Risk assessment and mitigation

Ben Harris
Joshua Gill
Niamh Hanratty
Amy Raymond
Matthew Czyzewski
Matt Rohatynskyj

In this document our team has outlined the process for risk management on the project and identified potential risks along with ways to mitigate them.

We began by identifying the potential risks we might encounter by thinking of the most common risks that would happen in general group projects, and then coming up with more specific risks for using game design engines and other open source software.

We then categorised the risks, concluding to the categories of Project risks, Product risks and Technology risks. Rather than having two separate categories for People and Requirements, we found a great overlap between risks within these respective categories, deciding to condense these categories into a single category. This makes a quality of life improvement for the team particularly during high-pressure times as information is grouped together logically, along with making the documentation look tidy.

Categories:

Project Risks - Risks found within this category will relate to any risks associated with personnel albeit the customer or team members and the progression of the project.

Product Risks - Risks found within this category relate to any risks associated with the state of the final build.

Technology Risks - Risks found within this category relate to any risks associated with the hardware and software used by the team throughout the development process.

Ownership and Mitigation

Post risk identification we chose to display the risks using tables containing the probability of the risks, ways to mitigate these risks and members responsible for each risk. We chose to utilise a low “bus factor” system for each risk, as this helped us to understand the impact should we be unable to mitigate the risk. Using this system enables us to create appropriate and focused mitigations along with having a number of team members in charge of controlling the risk along with creating a contingency plan.

In our group, we normally have two meetings a week. Once a week, we look through our risk registers as a group, adding any more risks that we have identified since starting more sections of our project. Each owner can then bring up any problems that they have had over the past week in relation to the risks they own and reassess the likelihood and severity ratings. Therefore through doing this we don’t overlook the risks and assume that ‘no news is good news’.

One of our main mitigation methods is making sure there is no single part of the project that relies solely on one person/platform. An example of this, used for P1and P2, is to not have one person doing one task, but instead having at least two members assigned to each task. We have also introduced shadow roles in case the chair, secretary, librarian or report editor are unable to carry out part of their role. Another example, for T1 and T2, is not to rely on one device or infrastructure for our project incase of failure of the system.

Project Risks					
ID	Description	Likelihood	Severity	Mitigation	Owner
P1	A member of the team may be ill or not able to complete their section.	M	M	Redundancy- we will assign at least two members to each task.	Matt Rohatynskyj and Joshua Gill
P2	A member of the team may not have the skill set or knowledge to complete their section.	L	H	We will have shadow roles which means another person can help/take over if that section is not up to the same standard as the rest.	Matt Rohatynskyj and Joshua Gill
P3	One section of the project could be very behind the planned time allocation shown on the gantt chart.	L	H	Keep documentation of the status of each section so all team members know which sections are complete/ incomplete/ behind	Amy Raymond
P4	Team members may not turn up to meetings.	H	M	Make sure that all team members know what they should be doing to make progress that week so that even if they don't go to the meeting, they can still add to the shared documents online. We will also keep notes of our meetings so absent members know what we covered.	All
P5	A member experiences issues with their development software	M	H	Prior to beginning development, ensure everyone can get the software working and knows how to deal with errors	All
P6	A member of the team finds their role difficult and swaps with another team member	L	H	Check in with all team members to make sure they are happy with their assigned tasks- helping out where necessary	Ben Harris and Niamh Hanratty

Product Risks					
ID	Description	Likelihood	Severity	Mitigation	Owner
R1	Library for software is vague and is badly documented so it would be hard to refer to when we need help with the software.	M	M	Choose software that we know there is good documentation for so this can reduce the risk of not knowing how to use the software for certain tasks.	Ben Harris and Matthew Czyzewski
R2	In our group, we could be at risk of accidental copyright infringement. This could be within writing our report or implementing our game code.	M	H	We should always reference work that we take inspiration from and give credit for any assets we use.	All
R3	Final code repository malfunctions due to previous multiple user pushes.	H	H	Protect the code repository, so the repository can only be updated after peer review	All
R4	We are unable to get the software to run consistently across all the required platforms	L	H	Choose a game engine which supports multiple device development.	All
R5	Product behaves differently on the customers device	L	H	Ensure any changes made to the project functions the same for all team members, with no unusual behaviour	All
R6	The game graphics we implement don't fit the ratio of the screen used for the open day.	L	L	While testing, we can try it on different screens and try to create a responsive design.	All

Technology Risks					
ID	Description	Likelihood	Severity	Mitigation	Owner
T1	A device storing files breaks, leaving no access to important files and losing all this progress.	L	H	Store and organise important files on a shared cloud.	Amy Raymond
T2	Web hosting services may go down.	L	M	Have a back-up software that we can use temporarily in case our main one crashes.	Ben Harris
T3	Software development software bugs out or gives flaky results	L	H	Clear the IDE's cache regularly of code implementation-compiling a fresh version every time.	Matt Rohatynskyj and Joshua Gill
T4	Someone from the team commits their code onto the master branch and overwrites other work accidentally.	M	H	Before commiting code, there will be two people reviewing it each time to make sure there aren't any conflicts.	Ben Harris and Matthew Czyzewski
T5	Graphics hardware causes the software to give inconsistent results across more powerful devices.	M	H	Ensure that the software has limiters put in place to prevent more powerful hardware giving the user an unfair advantage.	Ben Harris and Matthew Czyzewski
T6	A team member's computer crashes resulting in a loss of new updates.	L	H	Make sure everyone updates the universal code repository regularly to ensure everyone has the latest version.	All