**Evaluative Critical Review**

Overall our project management went well; we managed to meet all deadlines to an adequate standard successfully, this was done by a combination of methods and processes such as Agile, GIT and the division of worked based on skills. However, there were problems with our management when it came to things such as leadership and alternatives that would be beneficial to improve the standard of our work.

As a team, we decided early on that the best way to work would be to allocated tasks to individuals so that there was an even workload, and everyone always had work to complete and was contributing. This benefited us by ensuring that we still met every deadline, had any extra work completed and we’re able to support each other if anyone was struggling.

Using GitHub allowed us to have proper change and version control, constant access to all current and previous work as well as logging to see who has contributed what and when. This made sure that everyone had the most up to date versions of anything that they were working on or wanted access to and if errors were made gave us the ability to jump back to an earlier point in development.

Several agile techniques were used during the entire process such as stand-up meetings which we would have every week during our tutorial hours this was to ensure everyone knew what the tasks that needed to be completed were and to inform each other on what we had personally done or found in the case of individual research. We also used MoSCoW prioritisation to rank tasks that needed to be done and determine a logical order for them to be completed.

The main problem we faced was of no clear leadership structure as even though we took an agile based approach, we did not have anyone take the roles of scrum master or product owner and we each would do parts of these. However, if we had determined these rolls beforehand and had a clear leadership structure if this was the case instead of each picking the tasks that they wanted then the scrum master could have assigned tasks and roles to those best suited to them which would likely lead to an increase in quality.

Another problem we had was quality control, which came down to the group just self-reviewing work before submitting it. This method caused problems as you are unlikely to notice errors in your work compared to how someone else might review it. We could implement this next time by asking members from other sub-groups to review work that they haven't had as much influence in. We could have achieved this with a more apparent leadership structure.

Overall our project management proved to be very useful as it allowed us to complete all tasks to a good standard and with plenty of time to spare, gave each member their rolls and had everyone contribute evenly to the workload, have proper monitoring, control and general administration. However, it was not perfect and there were aspects that we could definitely mainly in the top level group management and administration to ensure that we produce the highest possible quality of deliverables.

One of the main problems of the implementation was putting the multiplayer in an AR environment. Me (Sehun) & Daniel were tasked in integrating a form of multiplayer where the players can play against each other competitively on different phones, either through a LAN or on Unity’s provided test servers (up to 10 clients).

We were able to build a multiplayer test project which works through a LAN network. We found implementing this in AR environment was more challenging than we thought. As in Vuforia, it's focused on using its built-in AR camera. Which you can only have one camera, we couldn’t find or think of a solution in time to circumvent this to split the camera or find a solution to integrate an alternative multiplayer solution.

We also chose to implement a quick easy to read the tutorial which is available as a menu option, which we designed in Photoshop and Unity.

We came up with a co-op alternative where the players can take turns (the game will notify the player's turn) this has an advantage of some of the social communication and can form skills and traits of good character such as trust and excellent social skills.

Designing for people with Autism is something we always kept in mind. We always thought and questioned, “How would a person with Autism understand this or play this?” So we did in-depth research on characteristics that people with Autism share, like not being able to understand particular meanings and vocabulary and not being able to read letters as they’re too small, or such struggling with staying focused.

So right from the get-go, we designed with all these in mind. Cameron created a game board, Joe and Josh were in charge of designing and implementing the main menu and board as well as Jack in implementing the models. They came up with a colour scheme with Josh, which they chose to implement a bright, colourful colour scheme. They looked at children apps and the colour themes they used, which would appeal as its very eye catching which would draw attention to the menu but still keeping text readable and while giving our app an identity. From the layout to playing the game, everything was made sure this experience was playable and enjoyable from the start for our audience. We planned well and recorded our process on GitHub sharing ideas and information through group meetings and our WhatsApp team group.

We also focused our efforts on ‘accessibility options’ providing extra functionality such as big text or an alternative colour scheme to ensure the game UI is easy to understand and go through. We wanted to ensure the game was playable and our target audience was catered for. I believe this is a great thing we performed well in; we always had or target audience in mind and made sure our game was an enjoyable experience for our audience. We came up with solutions to problems that came up, and we were very creative in this area.

Were this project to be implemented commercially, several areas of the project management, program design and product itself would likely need to be improved. Firstly, accountability in terms of development would need to be enhanced and visibly better. During the project, little auditing was done on code submissions, and it was possible for people to modify the code, upload it to the project and nobody notice this has happened, which may have led to problems. In a commercial implementation, a clearly defined process of submission, testing and approval would need to be considered to ensure that no unwanted or dangerous changed are committed to the main build. Specifically, somebody other than the developer of the code in question would need to be responsible for approving code for submission into the main build.

Secondly, better documentation of the application’s design and implementation would be needed. Many aspects of the design were never detailed anywhere, such as class interactions, state transitions and general program design. This often-left other team members having to work out and guess for themselves what another person’s work did. In a commercial application, the client would fully expect proper documentation to exist when work is done for them, which was not the case with our product. While we did have a general outline for user-interface design, the class interactions behind it were never discussed and this potentially impacted on the final quality of the application.

Next, the final sprint where we intended to implement multiplayer had to be dropped three-quarters of the way through. This was due to the discovery that the multiplayer aspect that we’d agreed on implementing, in the design we had proposed, didn’t appear to be feasible in practice. The absence of a feasibility study was particularly noticeable in this sprint, with the final build suffering significantly with the absence of what was considered to be the most important feature. While a workaround was developed to ensure we still met the brief, this fell far below what we had set out to achieve with our application and ultimately could have been easily avoided. This part of the Agile paradigm would have been a significant help to reduce the risk of this issue occurring again was a commercial implementation of our game created.

Finally, in a commercial implementation for our product, much more thorough testing would be required. As much of the code was done in individual units and incrementally added, and no clear testing process was defined, we ended up in a situation where only individual part of the code had been tested. However, the whole of these individual parts together was never tested nearly as thoroughly, and this created a major risk of significant bugs going into the final build unnoticed. In a commercial application, a clear testing process would need to be set out and adhered to before each build, with particular attention given to the interaction between individual units and less emphasis on the individual units themselves.