Risk Assessment and Mitigation

For our risk assessment we established various risks via methods such as; brainstorming sessions, research [1, 2, 3, 4] and filtering out certain risks which were deemed to not be particularly applicable to our small non-critical software [4]. For example individual group members came up with their own risks and then after a review period risks which were deemed to overlap or not be applicable were merged or eliminated respectively. We have chosen two separate scales; one three point scale to represent the likelihood of the associated risk [5, 6]

- Green - Risk is very unlikely to occur.
- Yellow - Risk has a fair chance of occurring.
- Red - Risk is very likely to occur.

and one 5 point scale to represent the impact of the associated risk

- 1/5 - Impact of the risk is trivial
- 2/5 - Impact of the risk is fairly small and can be solved easily
- 3/5 - Impact of the risk is fair and may require some time to be dedicated to solving
- 4/5 - Impact of the risk is quite significant and may require a significant amount of time to solve
- 5/5 - Impact of the risk is great and can significantly affect the time scale of the project and potentially have knock on effects

(Ordered first by likelihood Red to Green; second by Impact from 5/5 to 1/5)

Even if the likelihood of the risk is small and the impact is trivial, the risk should still not be ignored as the risk could potentially happen repeatedly throughout the project to the point where the impact is no longer trivial, such as small bugs in the code.
The reason that likelihood is a three point scale is because all the risks of our small non-critical project fit easily within this scale and will not over complicate things. It was decided however that impact should be a five point scale as impact was more broad and required more accuracy, a five point scale also allows us to better understand the impact of each risk to better prepare us to either avoid or overcome the risk should it occur.

Each risk is also assigned one or more of the following 5 categories, the first 3 denote what the risk effects and the last 2 state whether the risk is predictable or not [6]:

- Project risks - Affect project schedule or resources | **Ref: Proj.**
- Product risks - Affect functional requirements such as features or capabilities and non-functional requirements such as performance or availability | **Ref: Prod.**
- Business risk - Affects the organisation / group developing the software | **Ref: Bus.**
- Predictable risks - Risks that we know are possible and potentially avoided before they occur | **Ref: P.**
- Unpredictable risks - risks that we couldn't of predicted happening | **Ref: UP.**

Each of the risks will fall into one or more of these categories which were chosen specifically to help identify and prioritise particular risks based on what they will affect alongside the likelihood and impact of the risks.

Furthermore each risk is assigned to an individual or group of people who will be responsible for trying to prevent the risk occurring. If the risk does occur then the assigned person/s will be responsible for attempting to mitigate the risk for minimal impact. It is, however, not always possible for the individual or group to mitigate the risk and may require help from other group members.

- Team Leader - The head of the entire project. Responsible for ensuring deadlines and delegating workload
- Task Leader - The head of any smaller groups the team may divide into during the development of the game
- Developer - The developers and maintainers of the program
- Client Interface - The member of the team who coordinates with the customer and clients, if required to mitigate the risk

In the event that a risk occurs which has not been previously assessed within this document, a group meeting should be held in which a discussion of the best method of mitigating the occuring risk. Once the risk has been mitigated an addition to the table below table should state the risk which occured, with a description and the method used to mitigate the risk should also be stated. The table should be updated if and when any new potential risks are found in order for this document to be as accurate and as helpful as possible.

