**PROJECT POSTMORTEM SUBMISSION FRIDAY 4TH MAY 2018**

Once you have made your final presentation WE NEED YOU TO SUBMIT THE FOLLOWING COMPONENTS UPLOADED TO A SEPERATELY LABELLED GITHUB FOLDER

1. A SINGLE PAGE OF A4 (MAXIMUM) WHICH LISTS THE OVERVIEW OF THE ASSETS YOU HAVE PRODUCED FOR THE PROJECT, WHETHER THEY HAVE MADE IT INTO THE FINAL GAME OR NOT.
2. A COMPLETED REVIEW OF THE PROJECT **USING THE TEMPLATE PROVIDED BELOW**. PLEASE REMEMBER THAT THE MORE DETAIL YOU ADD TO THIS COMPONENT THE EASIER IT IS FOR US TO JUDGE YOUR WORK. SO AVOID SINGLE LINES OF TEXT. **EXPLAIN WHAT YOU MEAN**.

|  |  |
| --- | --- |
| **STUDENT NAME** | Adam Speers (S191345) |
| **PROJECT NAME** | Group 3 – Table for Two |
| What do you think went well on the project? | The initial team brain storming came up with a strong game idea based on a memory mechanic. I did some research into the workings of short term memory and how much a person was likely to be able to remember (phonological loop) and used this to inform our design.  We quickly prototyped some paper-based ideas and settled on a game based on a food theme, as we thought this would be a good fit with the target demographic.  The team worked well together, and we all had different strengths, which we used to our advantage to make good use of the time available.  The organization of the GitHub folder structure made uploading and subsequently finding work easier. This was much easier when using the desktop client as creating folders via the web interface seems very difficult. |
| What do you think needed improvement on the project? | Code development.  The tools we used meant that simultaneous editing of the unity project files wasn’t possible without causing code contention issues. There was also an issue that the version of unity being used by different members of the team changed mid project also causing issues.  To improve, we should have much more tightly controlled development time windows, with clear owners of the code at the time identified and communicated to all parties.  Preferably we could make use of further tools that allow proper source control so that team members can see that the code base is being edited by someone else, and better tools for allowing merge of changes.  We should state at the start of the project the toolset and versions of all tools to be used. These versions should be maintained throughout the project, or at the least a version upgrade planned into timebox.  Any changes uploaded to Github that have changed the current version of the game should include the team member including a new build of the game so that all team member can launch the latest prototype and see progress / feedback.  Timebox planning.  We should have created a clearer timebox plan for each of the sprints with a user story catalogue holding stories that were prioritized.  We should have reviewed all the stories to identify any dependencies, and then planned this in to avoid blockers.  Any stories that were not developed in a timebox should be identified and go back into the catalog.  The time estimates allocated to stories were not consistent, we should in future hold an end of timebox meeting and review these and future stories to come up with better estimates.  In future we could introduce a formal estimating technique such as poker planning and use these to improve estimating through group estimation sessions. It may also be useful to have a guest experienced programmers / designers attend these sessions to provide guidance.  Playtesting  In future we should organize more structured playtesting sessions, we could have coordinated with some of the other teams to have peer review playtesting each Wednesday PM and have this communicated to all groups. This collaboration would have led to bugs /issues being identified earlier. |
| What do you think of your own contribution to the project? | I think my contribution was important to the success of our project, I researched the memory mechanic so that we had some informed decisions about how complicated the recipes should be and how we could make the game more difficult over time as a result.  Although we hadn’t been taught Unity in semester 1 this was chosen as the main development platform, I took it upon myself to learn unity to aid Elliot in the programming effort.  As a result, I have been able to add value to the project by creating assets and incorporating them into the game directly, as a result adding all the ingredients (138) into the game and associating them with images and sounds was much more efficient. Understanding the toolset also gave me some insight into what was possible and in what timeframes with the tolls we had available. This helped when planning as it allowed me to be mindful of over scoping and including ideas that we wouldn’t have time to develop. |
| **OVERVIEW** |  |
| **Thinking about the project you have worked on this year, what are the important lessons that you will take away from the experience for your next group project?** | The overall experience was very positive, the most important lessons are to iterate ideas quickly at the beginning using rapid prototyping (not necessarily Unreal or Unity, paper is fine). And to get feedback about what will and won’t work to meet the brief.  Planning of timeboxes needs to be more formal and estimates agreed, these should be revisited each planning session as the groups experience improves. We should seek to identify early where there are dependencies and ensure that these are planned in order and prioritized accordingly.  Playtesting should be done frequently and we should seek to use other project teams to help each other with peer testing, preferably timetabled each week so that there is buy in from the other teams.  We should produce a document at the start highlighting the tools and versions that we will use throughout the project to avoid any wasted time in migrating solutions from one version to the next.  We should seek to avoid code contention issues by having the commits and synchronization of the development code more tightly controlled.  The prototype on Github should always be the latest build so that any team member can quickly play / perform testing without worrying that this isn’t the correct version. |