COMP3001: Team K

Mavrommatis, Marinos
mm1g10@ecs.soton.ac.uk

Petroaica, Alex
first2.last2@xxxxx.com

Karkallis, Panicos
pk1g10@ecs.soton.ac.uk

Perez-Mavrogenis, Dionisio
dpm3g10@ecs.soton.ac.uk

Zardilis, Argyris
az2g10@ecs.soton.ac.uk

Svensson, Kim
ks6g10@ecs.soton.ac.uk

Oikonomou, Alexandros
ao2g10@ecs.soton.ac.uk

January 9, 2013

1 Description of prototype functionality

1.1 Background

The Internet, since its inception, has been a basis for information and knowledge sharing. With the advent of the Web in the last two decades it has also become increasingly interactive. This increasing amount of interactivity started out with Usenet newsgroups, the IRC and discussion groups which were (and still are) very popular especially among technology people. The ease of use of web tools and the widespread adoption of the PC and the World Wide Web has made the average user able to interact easily with others. In the last decade, users have evolved from mere consumers of static documents to publishers/consumers of dynamic information, through the use of blogs and social networking sites. This led to the creation of themed online-communities allowing people to interact and share knowledge on just about any topic. Especially popular though are technology related communities such as discussion forums for a specific topic (Ubuntu Users) or question-answer based websites (Stack Overflow).

We have found, through our university experience, that we rely heavily on user-generated information sharing websites such as Wikipedia, blogs or question-answer websites(which often involve discussion) for our work. Also the use of social networking sites for communication(primarily) and information sharing in the University environment - either among group members for a specific coursework, students in a specific course or society- has become a de facto standard. Our aim was the creation of a web application targeted at students of our department (ECS) that combined characteristics of question-answer based websites(best answer, user reputation), the interactivity offered by Facebook groups for communication and information sharing and also some form of module exploration (module-specific information that do not show up at the Syllabus) for lower-year students that would assist in the module selection process.

1.2 Prototype

Our application tries to bridge the gap between information sharing websites and Facebook groups by providing a way to do both through a forum-like interface. Registered users can subscribe to different modules to see module-specific posts. They can also create their own threads to discuss issues and ask questions, or reply to threads created by other users. Replies can be up-voted, down-voted by users and accepted as answers by the thread creator. The number of a user's accepted answers is shown next to their name when replying, as a form of reward/motivation.

The functionality of the application is extended by prompting users to anonymously contribute to module specific statistics, instantly displayed to all users. In particular, users can contribute their mark on specific assessments and rate assignments and lecturers. These are usually collected towards the end of the semester and are mostly important to prospective students.

Furthermore, threre is a search utility for threads and an administrator section. Users can search for threads by matching words in the thread title, the thread tag list and the poster's username and fullname. Administrators can edit user/module information, delete threads and make other users administrators.

2 Tools and techniques used

Google App Engine SDK
Used for development and testing of the application before the code was uploaded.

Mozilla Firebug
We used Mozilla Firefox, Google Chrome and Opera for our development.

Mozilla Firebug+Chrome equivalent
Used for testing and debugging Javascript.

Git
Git is a distributed version control and source-code management system. Developing with version control has many advantages especially in collaborative projects. It allows for tracking of changes per user, reverting back to previous versions, branching to test experimental features.

Github
Github is a free and open-source web-based project hosting service that uses the Git version control system. It provides a nice graphical interface to git repositories (diffs, repository history, changes by user, relevant statistics). It also provides hosting so it can be used as a central copy of the repository.

Facebook
A facebook group was created for internal communication and discussion between members of the group.

Google Docs
Google docs were used for more formal and persistent communication of ideas, tasks to be done and requirements. It allows for collaborative live editing, a feature that proved very useful.

Last but not least, a python framework was used to aid the completion of this project. The framework used is webapp2, which comes bundled with the Google Appengine SDK. It was chosen, among other python frameworks, because it is very lightweight, it served its purpose (serving both GET and POST requests, having session control, etc.) and was very well documented by Google. A lot of the functionality is carried out by performing AJAX[1] requests, in an effort to provide a better, smoother user experience. Finally, the content was presented to the users via templates, using the jinja templating engine.

As far as techniques are concerned we used the Agile method for software development. The requirements for the forum were changing constantly, because of difficulties balancing between restrictions of the project (lines of code) and functionality. Members of the team were assigned tasks usually in pairs, with the aim that in each task one member would do the python back-end work and the other the javascript and misc front-end work. This property would change when the pair would be assigned a different task. Finally, new tasks were usually revisions of the same functionality, thus peer code reviewing was achieved and in many cases improved the quality of code.

