Keith Ellison

Sprint review and retrospective

CS 250

How did switching to agile help in the development of SNHU Travel? Working in the agile style allows an entire scrum team a level of dexterity that came in handy when we were given new directives well into development to now pivot towards wellness getaways and retreats. Adapting on the fly conceptually appeals to me on it’s own, being mandated to do so by a new design paradigm might be one of the most exciting aspects of the project. Having to work with a smaller team represents itself to me the same way, this time standing out to me like old-school band; they’re “lean and mean” the way an old school band might be or might become if they ever get to be too bloated. I know I’ve mentioned before that I have personally learned the value of milestones; so early and often delivering progress reports to the product owners serves the dual purposes of settings repeated standards for development that can keep a team focused, but also keeps those owners engaged in the process enough that they can provide viable feedback as development continues.

Frequent Scrum meetings, likely every morning as part of the development process encourages interaction and the free exchange of ideas. While working with the user stories I distinctly remember the later one that imposed sweeping changes on all the work that had been done up to that point. Of course this is easily supported within the Agile methodology, as we could easily rollback to an earlier iteration of the code and build back up from there since the framework is already in place, and the increments of development are meant to preserve our progress while enabling up to quickly adapt. That’s the example that stuck with me the most clearly, but there’s the first story, and the basic challenges associated with getting started on a project; over the course of development before the aforementioned late game pivot, we were developing those working iterations, getting familiar with our customers, each other, and the tools we work with, that last one represents a universal advantage in developing software; experience.

I feel that I’ve already explored the pivot pretty thoroughly, even so, if I were going to add a few finishing touches; I could say that I was excited by it as a twist so late into development, to now have circle back to a previous checkpoint like it’s a system restore point on my personal PC, and from there, developing it back up with a ‘new’ build. For me, the daily scrum every morning is the time to catch up and prepare for the next push over the course of the work day, and by push, I mean the big drive towards our next production milestone. To begin, I would ask If anyone felt “unsure”, specifically , “unsure” about the quality of their work so that someone can go over it, assess it’s quality and note potential problems; big ones get addressed right away, smaller ones are noted, and we move on; “Sir, I think I’m struggling with assigning values that will display the right images at the right time, I do a practice run in the app and the whole thing breaks.”, someone might’ve said something like this for example, and the reply; “Ok, send me your IP address, your code, or we can speak tomorrow morning, first thing.”. I feel that one someone, especially a team member, doesn’t feel shamed or less than compared to the rest of the group, they are more inclined to seek help when it’s needed. Considering that, conversation, open dialogue is encouraged, messaging someone with more experience isn’t penalized, and so on.

This is another one that it seems I’ve touched upon too, communication as a tool is the biggest priority to me, besides flexibility, the frankly exciting challenge of adapting on the fly, changing priorities while maintaining the team’s momentum.

While working on SNHU Travel, agile provided the flexibility obviously to adapt to the late game twist, which as mentioned, stood out to me most. I don’t remember any negatives to using the style that I’d been looking forward to. The freedom to adapt to changes in our priorities, in fact, being expectant of problems or a need to adapt, is a great asset. Communication in teamwork is king, and scrum accounts for that, so it all ended up being pros, at least to me. I can’t imagine how a style that expects, if not encourages adaptation and change would be bad for programming.