Ivan Barria

CSC 4790

Dr. Daniel Joyce

12/17/2012

Post Mortem

The overall goal of the project was to create a homework software for tutors that would ease the task of assigning and correcting homework. The software gives the tutors the ability to bundle questions into exams and then assign the exams to particular students. Through the use of jaccardian similarity and memory based collaborative filtering the software is able to provide the tutors with a list of recommended questions for each individual student. The software provides dedicated interfaces for the tutor and students and it is accessible through a web browser.

The first stage of the software was to get the technology up on a cloud. This was very easy to do with heroku. Next I had to figure out a way to organize the data in a way that could adapt to many different types of questions and a dynamic categorization scheme. This took me a while and was changing at all times but at the end I came up with a solution that worked. Other parts of projects involved making all the technology work together. The gems, plugins, and javascript. The last part was to build the UI and style it, this was very tedious and at the end I ended up using a CSS template.

The development of the software was anything but straightforward. The direction kept changing from the beginning until the very end. This was my first big software project and I have learned a lot from it. Most importantly I should’ve had a more resolute idea of the what the project was and what problems it solved and focused on it. Instead, the project had an infinite list of features that kept on increasing every day. This in part was due to the fact that the project was build for a startup company and there were a lot of direction changes. I decided to call it “That homework software” (THS) because the direction kept changing so much that I never had a good short name to describe what the software did.

At this point the core of the software is done and it is completely functional. However several minor features/aspects are missing. The next step of the project is to implement more rigorous access control so that students can’t access each others assignments or stats through URL manipulation. Beyond that there are several UI characteristics which could be improved to make software seem more modern. Extra features like a chat or comments for questions could also be implemented. et the technology up on a cloud. This was very easy to do with heroku. Next, I had to figure out a way to organize the data in a way that could adapt to many different types of questions and a dynamic categorization scheme. This took me a while and was changing at all times but at the end I came up with a solution that worked. Other parts of projects involved making all the technology work together. The gems, plugins, and javascript. The last part was to build the UI and style it, this was very tedious and at the end I ended up using a CSS template.

If I would’ve done something differently it would definitely be the management aspect of the project. The project should’ve had clear specifications requirements for a minimum viable product and the direction shouldn’t have changed until that was done. Features can always be added later. Also features like the design should’ve been delayed until all the functionality was done. The largest single waste of time in the project was adapting a design that was build from a marketing point of view. The design was beautiful and contained a lot of features but these hadn’t been thought through and the great majority were only for looks. For example, we had a dashboard that had a feed similar to facebook's wall; the problem was that no one knew what information was going to populate that feed. Having to stall the development of the core of the software to work on features just so that they could fit into a design that had no backend architecture behind it was not efficient or smart.