Webapps Project 2014: Final Report

Alan Vey, Octavian Tuchila, Sean Naderi

June 11, 2014

1 Introduction

This is an *introductory comment*. LaTeX can typeset some very complex documents, but it is also **quite easy to get started**. Try playing around with this file and see. Don't worry that this page looks very spaced out, LaTeX arranges the page for you (much less work to do than in Word!) so if there was more content it would close up the gaps.

To start a new paragraph just leave a blank line. If you do not like numbered sections, use the section*{...} environment instead.

1.1 Some Code

```
The verbatim environment outputs the source without changing
it in any way.
      This
            includes line breaks
   and indentation.
It is useful to reproduce code snippets.
```

To include maths formulas in text put them between $ symbols like this $f(x) = x \times 5$. Or to display a formula on a line on its own you do this:

$$g(y) = y^2$$

2 Project Management

2.1 Group Structure

Alan was the group leader and in charge of the planning of the assignment. He also was the primary developer and so set up the project, wrote the code for all functionality of the site and sorted out all research into third party tools and deployment options. He also set up the testing infrastructure for the application.

In the beginning stages, Octavian was in charge of developing the file system to be used for the site but was met with several difficulties and so after looking into the problems, Alan realised that we could incorporate EtherPad for collaborative document editing, which handles storage itself. Thus, Octavian was assigned to the design of the entire website.

Sean was put in charge of adding additional tests for new features and design elements. Sean used our project to plan our presentation. His feedback was useful in hashing out the small details of the site for both functionality and design.

2.2 Implementation Language

We decided to use the Ruby on Rails framework to develop our application. This meant between us we needed knowledge of Ruby, JavaScript/CoffeeScript, HTML and embedding, CSS and of course the Rails framework itself. Domain specific languages such as RSpec, Capybara and Factorygirl also played a role. Limited knowledge of SQL aided some debugging issues we had relating to the database.

All three of us worked on the WACC complier project together which we implemented in CoffeeScript and JavaScript giving us a solid foundation in these languages. The Ruby on Rails exercise had taught us the basics of web development so combined with the fact that none of us had any prior experience creating web based applications at least we has a good understanding of these languages. This was one of the main reasons for choosing Rails.

We all found it really useful learning Haskell, Java and C when we first commenced our studies as these are all strongly typed languages and have a ridged development style which lead to a very solid understanding of programming. We wanted to apply the same principal to web applications by learning to develop with a framework that has a ridged set of development principals before using tools which allow for more freedom.

Considering most of our programming knowledge is in the three languages mentioned above, we though expanding our understanding of dynamic ones would greatly benefit us making Ruby and JavaScript/CoffeeScript ideal. The process of embedding both in HTML and making use of the reflective nature of Ruby has been really interesting.

A start up Alan is working on in the summer makes use of this framework and so he was especially keen on finding out more about it.

2.3   Design Process

Our division of labour lead to an efficient design process. There was no code duplication and since our responsibilities allowed for little overlap we did not have to spend much time combining the different elements.

Initially, Alan considered using pivotal tracker to plan out our project, however this seemed to offer more functionality than required and would lead to too much complexity in setting up the project so we settled on Git, Gitlab milestones/issues and our website itself. Communication was taken care of by having regular meetings planned by Alan as well as a Facebook group chat and FaceTime.

We used Git as a backup and collaborative development tool. Alan set up the project and got a solid foundation implemented, tested and pushed to our master branch. From this point onwards any new functionality Alan added was done on a separate branch and merged in upon completion. Octavian would branch every time new functionality was added to manage the design of that part of the newly added components and merge with master when he was done. Sean would await the completion of functionality at which point he branched off and conducted modular testing of the models, merging back in when done and branch off again when the design was done to conduct integration testing again merging his work back to master upon completion.

At the beginning of the project we thought of the milestones that were required and set them up on Gitlab. We subsequently added issues when they arose or we started working on a new section, assigning them to whoever we felt was best equipped to deal with them and allocating them to a milestone. This was very useful to know exactly who was working on what and making sure any problems a team member discovered were addressed by the appropriate party.

During the feedback for our Milestone 1 review, Mark suggested we use our site to plan a part of this project. We really liked the idea and decided to plan the presentation with it writing up all supporting documentation on our collaborative real-time software. This was not only useful for the organisation of the presentation but allowed us to find minor bugs and improvements for our site.

We usually met every two to three days to make sure the project was on track and do any of the additional tasks as well as help each other out with what we had learned. Sean would record meeting minutes and send everyone a copy on our Facebook group if required. Octavian took a keen interest in the legal aspects and helped Sean and Alan gain a better understanding after some research he had conducted. The meetings were planned using a Facebook group chat which also served as a useful tool for sharing links we found important and helping each other out with any issues we came across that we did not need to meet for.

All in all we feel this project has been a success as we have overcome many difficulties we encountered in previous tasks we completed together such as code duplication, missing deadlines, under performing and not helping each other adequately. We will definitely be planning future tasks in much more detail as we have done with this one and have all learned much about all the software we used to facilitate this.

3 Program Description

4 Legal Aspects

5 Conclusion

References

[1] Leslie Lamport, *LaTeX: A Document Preparation System.* Addison Wesley, Massachusetts, 2nd Edition, 1994.