WEBSCRP – FINAL REPORT

Lifecycle and Scheduling by 663652

Lifecycle

The expectation of the project to follow a cyclic pattern of development with smaller iterations proved somewhat true. Many areas of the program were redone repeatedly with improvements or changes to the design as new methods and research was found.

One difference to the expected cyclic pattern was the scale of iterations. The traditional expectation is that subsequent designs after the initial will be smaller and require less time. However, in some cases my initial implementation only highlighted the need either a different approach, or a more thorough one. An example of this is the product basket. By analysing the git commit log it is possible to see how this feature was changed over time, and how a large quantity of the code is replaced or removed through the course of development.

Date	Basket Functionality	Code Change (lines)
01/02	• Client creates and stores a 'Basket ID' cookie. • This 'Basket ID' is a key in a MySQL table containing all baskets stored in the system. • 'Add to Basket' function adds the 'Product ID' to the database's 'BASKETS' table along with the 'Basket ID' field.	+153 additions -3 deletions
07/02	• Refactor. • Basket preview now shown in the menu. • Full basket can be viewed by clicking the basket preview. • These functions both query the database for the current list of products matching the 'Basket ID'.	+312 additions -172 deletions
10/02	• Basket now supports quantities of items. • Separate concerns into multiple files.	+87 additions -51 deletions
15/02	• Store basket in local storage instead of cookies. ○ Saving bandwidth and server resources. • Fetch product details from server on basket page only. • Cache preview prices and calculate totals locally.	+54 additions -214 deletions
16/02	• Detect duplicate item additions and update quantities. • Allow removal of items from the basket via basket page.	+45 additions -13 deletions

Modifying some features can also cause knock-on effects in other areas of the application. In rewriting the API, all of the database querying processes had to be updated to be functional again as well. This meant some changes negatively affected current progress in the application.

Therefore the true lifecycle for this project would still have been a cyclic one, but not necessarily in the form of a spiral. Given the fluctuation in time requirement, and the nature of some changes negatively effecting functionality – the graphical representation of the lifecycle may have circular iterations of *increasing* diameter at points before reaching the expected spiral shape.

Scheduling

The initial (and updated) plans served as good checklists for monitoring progress and ensuring the project would be finished within deadlines. However, progress often occurred more sporadically than planned, and I often completed the tasks in a different order to my plan based on what was recently covered in lectures, or what I felt would aid in the creation of additional features.

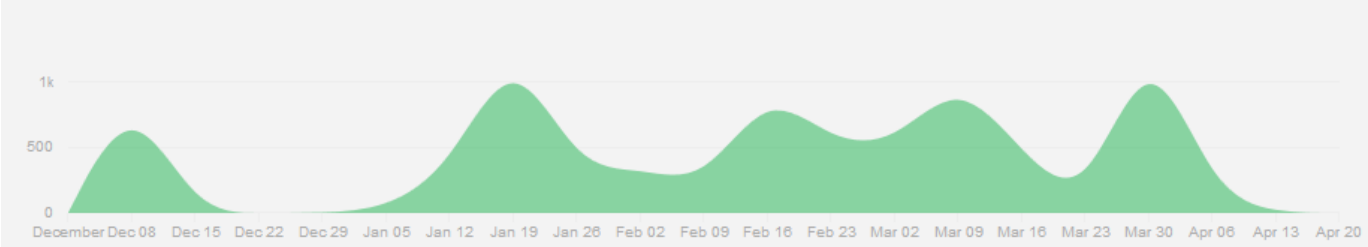
Where a task would have four days assigned, most progress would occur in one stretch instead of spread across the time. Better sub-division of a task prior to starting it would allow me to schedule it better; as well as allowing verification of all its necessary features.

The time I was able to devote to the project was somewhat unpredictable, due to nature of some other university units and any unexpected distractions; so I attempted to get ahead of the schedule where possible, allowing myself more slack in the future. I failed to capitalise on this additional time, and should have planned work on additional functionality.

Towards the end of the available project time, I may have prioritised too much of my time to the product itself instead the documentation, which could have been improved. This is somewhat due to my lack of in-class documentation being written alongside the code. The lesson learnt here was to comment and document code as it is being written, instead of afterwards when some aspects are not understood quite as well.

Total hours spent on the project were not logged, which would have been useful for building experience, as well as for future planning knowledge. An estimate based on around 10 hours a week, which seems roughly accurate, would yield a total time of 200 hours. A high proportion of this time was spent experimenting rather than developing, as I had to discover methods for approaching many of the features.

The chart below tracks the rate of code additions over the full course of the project.



