When tasked with completing our group project in a short space of time, and with so many other academic and personal responsibilities, it was clear that an organised approach was essential. To ensure we worked as efficiently as possible and with all members undertaking an equal workload, we decided to utilise the agile methodology.

The agile approach to software development allows a project to evolve and change through its development cycle, due to this, each team member must be flexible and willing to adapt to change. Rather than using the waterfall development methodology, which can be compared to “running a long marathon to an imaginary destination” (Bibik, 2018, pg. 16). We opted to use the ‘sprints’ concept. By utilising this concept, we divided our workload into small chunks or tasks to be completed during sprints or short periods of time. This enabled us to “plan and predict the outcome” of our work after each sprint and then “decide where to go next” (Bibik, 2018, pg. 16).

Before each sprint, our group participated in a planning meeting. These planning sessions took place within Blackboard’s Collaborate Ultra groups or Discord, where we set up a group server to ensure we could meet whenever necessary. During these meetings, we would, as a group, decide what tasks we would commit to completing during our next development sprint. These were essential meetings as if they had not taken place “the team would not be able to deliver what it commits to or will work extra hours to complete all the Sprint deliverables” (Bibik, 2018, pg. 21). Due to the intensity of our study and personal commitments, this would not have been possible.

During our first planning session, we identified the key tasks we wanted to be completed by the end of our first sprint, which we decided would be a weeklong in length. We decided to have a ‘User’ super-class; a ‘FullTimeInfluencer’ and a ‘CasualInfluencer’ sub-class; and a ‘Test’ program completed. These programs and classes contained stripped down versions of the features we wished to include in our final program.

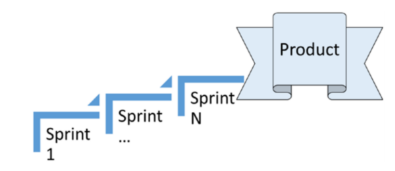
We then decided, as a group, who would complete each task. If someone felt they would be suited to one task over another, they could voice their opinion, and this would be considered by their fellow group members during this meeting. This worked effectively as “the most important part (of working as a scrum team) is that the team will be self-governed” (Bibik, 2018, pg. 16).

During our sprints, we also made use of daily meetings, usually on the group Discord server. During these meetings, we discussed our progress so far and any potential difficulties we were having. These meetings were invaluable to our group as they made it easier to detect any mistakes or inefficient code which could cause issues in the future as our completion date grew closer.

These meetings also enabled each group member to embrace change and become more flexible with our programming. If we had not met regularly to discuss our progress until after each sprint was completed, and the group decided to alter a feature one member worked on, they might feel as if they had wasted their time programming it. However, if this feedback was given while they were programming during the sprint, the task of changing the code would be less laborious.

After each sprint, we then had a sprint review meeting. During these, each member uploaded their work into a shared GitHub folder and talked the other members through their code. The review meetings were longer than our usual daily meetings and acted as another opportunity for each group member to voice their opinion. It also allowed each member to suggest how we could improve the work carried out from our previous sprint, thus allowing us to continuously improve our program. Once we had finished discussing the work completed during the sprint, we could then decide what our next steps would be, what tasks needed to be completed during our next sprint and how this workload would be evenly split.

As this cycle of sprints, daily meetings and debriefs repeated every week, we soon found ourselves closer and closer to our final program being completed. This system of organisation allowed us to be efficient in our programming and hence more ambitious in the features we wished to include, as each iteration improved upon its predecessor.

Figure 1, Sprint Iterations (Bibik, 2018, pg. 15).

References:

Ilya Bibik (2018). *How to Kill the Scrum Monster*. New York: Apress.