Identified Risk (Risk Ref)	Owner	Category	Causes	Likelihood	Impact	Mitigation
Over Ambitious (R1)	Team Leader	**Proj. Prod. P.**	During the initial design phase we may try to aim too high by aiming to create a game, similar in size and quality of commercial games which is simply not viable for the size of our team, our funding and the time we have	🟥	4/5	Focus on core aspects of the game and add only necessary features which add significant value to the game
Inaccurate planning and scheduling (R2)	Team Leader	**Proj. UP.**	If we try to plan or schedule the development to strictly or have to many dependencies within the schedule, with our inexperience and the unpredictable nature of software and game development there is a chance the schedule hinders our progress more than helps or means we may risk running out of time.	🟥	4/5	Don’t be too strict with planning and scheduling allow for freedom in the schedule and try to spread dependencies as much as possible to allow us to go over schedule
New project uses different environment and/or framework (R17)	Developer	**Proj. Prod. P.**	Members of our team who are not familiar with the new environment and/or framework (e.g Programming language or Software such as Unity) will have to learn the new system before being able to contribute to the team, thus wasting potentially valuable time	🟥	3/5	When choosing the product to take over, be aware and cautious of the environment/ framework used to attempt to minimise the learning curve of the new system.
Tools used cause issues (R18)	Team Leader Developer	**Proj. UP.**	It became apparent during the development of our project that certain tools (namely GIT and JUnit) caused issues with some members of our team, particularly, early on in the development	🟥	3/5	A meeting was held in which any issues, with the tools we were using or otherwise, were voiced and helped to be rectified by the other team members

Quality issues and bugs (R3)	Developer	**Proj. Prod. P.**	It is likely that throughout the development phase of the project, quality issues and bugs will occur which we may notice straight away or may not be realised until much later in the development, either way these issues could cause significant time delays	🟥	3/5	Try to be as thorough as possible when writing code. Test frequently. Attempt to resolve any bugs found as quickly as possible
Team member is absent - Short term (R4)	Team / Task Leader	**Proj. Bus. UP.**	There is a chance that throughout the project a team member may become absent for a short period of time for any number of reasons, such as; illness, personal reasons, other dedications etc	🟥	2/5	Try to give as much notice as possible and delegate the missing members workload as evenly as possible
Procedural risk (R5)	Team Leader	**Proj. UP.**	Day-to-day operational activities might hamper due to improper process implementation, conflicting priorities, or a lack of clarity in responsibilities.	🟥	1/5	Good planning and increase communication via appropriate software
Making changes during development (R6)	Team / Task Leader Developer	**Proj. Prod. P.**	There is a chance we make changes during development, we may realise something doesn't work the way we thought, something is more complex than originally thought and will be too difficult or time consuming to implement or we may want to modify or add a feature all of which could cause knock on effects and be considerably time consuming.	🟨	4/5	Plan extensively, and upkeep regular client and review meetings as a group to ensure changes can be made as smoothly as possible
Sudden requirement growth (R7)	Team Leader Client Interface	**Proj. Prod. UP.**	As the project progresses, a sudden growth of requirement out of our control can create a last-minute unexpected hurdle that can delay the project or cause wider problems within the program	🟨	4/5	Upkeep regular contact with the client to reduce the impact of sudden requirement growth
Unmaintainable code (R8)	Developer	**Proj. Prod. P.**	There is a chance, especially with our inexperience that code may be messy, hard to read, unorganised making code difficult to understand and changes and	🟨	4/5	Regularly review code to modify any unmaintainable code. Comment and follow

			expansions on code almost impossible	🟨	🟧	convention
Demotivation (R9)	Team / Task Leader	**Proj. P.**	Games are hard work to develop, for some, motivation peaks during the initial idea phases with the vision of what the game could be. However once the development starts, the realisation of the workload sets in and people start to run into obstacle after obstacle, causing motivation to decrease. Especially if aims are set too high.	🟨	3/5	Make sure team members enjoy their assigned roles. Regular communication through either meetings or appropriate software will reduce impact here.
Project overtaken contains critical errors (R19)	Developer	**Proj. Prod. UP.**	If the are critical errors within the new project, time must be spent finding and solving these issues before any progress can be made	🟨	3/5	It is critical that in the selection of the new project we ensure and review the project runs as showcased and little to no errors exist
Project overtaken is difficult to understand (R20)	Team leader Developer	**Proj. P.**	If the code of the project we takeover is difficult to understand then time must be spent both learning the program and cleaning the code to make it easier to understand	🟨	3/5	Ensure, during the selection of the new project, we have at least some understanding of the code before choosing the project
Insufficient documentation of new project (R21)	Team Leader	**Proj. P.**	There may not be sufficient documentation on either the project itself or the tools used by the previous team, meaning time must be spent researching this	🟨	3/5	When selecting a new project, ensure that sufficient documentation is provided
Lack of knowledge or understanding (R10)	Team Leader	**Proj. UP.**	Throughout the project it is likely we will find something we don't completely understand and need to research further, leading to mistakes, tasks taking more time than initially thought due to research and tasks being more complicated than originally thought.	🟨	3/5	Research as much as possible and delegate tasks to people who have a better understanding of the specific task. Help from other group members may be beneficial
Compromising during design phase (R11)	Team Leader Client Interface	**Proj. Prod. P.**	In order to get 'stuck into' the next task, there's a possibility that the design phase may be rushed or things may be overseen or	🟨	3/5	Review meetings during and after the design phase can allow us to ensure that we