[1] A technique in Javascript programming for refreshing parts of the web-pagedynamically, as opposed to refreshing the whole web-page

3 Relevant statistics

Language/File type	Non-blank lines
Python	1304
Javascript	527
css	579
config	45
html	1292
Total	**3747**

Table 1: Lines of code for project per language

Code imported from external sources (libaries):

jQuery
: Providing an abstraction layer above Javascript functionality , abstracting away browser specific details or problems (e.g. performing AJAX in Mozilla Firefox and Microsoft IE requires different syntax) and providing a faster and simpler selection syntax, allowing us to focus more on the problem than implementation details.

jQuery UI
: Library build on top of jQuery that offers more advanced features, interactive widgets and effects.

jQuery.ScrollTo
: Plugin to jQuery for scrolling elements.

4 Overview of design and implementation

Our main design philosophy was to create an easy to use forum-like interface that also has the characteristics of other online communities or question-answer websites with the notion of measuring the importance or usefulness of answers based on user ratings. It also had to be tailored to the needs of students of our department by providing user generated module information. We felt that these two requirements were what had to be the starting point for all our design decisions as these were the features that would set our application apart from alternatives and also make it attractive to students. The main design decisions regarded the back-end, namely the database and what was going to be stored and how to suit our needs, and the front-end, how the informations was going to be presented to the user in a nice way.

For the front-end part we took ideas from other online-communities websites which had the notion of importance of posts based on user ratings like 'reddit' or 'stack overflow'. We wanted to emphasize the interactivity and discussion aspect of each topic or thread raised so we decided that users can either reply to a specific thread/question directly or reply to another user's reply. Replying to another reply indents the post thus creating sub-threads or sub-discussions which are easily recognizable from the presentation. Also to accommodate the requirement of post rating, we have allowed for replies to be up-voted or down-voted by users which provides confidence in the correctness of a reply. For our other goal, that of module exploration, the user-generated statistics for each module are shown on top of each module specific page along with traditional module information like lecturers and coursework deadline information. Having to provide a good user experience and aid interactivity, our application uses AJAX extensively (for voting on posts or replying to threads, etc.), in order for the users not to have to refresh the page to see the changes-this is were our decision to use jQuery paid of, as DOM selectors and AJAX are performed by jQuery-specific syntax rather than browser-specific syntax (Microsoft IE requires a different syntax than all other browsers).

For the back-end we used the Datastore facilities as provided by Google App Engine to accommodate all the necessary information and entities interactions. We again looked at our main requirements to shape the design of the database and also drawn from our personal experience or what we thought might be useful. The main challenge here was to keep track of the reply hierarchy needed for the correct indented presentation of the threads.

Even though the Datastore is not a traditional database, we have taken advantage of a lot of its more advanced features (like inheritance, references to other entities, amongst the most important) in order to try and make an underlying model that would be decoupled, consistent, be normalised according to the normal forms and have as little redundancy as possible. For example, for keeping track of a user's posts, we have a user entity and a post entity, and the post entity holds a variable for identifying the user who made the post.

5 Critical evaluation

We feel that we have fulfilled the main requirements that we have set at the start of the project. The application provides a nice and intuitive user interface for discussion incorporating the required characteristics such as replies ratings and user reputation. These concepts and the way that they were incorporated in the interface will undoubtedly be familiar to many of the potential users making the learning and adjustment process easier. Also the user-generated information in the form of statistics for each module along with topics and replies could prove to be very valuable for current and most importantly future students. From our experience we have observed that peer learning is one of the best ways to learn at the university level and we believe that this system with user reputation is exactly what is needed to give an extra push to people to help out and the nice to the eye reply-hierarchy layout will improve interactivity and spark discussions.

However as with all user-centric websites where almost all of the content is user-generated the main ingredient for their success is a strong user-base. An important first step is to attract users who might be using other means of communication and although we believe that having targeted the application to the needs of students of our department thus making it better for this purpose, it will still be hard to move from other technologies which have become prevalent.

Because the application developed is at a prototype level, some functionality addition will be needed to reach production quality, for example a more graceful error handling mechanism. Furthermore, a security analysis should be carried out to protect the users from cross-site-scripting and other attacks. Making the application hard to compromise was not only outside of our time schedule as it is a lengthy process, but outside of the scope of the Scripting Demo Prototype Competition. Finally, as it is for all prototype projects, the efficiency of the website was not the main concern. In particular, query optimisation and other techniques such as cookies and careful cashing are currently lacking.