Sprint Review and Retrospective

Introduction

Over the past few weeks, we have worked as a team to successfully deliver a solid software product using the widely adopted Agile framework. Utilizing this framework allowed us to quickly deliver a working iteration of the project, provide a structure for continued support for the project, and expand our communication and collaboration skills as a team. The following contents in this document review our work for this sprint and analyze how well our approach to Agile fulfilled our expectations.

Scrum Members

Starting with our Product Owner who initially met with our client, the various requirements and rough features of what the client required to be part of their vision began to take shape. Through these client meetings, the Product Owner gathered that the client desired a website built for their travel agency that initially only specified the ability to list the top destinations for travel. The owner then proceeded to hold a focus group of the travel company's clients to then collect information on what features may have been desirable for the website. The culmination of all of this information was condensed into user stories for the product backlog, which contained detailed but partitioned requirements. This feedback allowed the team to come together and begin deciding which user stories would be best to include for the upcoming sprint that they felt best represented the desires of client's customers in the first iteration.

The Scrum Master did a great job assembling the team to develop an Agile charter that would become the basis for how the team operated over the course of the project and how we would apply Agile in our environment. Additionally, the Scrum Master provided aid to the Product Owner and the team by assisting in the development of the initial sprint backlog with the

most appropriate user stories, so that work could begin on planning the first sprint with what user stories, like the top 5 destinations list, that would be included in the first iteration. In addition, the Scrum Master helped facilitate our daily scrum meetings by asking the three questions of what did you do yesterday, what will you do today, and are they any impediments that you are facing? This was a vital point for our communications that allowed the team to share what was troubling them and helped guide the sprint on a day-to-day basis toward the direction of productivity. Throughout development, the Scrum Master helped team members with whatever issues were impeding their productivity, as well as assisting team members with adhering to the Agile charter.

The Tester was responsible for taking the user stories that were chosen for the first iteration from the backlog and writing test cases for the Developer to begin implementing those cases into usable software. Many of the user stories were written into pointed test cases that specifically outlined the requirements for the software to function properly. For example, in regards to a list of the top 5 destinations, the test case explicitly stated what should be clicked on followed by what should display as a result of that, as well as the requirements for what would deem the test case successfully implemented, like only displaying certain destinations. As the requirements changed, the Tester rapidly adapted to the need for the cases to conform to these new requirements, such as when the list of top destinations was required to be a particular format or when these top destinations were to be of a targeted category.

Throughout the development process, the Developer worked closely with the Tester in order to properly implement the written test cases into software that functioned as intended by the planning. Much of the Developer's time was spent working within Eclipse to code the requirements and bring the website to life. Initially, the code was updated to include the top

destinations for travel but there were times when the code needed to be augmented to conform to new requirements.

User Stories to Completion

The user stories are the core of where the team turns bite-sized requirements into tasks that bring the software to life. Beginning with the requirements from the head of the client's organization and the focus group held afterwards that included customers of the client, the user stories began to take shape in the product backlog, holding what the team felt were the most important requirements that aligned with the client's requests. This backlog was then groomed to find what would be appropriate deliverables for the first iteration of development, taken to the sprint planning phase where the team decided how to rank these stories in degrees of priority, and finally divided up amongst the team. The Tester wrote very specific test cases from the user stories that outlined what would constitute a fully implemented feature. The Tester was in constant communication with the Developer, who then worked with these test cases to implement them into the code as a functional feature.

Working Through Changes

One of the primary drivers that lead to the Agile methodology being adopted over the waterfall method was because of how it allowed the team to quickly shift resources and planning when the project took an unexpected turn. There was a moment in development where the requirements shifted due to client feedback from previous iterations, which constituted a change in the development plan. This change was not a difficult one to adapt to because the framework for the team was such that changes could be quickly made at all levels to accommodate this new requirement. The team was brought together and a discussion was had based on the new requirement feedback. The Tester then quickly got to work augmenting the existing test cases

into what the test cases might look like with the new requirements. These changes were given to the developer, who promptly updated the codebase to reflect the new changes. The rest of the processes went on a usual and constant communication was held to make sure these changes could be smoothly implemented.

Communication and Tools

Open and transparent communication is a cornerstone of the Agile methodology that allows projects to maintain a productive heading and accounts for issues that arise over the course of development to be promptly addressed. Among the various methods employed, the daily scrum was perhaps the most effective in allowing the project to maintain a healthy and efficient course. Team members could report what they are working on and bring forth any issues that they faced on a day to day basis. The team can then reorient themselves in those day to day operations based on what comes up to most effectively fulfill the tasks that are slated.

A variety of tools are available that further helped facilitate productivity and communication among the team. Continuing with the daily scrum example, the scrum board provided a neat, clear, and effective way for all members of the team to view the daily status of the project at a glance. The scrum board included everything from the backlogs of user stories, tasks in progress, and features implemented, among others. Having the team together every day to view this board helped facilitate discussion on how the team was doing, where they may be falling behind, and where was a good place to continue from.

Agile vs. Waterfall in Closing

Coming to understand the Agile process and witnessing its performance in the project, it is clear why the methodology is a popular choice for software development. Agile’s strengths lies in its ability to foster an adaptable environment where the project can shift to accommodate

big changes, its quick delivery of working iterations of a product that can give a client something to use or provide immediate feedback on, and its emphasis on constant communication and collaboration between team members that creates a productive and cohesive environment. The biggest weaknesses of Agile may be that the methodology is so focused on delivery of a product, that sometimes proper documentation falls by the wayside, whether due to the quick pace or ever-changing requirements. Additionally, with the constant iterative nature of Agile, a clear and final goal can be hard to nail down if the customer feels it necessary to add new features with every iteration.

Was the Scrum-agile approach appropriate for the project? That is difficult to say given the relatively simple nature of the design that made it difficult to visualize what building a more expansive application from scratch with the listed requirements would entail. Ultimately, it was a productive system that was well organized and executed given the relatively vague requirements given by the client in the beginning. Agile seems to excel where the final product is more unknown than known and the teams have to trailblaze in a way to develop an initial iteration the client is willing to build off of. Additionally, the project included a distinct change in requirements near the end of the development, which proved easy to adapt to with agile. In a waterfall method, the planning for implementing a list and including many kinds of vacations may have been set in stone from the beginning and the testing phases at the end of development may have already been in motion. This would have made it exceedingly difficult to go back and change the functionality of the website since the framework of the project was such that each step relied on the previous step being completed, and therefore was not conducive to flexible, abrupt changes.