			missed completely leading to problems during the development phase, wasting valuable programming hours as the design is the most important part	🟨	🟨	have designed extensively as possible and overcome any issues that may arise from not designing fully
Not completed on time (R12)	Team Leader	**Prod. UP.**	There is the risk that if any of these risks come into fruition, the project may not be completed on time, especially as software and games are notoriously difficult to estimate and predict correctly. It is unlikely we will run out of time	🟩	5/5	If near the end of the project the risk of the project not being completed on time is present then, quick panning and re-delegation will be required to complete as much work as quickly as possible
Loss of data (R13)	Team Leader Developer	**Proj. Bus. UP.**	There is a risk that at some point during the project we may lose data, be it, documentation or code.	🟩	5/5	Save everything on multiple cloud services, and have all members save an updated local copy of all data, then all data is recoverable
Inadequate or incomplete testing (R22)	Test Leader	**Prod. P.**	Our inexperience could lead to us producing not enough tests and tests that do not assess the entirety of the code.	🟩	5/5	The Test Lead should produce a full plan for how we will test the project
Team member is absent - Long term (R14)	Team Leader	**Proj. Bus. UP.**	There is a chance that throughout the project we may lose a team member for a large period of time or even for the remaining duration of the project potentially due to serious illness or even dropping from the course.	🟩	4/5	Make sure everyone in the group is happy. Be fair, set everyone deadlines and spread workload as evenly as possible. May be unavoidable
Unavoidable / External risks (R15)	Everyone	**Proj. Prod. Bus. UP.**	These are external changes that are out of our control such as the obsolescence of software.	🟩	3/5	Naturally unavoidable. Use the client meetings for advice in these situations.
Productivity issues (R16)	Team / Task Leader	**Proj. Bus. P.**	It is common place for developers to take things easy at the beginning of the project to try and spread the	🟩	2/5	Spread workload evenly from beginning to end and try to

			workload more evenly or to follow a schedule, however this can lead to team members losing significant time to complete the project	🟩	🟨	motivate team members as much as possible

References

[1] “Applied Risk Management for Small Game Development Teams”, GitBooks, 2015, [Online]. Available: https://ashes999.gitbooks.io/applied-risk-management-for-small-game-develop/content/index.html

[2] M. Schmalz, A. Finn, H. Taylor, “Risk Management in Video Game Development Projects”, IEEE, 2014, [Online]. Available: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6759136

[3] Staff Writer, “Top 10 Software Development Risks”, ITProPortal, (2010, June, 14). [Online]. Available: https://www.itproportal.com/2010/06/14/top-ten-software-development-risks/

[4] T. Arnuphaptrairong, “Top Ten Lists of Software Project Risks: Evidence from the Literature Survey”, IMECS, (2011, March, 16). [Online]. Available: http://www.iaeng.org/publication/IMECS2011/IMECS2011_pp732-737.pdf

[5] D. Sharma, “Risk management in software engineering”, Slideshare, (2014, Aug, 17). [Online]. Available: http://www.slideshare.net/deepkumar814/risk-management-in-software-engineering-38062357

[6] H. Hoodat and H. Rashidi, "Classification and Analysis of Risks in Software Engineering", Waset.org, 2009 http://waset.org/publications/9245/classification-and-analysis-of-risks-in-software-engineering.