Assessment Two – RAD Project

Daniel Swain 23/04/2016

Methodology and Business Rules:

The methodology we’ve used for completing the Real Estate Website project is Scrum, an agile software development methodology. Scrum sets clear roles for team members and project leaders to facilitate agile development to complete small mini projects, aka ‘stories’, in short durations, aka ‘sprints’.

We’ve applied Scrum to our project using tools such as Slack, Trello and Github. We’re using Trello as our Scrum board. As Scrum Master I update Trello with our sprint tasks and provide alerts through Slack when tasks are assigned and completed. Github allows for all team members to see, edit and play with the code that others are making in their sprints, either to update or to extend into their own feature branch.

Due to the nature of agile development (small, modular tasks, completed in quick succession) there is a great feeling of immediacy amongst the team. Team members can see working code being added regularly, code which adds new features or functionality, and this increases motivation to tackle their own modules.

For a project where the client interaction has been minimal, this modularity and immediacy is vital as it allows the team to be flexible to any decisions by the client without having to restructure the project. These are the main reasons why an agile development methodology like Scrum is better suited to this project.

Business rules are recorded primarily in Trello, accompanying the story card either as the acceptance criteria or as a separate task for a future sprint. As stories were discussed in more detail after Scrum meetings I, along with the product owner and team leader, would add descriptions to our story cards as to what would satisfy the completion of that story.

Again Scrum is well suited to this task as all team members have direct input in the Scrum meeting, discussing their story progress, and giving opportunity for other members and the management team members to notify of any new business rules set after discussions with the client.

Modularity and Iteration:

We focused on modularity before we had set up the project and application in Rails and on Github. During the first sprint we outlined the overall sitemap for the site, what pages would go where and what would link and interact with each other. This allowed us to identify areas that were shared across pages, as well as what would constitute the smallest development task.

We set up individual controllers in Rails for each main page, allowing team members to work on their own module prior to integrating it into the master branch on Github. Global functionality like the header, nav menu, footer and login/signup actions were also split into separate tasks and were tied in once development had progressed.

In initial discussions we established preferences to split site rendering into a collection of partial html.erb files of smaller scope following a don’t repeat yourself (DRY) ideology. This had the added benefit of enabling future expansion without having to identify the areas needing change in one large block of code.

The use of partials also allows for continuous change based upon client feedback, and streamlines the process with less reword. An example of that is through the use of common variable names and class names in our stylesheet definitions. By setting these variables before major site development had begun it allows us to change colouring and styling of objects site wide from one or two files, rather than each and every web page.

Both the modularity and focus on streamlining change should hold us in good stead for future user integration, allowing us to achieve their desired goals in shorter timeframes.

Quality Assurance and Testing:

All team members are completing their stories and development tasks on their own feature branches on Github. This minimises the potential for team members to affect other team members during the development of their features and provides a well understood and documented method for integrating all work from all areas.

Coding standards were set prior to the commencement of development tasks, with all team members agreeing to the standards across a range of expected programming languages. This facilitates easier code review due to standardised coding and documentation. We have also assigned a single team member to handle merge requests when team members have finished on a feature and with to pull it into the master codebase. By focusing on standards and documentation we make quality assurance testing easier as team members are able to find and document bugs or issues in other team members code quickly and effectively.

This is demonstrated on Github where we have currently identified 15 issues in this 3-week sprint from team member’s feature branches when they were pulled into master. Of these 15 issues, we had 32 comments (an average of 2 per issue) with 12 being marked as closed and the remaining 3 being actively worked on or platform specific issues that are hard to identify the cause for.

By continually developing on the project team members are helping to identify issues and perform quality assurance. Many hands make light work, and in this situation the high amount of activity (>220 individual commits in ~ 3-4 weeks) by our team members increases the likelihood of identifying and fixing issues during development.

Demonstration:

By following the Scrum method for the development of this Real Estate Website we have found development to be progressing at a rapid pace with continual improvement and addition of new features. I’ve recorded footage of the current progress of our site to demonstrate the effectiveness of our methodology in achieving our project goals.

[Dropbox link to video (22MB)](https://www.dropbox.com/s/al7kzhc9lawlvpc/Real%20Estate%20Website%20Sprint%202%20Demonstration.mp4?dl=0)