Risk Assessment and Mitigation

The identification and management of risks that threaten our project throughout the software development lifecycle is important in order for our project to succeed. We broke the management process down into two phases; risk assessment (identification and analysis) and risk mitigation (planning and monitoring).

In the first phase, we identified all the potential problems that our team and project would face, categorising each risk into similar groups that represented what that group of risks involved. The different risk categories are technology, people, tools, requirements, estimation and process.

The technology risk category is derived from the software and hardware technologies that are being used as part of the system being developed. People risks are to do with the availability, skill level and retention of the people on the development team. Similar to technology, tools risks relate to the use, availability and reliability of support software used by the team such as any Computer-Aided Software Engineering (CASE) tools, Version Control Systems and UML tools. Requirements risks are associated with anticipating the continuous change of requirements as well as any that may be vague and unmeasurable. Since the whole project revolves around the requirements, it was important to come up with measures to take if such situations arose. Estimation risks are derived from inaccuracies in estimating the resources and the time required to build the product properly. Finally, process risks are related to whether the team uses a defined, appropriate software development process and to whether the team members actually follow the process.

After identifying our risks, we analysed each one, indicating the impact each risk would have on our project by giving its severity and the likelihood that a particular risk would occur. As SEPR is a small software project, we decided to keep our risk-severity format simple. We will be using a Low, Medium and High scale as our risk levels. Low risk level is where a risk is identified as having minor effects on the schedule, and the probability of occurrence is sufficiently low enough to give only a minor concern. Medium risk level is where a risk is identified as having possible effects on the end product and/or the schedule. The probability of occurrence is moderate and should raise awareness so we can keep an eye on contributing factors of the risk to try and predict if a risk will occur. High risk level is where a risk is identified as having a high probability of occurrence and the consequence having a significant impact on the schedule and/or the end product. We have included a small table to illustrate our risk format for different levels of likelihood and severity.

	Severity		
Likelihood	***Minor***	***Medium***	***Major***
Low	LOW	LOW	MEDIUM
Moderate	LOW	MEDIUM	HIGH
High	MEDIUM	HIGH	HIGH

In the second phase, we devised avoidance and minimisation strategies for each risk to try reduce the chance of them happening or avoid them from happening altogether. We then decided on risk 'owners' in the team to keep a track of a particular group of risks each, ensuring we are ready for any potential risks that are about to occur and whether we need to take action with the mitigations listed below.

We summarised our work into a risk register below which tells us the likelihood, impact and mitigation measures we will take.

ID	Description of Risk	Severity and Likelihood	Mitigation
Technology Risks			
1	Using a complex programming language to develop the product might be harder for some members of the project team to work with, which may mean unnecessary extra time spent on work.	LOW	In order to prevent this from happening, the group should mutually agree to use a language that all members are comfortable with using. The more experienced members of the team should help the less experienced with understanding the basics of the programming language.
2	Implementation of the specification may be difficult with the programming language chosen. This will mean time will have to be spent finding and adjusting to a different language.	LOW	In order to avoid this happening, the language chosen should be able to implement all of the points in the specification.
3	If the hardware and software are prone to bugs and are “slow”, they may be inappropriate for developing a system. This can affect development, especially during the coding and testing stages.	MEDIUM	In order to reduce the time wasted on finding bugs, the code should be kept as simple and clean as possible with good documentation of code. We should use existing libraries where possible as well as keep the code modularised. The hardware used should at least be fast enough to develop the game in.
4	If the architecture is too complex to be implemented, the architecture design will have to be simplified or modified. This will mean time wasted on re-designing.	MEDIUM	In order to prevent this from happening, the architecture should be designed and all team members should agree on whether it is possible for them to implement all the methods and entities.
People Risks			
5	If the skill level of the team members are not sufficient enough for the development of the project, this could mean either a product that is not of the quality expected by the client, or a product that won’t be met by the deadline due to said members learning the required skills whilst developing the product.	LOW	In order to prevent this from happening, the development team will be required to learn the languages and techniques necessary to develop the game. If any members of the team are particularly experienced with a facet of the development tools, they should instruct those who are not as well versed.

6	Unless planned, a missing team member from the project will have an effect on the schedule of the project because that person is not available to do the work assigned to them.	MEDIUM	In order to maximise efficiency, the missing member should contact the rest of the team immediately to confirm if their absence will affect their work. If so, the missing member’s work should be delegated among the team.
7	Disagreements between members may delay project deadlines as time may be needed to agree on something. In the worst case, it may cause members to leave.	MEDIUM	In order to avoid any delays, if there is a disagreement, it should be brought to the attention of all team members and a decision should be made democratically.
8	If a team member leaves permanently, that will mean their workload has to be shared amongst the remaining team members.	HIGH	In order to finish the work within the deadline, the work should be delegated. An individual member should not be heavily relied upon.
Tools Risks			
9	Lack of understanding of the support software tools used in the process by team members will slow down development of the product, which will affect deadlines.	MEDIUM	In order to minimise the amount of wasted time, experienced team members should help the less experienced members understand the tools. Less experienced team members should also seek online tutorials or documentation for tools if needed.
10	Any important updates for the support tools used whilst in development may mean members have to adjust to new interface arrangements or other significant changes in the software.	MEDIUM	Due to SEPR being such a small project, unless updates are absolutely necessary, all tools should be kept on the same version until the project is finished. If an update is mandatory, team members will have to help each other adjust to any major changes in the software, or seek online help.
11	If any of the support tools fail to function in any way, or do not work as well as anticipated, this will slow down development as a fix will have to be found or a new tool will have to be used.	HIGH	If the performance of a particular tool has come to the attention of most of the team, a replacement tool will have to be found as soon as possible.
Requirements			
12	Changing of requirements may mean the team has to be ready to modify the project to suit the new changes otherwise the	MEDIUM	All team members should be prepared and be made aware that changes in requirements are expected. If a requirement

	wrong product will be delivered.		changes, the team should discuss what changes to the system must be made accordingly.
13	Requirements that are incomplete, ambiguous or untestable will lead to the wrong product or something the client was not expecting.	HIGH	If a requirement has come to the attention of most of the team due to being ambiguous and unclear, the client should be consulted to either remove it or change it to be more precise.
Estimation Risks			
14	Inaccurate or insufficient planning of the number of tools required to complete the project can lead to an incomplete product or poor quality outcome because of wrong estimations in resources.	MEDIUM	All the necessary tools and required software should be identified early on in the development process. If any additional tools are needed, the team should be notified and an agreement should be made about whether it is needed.
15	Inaccurate or insufficient planning of the project schedule will lead to missing deadlines because of wrong estimations in time.	HIGH	Sufficient time should be given for each piece of work when planning and milestones. Deadlines should not be underestimated and team members should agree with the time allocated for each piece of work.
Process Risks			
16	Development of the project without an appropriate, defined process is like working without a good plan and management, which could lead to a weak final product.	HIGH	A suitable, appropriate development process should be mutually agreed upon by all team members.
17	Team members may not follow the desired process, either because they prefer not to or they don’t know how to. This can mean it will be harder to coordinate which members are doing what in the team.	HIGH	Team members should follow the agreed process and try to do the work assigned to them. If team members do not know the principles of the process, they should ask more experienced team members or seek information online.