn be searched, shared and rated.

- WISE-Fit: This project was centered around sports and fitness. Users were able to search, view, create and share workout sessions (combining exercises) for various sports. Web Services were used for retrieving exercise information but are also used for functionality such as BMI calculators, calorie counting or instructional movies. Users could also update their progress (e.g. the amount of kilometers run or amount of repetitions achieved) which was then visualised in nice graphs (which can be shared). Finally, users can also search and mark interesting locations on a map (e.g. nice routes for running or a good gym).

When coming up with your own topic, we encourage you to be creative and add interesting features to your project. Original functionality (or lack thereof) will influence your final grade!

3. Functional Requirements

Regardless of the topic you choose, we ask you to implement some basic functionality.

- Users need to register to be able to use your application. To do so they are required to fill in a form and enter details such as a username, a password and an email address. After registering a user can log in with the specified credentials and log out when they want to.
- Each user has a basic profile where they see an overview of their account details.
- A user should be able to view, search, create and modify the information at the core of your application. For example, whether your website is about movies, games, restaurants or fitness exercises, the user should be able to:
    - List them, for instance in a list or a grid.
    - Search them based on different fields (e.g. title, name, author, genre or date).
    - Create them when they are not already in the system.
    - Modify them when a mistake is present or for adding additional details.
- As the project is a Web 2.0 application the social aspect should be clearly present in your application. Other than just sharing via social media networks, it should also be possible to collaborate and gather knowledge as a group (crowdsourcing). How you fulfil this requirement is up to you but examples include commenting, rating, tagging and other forms of collaborative content creation. As another example you could allow your system to set up group activities with other users on your site (e.g. a joint fitness session or a movie night with your friends).

Important note: these requirements are intentionally vague, so you can realise your own ideas on how to achieve the main goal (when in doubt, you can always contact us to make sure the envisioned web application is sufficient). **Do not limit yourself to a minimal set of functionality**! As mentioned, this is an important part of the project; if you have only a minimal realisation, you will not be able to receive the maximum grades since we expect some creative ideas from your side.

4. Technical Requirements

- **AJAX** should be used in certain places in the website to increase the interactivity and responsiveness: for instance, checking the availability of a username during registration, or having an autocomplete function when searching. Note that it is not necessary to use AJAX everywhere; only in places where it most increases the user experience.
- **Form validity** For each web form, (at least) a server-side validity check should be provided. In case incorrect data was entered, the user is redirected to the form where the incorrect fields are annotated and information that was entered correctly is already filled in.
- **CSS** The website should also be visually attractive to the user. To ensure consistency of layout and presentation across the website pages, Cascading Style Sheets (CSS) should be used.
- **HTML5** supplies a range of often-desired functionality missing in the current HTML version (e.g. 2D drawing, local storage, etc.) and together with CSS3 this allows you to make very powerful, interactive, visually attractive websites. Therefore, students are encouraged to use HTML5 in their website. We require that you use at least one feature specific to HTML5.
- **Web Services** Your application should use at least one external web service for retrieving information (using SOAP or REST). Examples include the retrieval of the local weather in real time or retrieving images for specific objects on demand (e.g. getting the poster image for a movie when a user navigates to its overview).
- **Publish Content** User-generated data (e.g. new exercises, movies, reviews, etc.) should in turn be published, either separately via a web service (using SOAP or REST), or embedded in the web pages via machine-readable semantic annotations (e.g. RDFa).
- **Mobility** Since it would be very interesting to use this website in mobile settings (e.g. to follow up on training schedules), developing a dedicated, fully-fledged mobile web site (i.e. fine-tuned towards mobile devices) could be a focus of your project. Note that this requirement can also be met in a minimal way. For instance, care can be taken that single pages do not contain too much content, so mobile users can get a reasonable overview of the webpage without having to constantly zoom in / out. Another possibility is to tweak your existing stylesheets towards mobile usage.
- **Google Maps** Finally we require you to use the Google Maps API[1] in a meaningful way. In its most simple form it can be used for the simple visualisation of a location on the map. However, this can be extended to allow the user to place markers on the map, for geocoding or for the creation of routes and paths.