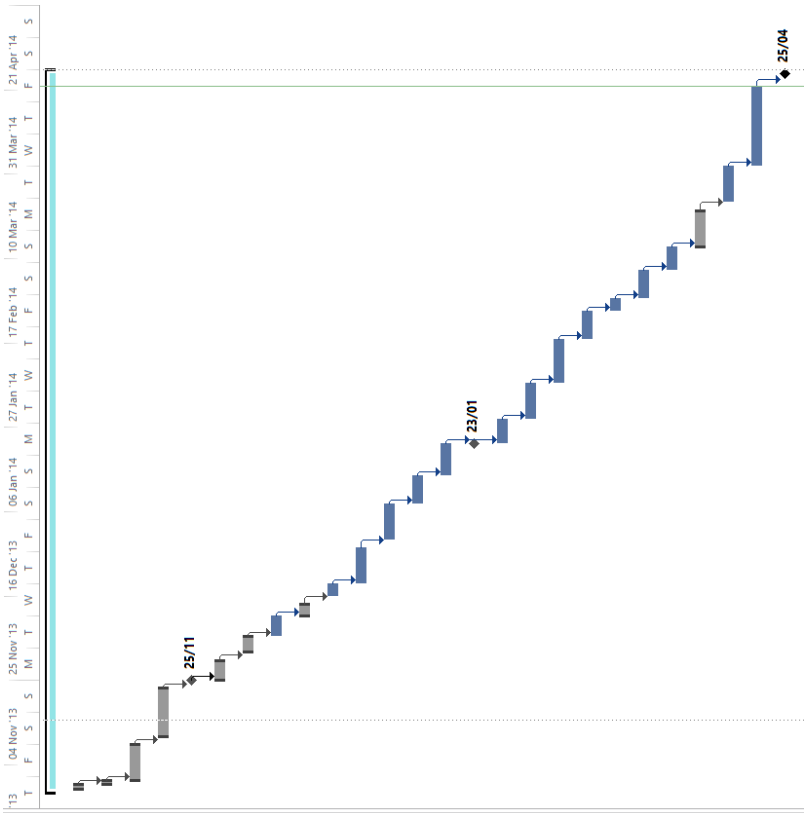
Lessons Learnt

Aside from the scheduling lessons learnt above, and a huge amount of knowledge regarding JavaScript, PHP, and the web as a whole; another key element I learnt was how useful the correct tools can be for developing a project.

- Text editors such as Sublime Text 3, with syntax highlighting for JS, PHP and HTML for faster development, error checking and formatting.
- Linters such as JSLint, for checking code quality and formatting.
- Web development tools such as Firebug. These were hugely useful in finding errors in the code, checking correct functionality and tracking server requests.
- Source version control such as GIT, for tracking changes to the repository, and allowing reversions if necessary.

Final Gantt

Through using the 'git log' tool, it is possible to create a more accurate final Gantt chart.



Task Name	Duration	Start	Finish
Develop Project	**130 days**	**Mon 28/10/13**	**Fri 25/04/14**
Define key and bonus functionality	1 day	Tue 29/10/13	Tue 29/10/13
Create wireframe of design	1 day	Wed 30/10/13	Wed 30/10/13
Research plausability of bonus functionality & define timescales	7 days	Thu 31/10/13	Fri 08/11/13
Write first interim report	10 days	Mon 11/11/13	Fri 22/11/13
Interim One	0 days	Mon 25/11/13	Mon 25/11/13
Design main headers, footers and layout	5 days	Mon 25/11/13	Fri 29/11/13
Build layout CSS	4 days	Mon 02/12/13	Thu 05/12/13
Design database structure	3 days	Fri 06/12/13	Tue 10/12/13
Implement Search	3 days	Wed 11/12/13	Fri 13/12/13
Build PHP to create DB	3 days	Mon 16/12/13	Wed 18/12/13
PHP to retrieve data from DB	7 days	Thu 19/12/13	Fri 27/12/13
PHP to update or delete items from DB	7 days	Mon 30/12/13	Tue 07/01/14
Add Image functionality	5 days	Wed 08/01/14	Tue 14/01/14
Implement Product Adding/Updating	6 days	Wed 15/01/14	Wed 22/01/14
Interim Two	0 days	Thu 23/01/14	Thu 23/01/14
Build dynamic product info page	4 days	Thu 23/01/14	Tue 28/01/14
Create feedback system	7 days	Wed 29/01/14	Thu 06/02/14
Implement product basket	7 days	Fri 07/02/14	Mon 17/02/14
Refactor code	5 days	Tue 18/02/14	Mon 24/02/14
Give feedback on product availability	3 days	Tue 25/02/14	Thu 27/02/14
Pages for updating/removing products	5 days	Fri 28/02/14	Thu 06/03/14
Ability to create, view, and process orders	4 days	Fri 07/03/14	Wed 12/03/14
Build Installation scripts & sample entries	7 days	Thu 13/03/14	Fri 21/03/14
Improve Mobile Compatability	7 days	Mon 24/03/14	Tue 01/04/14
Create Documentation & Test	14 days	Wed 02/04/14	Mon 21/04/14
Final Submission	0 days	Fri 25/04/14	Fri 25/04/14