CPSC 2350-M01  
 Jay Seung Yeon Lee  
 100 357 736

**QLA 2: Software Process Models and Agile Development**

Q1.  
A1) As a *user,* I can discover new movies that are related to the keyword input so that I have easier access to the movies in my interest.

A2) As a *user,* I can manage my user profile so that I can change the personal information and payment methods on my account.

B) As a *user*, I can change the viewport on my device when I turn it sideways so that I can watch movies on a bigger screen

Q2.  
A) When Waterfall and Kanban are compared from the viewpoint of team productivity assessment, the two comes very close. Waterfall and Kanban both have a great team productivity assessment since everything is all planned out from the beginning for Waterfall and it’s easier to keep track of where the team’s progression is at and how productive the team has been based on the team’s progression compared to the projected time for completion, and Kanban visualizes the tasks members and team has done utilizing Kanban board and cards.

Despite the close competition, I believe Kanban has an edge over Waterfall for the following reasons: For one, Kanban allows team’s productivity to be visualized and easy to locate the team’s whereabout within a step in a project or a task. Furthermore, Kanban specifies which member worked on which tasks, which allows the team productivity assessment to find out which members needs to step up and could potentially guide a way to improve team productivity.

B) When Waterfall and XP are compared in terms of customer collaboration, XP is easily the better software process model of the two. Waterfall hardly has any customer / user involvement during the development process, and XP’s development principle revolves around iterative development through customer involvement and feedback.

C) To compare Scrum and Kanban from the viewpoint of retrospective meetings, I believe it is hard to compare the two. Kanban does not have retrospective meetings because Kanban is not time-boxed and Kanban board itself visualizes the workflow thus retrospective meetings are not needed. However, Scrum’s retrospective meetings allows developers to communicate and forecast team’s project in a more structured manner.

I believe that the comparison between the two depends on the preference of the team, but I highly value Scrum’s retrospective meetings for the following reasons:

1. Retrospective meetings allow team members to look back and discuss about areas to improve and areas team has done well. (Enhances team culture and practices)
2. Grants meeting time as a team to discuss about potential risks and difficulties.

Q3.

*“As a food lover, I want to view the list of the top-10 rated restaurants in my vicinity.”*

I believe this story is violating Negotiable and Testable of the INVEST guideline. Negotiable aspect is violated because the end-goal of the story is clearly stated, but it did not clarify the standard of the top-10 rated restaurant. I strongly think that the story should Include how the restaurants are rated or from which platform they are rated to avoid possible constraints.

Secondly, justifying the area of “vicinity” is necessary for the program to be tested. If the vicinity listed is 500m, 5km, or even 10km needs to be specified to be tested and even to be programmed.

Lastly, story itself seems to only provide precondition of the story. Action taken and expected results seems vague.