[1] https://developers.google.com/maps/documentation/javascript/

During the exercise sessions you will get familiar with some concepts, frameworks and libraries that can be used to fulfil these requirements. However, you are completely free to use any other programming language, framework or library that will help you with the implementation. To keep the project fair you are not allowed to use content management systems (CMS) where you can graphically click a fully functional site together (e.g. Wordpress, Drupal or Joomla). You are allowed to use graphical tools for the front-end (HTML and CSS) but we require you to implement the logic behind it yourself. In Appendix A you will find a list of useful frameworks and libraries. When in doubt whether a particular framework or library can be used, feel free to contact us.

5. Project organisation

Students will be asked to form teams of three persons each. Please send your team info (members + team name) to rroels@vub.ac.be **before** October 4th. If you do not provide a team name you will be given one. Each team will have to give two presentations about the project: one in the middle of the semester and one at the end. The first presentation will discuss your general approach to the project, while the second one will consist of a demo and a short overview of the web application architecture. In the first presentation, be sure to clearly outline the functionality you will support in your web application! You will receive more information on these presentations later on. You can find the exact dates on the PointCarré webpage (under Agenda).

At the end of the semester, the project is handed in using the "Assignment" feature of PointCarré. This can be found under the Tools section of the main WIS PointCarré page. Here, you will upload a single zip-file called WIS-[team_name].zip (e.g. WIS-Team_42.zip). The zip file should contain the following:

1. The source code of the web application, together with detailed and **complete** instructions on how to run it (i.e. step for step instructions together with possible dependencies, required software, etc.). Note that this source code should be **documented**: every important method and class should be annotated with a description of its goal and functionality.

2. A report describing your application. This report summarises how you realised each of the required functions, possibly with a motivation on why you made certain implementation choices. This report should be **complete**: it should contain an overview of the web application architecture and separate sections for each of the implemented functions (**max. # pages: 20**). If certain functionality is not mentioned in the report we might not know that it exists which in turn might result in a loss of points.

3. For the evaluation of your project we also require you to provide <u>one</u> of the following:
    - A link to a working version of your website (with username/password if needed)
    - A sequence of (annotated) screenshots that demonstrate all the implemented functionality
    - A short movie where you give a tour of your website

Appendix A: Useful Frameworks and Libraries

Server Side:

As mentioned previously you are allowed to use any programming language for this assignment. This includes popular languages such as PHP or Java Servlets/JSP (http://tomcat.apache.org/). However, we would recommend that you also make use of a framework for web development. These frameworks provide you with helpful functionality that will help you in fulfilling the requirements. This includes functionality such as user and session management, database connectivity, object persistence, security, form validation and much more. While all this can be achieved by implementing it yourself, we recommend that you use a framework for this so that you can focus more on the implementation of interesting topic-related functionality. We briefly list some of the more popular frameworks for some different programming languages.

Java	**Scala**
Struts - http://struts.apache.org/	Lift - http://liftweb.net/
Tapestry - http://tapestry.apache.org/	**Groovy**
Spring - projects.spring.io/spring-webflow/	Grails - http://grails.org/
JavaScript	**Ruby**
Node.js - http://nodejs.org/	Ruby on Rails - http://rubyonrails.org/
Express - http://expressjs.com/	Sinatra - http://www.sinatrarb.com/
Ember - http://emberjs.com/	**PHP**
Python	CodeIgniter - http://ellislab.com/codeigniter
Django - https://www.djangoproject.com/	CakePHP - http://cakephp.org/
.NET	Yii - http://www.yiiframework.com/
MVC - http://www.asp.net/mvc/mvc4	Zend - http://www.zend.com/en/
Silverlight - http://microsoft.com/silverlight/	

See http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks for a more complete overview.

Client Side:

jQuery is a de facto standard as a JavaScript library. Its most important uses are accessing and modifying the DOM tree and AJAX requests. http://jquery.com/

jQuery UI is an extension of jQuery that focuses on user interfaces for RIAs. It provides you with ready-to-use UI elements and abstracts functionality such as drag-and-drop. http://jqueryui.com/

CSS Preprocessors:

As a programmer you might get the feeling that standard CSS is fairly limited (e.g. no nesting or no variables/functions). For this reason CSS preprocessors were made. These provide a more useful CSS syntax and compile this to standard CSS at runtime (comparable to macros in languages such as C).

LESS - http://lesscss.org/
Sass - http://sass-lang.com/
Stylus - http://learnboost.github.io/stylus/