Risk Assessment

Cohort 3 - Group 4
AJAJARA

Adam Fuller
James Goude
Abbie Spears
Rosie Sherwood
Artem Frolov
Alexander Gibson

Risk Management Process

Our risk management process involves the categorisation of risks into three classifications based on their potential impact on the Software Development lifecycle associated with our project. These are described below:

- **Project:** Risks relating to software implementations and utilities within the project.
- **People:** Risks relating to individual team members and the impact of illness / lack of participation on the design and documentation processes.
- **Product:** Risks relating to the fulfilment of requirements and the extent to which our product meets our client's specifications and needs.

The 'Description' column provides details about each risk, while the 'Mitigation / Prevention Strategy' column outlines the plans to address the risk if it occurs or the strategies already implemented to reduce it's likelihood.

Upon the team's reflection and careful consideration of each risk, both their likelihood and severity were assessed and ranked. This ranking was conducted both before and after implementing appropriate prevention and mitigation measures. Severity was ranked on a 3-point scale from high to low:

- **High Severity**: risks that would have the most severe impact on the project, such as events that threaten the integrity of documentation or software,
- **Medium Severity:** Risks that provide a substantial challenge will be allocated this category. This will typically reflect risks that increase the workload of another member(s) or major, but resolvable, issues with software or documentation,
- **Low Severity:** risks that would have little to no significant impact on the project, such as short term illnesses or meeting internal deadlines as these are relatively simple to manage,

Likelihood was also ranked on a 3-point scale, each risk was rated from Low to High Likelihood, based on the probability of the risk materializing:

- **High Likelihood**: Risks that are highly probable and require immediate attention and mitigation.
- **Medium Likelihood**: Risks with a moderate chance of occurrence, requiring monitoring and contingency planning.
- **Low Likelihood**: Risks that are unlikely to occur but should still be acknowledged and prepared for.

Each risk was assigned an 'owner' to be responsible for monitoring and addressing the risk, based on individual roles within the project. Our risks will be displayed in a grid format to provide clarity and ease of access / review of the problems we identify.

Software Development Methodology

An integral component of our mitigation strategy is the fact we will be employing the agile software development methodology in our project. This will consist of weekly sprint meetings

where each team member will address elements of the project they will develop / complete by the next weekly sprint, allowing individuals to take responsibility for specific tasks and share the workload evenly. Whilst often common practice, we have chosen not to assign a scrum master to act as the figurehead of this methodology as we feel this will reduce the likelihood of serious disagreement, which could threaten / delay the project, and instead promote healthy collaboration between peers.

This methodology will also aid in the identification of risks in the project and provide each member the opportunity to voice concerns each week with which other individuals can assist and subsequently mitigate. Furthermore, by setting individual deadlines each week, we reduce the risk of failing to meet deadlines as each member should be aware and confident in the output required by the next meeting. With these factors in place, we are confident that any problems faced in the project will be dealt with effectively.

Any changes made to this document after Assessment 1 have been highlighted using a variety of colors that indicate the reason for the change: Yellow for rewording or correcting grammatical errors, Red for removed text, and Green for newly added text.

Risk assessment table

ID	Type	Description	Risk Rating Before Implementing Mitigation Strategy	Mitigation/ Prevention Strategy	Risk Rating After Implementing Mitigation Strategy	Owner
R1	People	Team member in charge of implementation is unable to attend a meeting.	Likelihood: Low Medium Severity: Medium	Two people have been assigned to focus on the implementation which avoids dependency on a single individual, mitigating the impact of illness or other unforeseen circumstances. Assign multiple people to work on the Game Logic, and have other team members shadow the main contributors, so they can step in if necessary. Could introduce delays across several parts of the project as the other components are built off of being aware of state changes introduced by the	Likelihood: Low Severity: Low	Artem James

				game logic.		
R2	People	Secretary is unwell or does not show up. Team member in charge of planning is unable to attend a meeting.	Likelihood: Medium Severity: Medium	Upon discussions of role secondary Secretary was appointed to reduce the bus factor, were the primary Secretary to fall unwell or be unable to attend a meeting. A second team member was designated to shadow the primary planner, ensuring they can step in if the primary planner faces a short-term illness. Use trello to allow all members to keep track of project's progress whilst project planning lead is absent.	Likelihood: Low Severity: Low	Adam Rosie
R3	People	Build/version control maintainer stops showing up/ participating.	Likelihood: Low Severity: High	Assign multiple team members to maintain the GitHub repository and ensure everyone on the project has a basic understanding of version control.	Likelihood: Low Severity: Medium	Artem James
R4	People	A team member is unable to work on the project at all.	Likelihood: Low Severity: High	For most sections of the project, multiple team members have been assigned to work collaboratively. Any individuals with sole responsibility for an aspect of the project provide regular updates of their progress to allow another member to take over in any unforeseen circumstances.	Likelihood: Low Severity: Medium	Everyone
R5	Product	Not fully meeting the product requirements in time.	Likelihood: Medium Severity: High	Schedule regular meetings and be realistic about the time it takes to implement a feature (Software developers tend to underestimate software	Likelihood: Low Severity: Medium	Adam

				estimates [1]). Create weekly plans to assign deadlines for different sections of the project. Allow time before the deadline to implement any features that were underestimated.		
R6	Project	Losing locally stored project data.	Likelihood: Low Severity: High	Use GitHub to manage version control and project backup. Project Repository Website Repository Create separate branches for each feature and merge them individually into the main branch. Address any merge conflicts during the process, and if necessary, revert to a previous commit to ensure stability.	Likelihood: Low Severity: Low	James
R7	Product	Incorrect behaviour of game at runtime.	Likelihood: Medium Severity: High	Have a Gradle task that runs unit tests at build time and have robust and consistent error handling throughout the codebase.	Likelihood: Medium Severity: Low	Artem
R8	Project	Merging bad code to source	Likelihood: Low Severity: Medium	Add branch protection rules to the GitHub repository in order to be able to conduct code reviews on pushed code and to be able to approve pull requests before merging. Although unintended merges can be avoided with branches alone, branch protection will add an extra level of security as a pull request will have to go through another layer of checks.	Likelihood: Low Severity: Medium	James
R9	Product	Client unhappy with product direction	Likelihood: Medium Severity: High	Regularly review the product Brief to ensure the product meets the set requirements. Maintain regular communication with the	Likelihood: Low Severity: Medium	Abbie

				client through emails and meetings to ensure the project aligns with their expectations.		
R10	Product	Inconsistent naming of attributes and methods in source code	Likelihood: Medium Severity: Medium	Create a class designated to storing and explaining any constants used in the game to aid understanding and traceability of code. Utilise GitHub branch protection and pull request reviews to identify and correct any poorly named variables.	Likelihood: Low Severity: Low	Artem James
R11	Project	Overwhelming a single class with game logic components	Likelihood: Low Severity: High	Separate game logic components across suitable classes to aid debugging and readability of code.	Likelihood: Low Severity: Medium	Artem James
R12	Project	Game runs poorly on hardware (Stuttering/ Slow)	Likelihood: Low Severity: High	Aim to keep implementation of the game as light as possible whilst containing all wanted components. Use a graphical style suited to games intended for lightweight hardware (8-bit, Top Down).	Likelihood: Low Severity: Medium	Artem James
R13	Project	Code does not conform to Google style guide	Likelihood: Medium Severity: Low	Set up GitHub actions and development scripts to automatically check code style when a pull request is made.	Likelihood: Low Severity: Medium	Artem James
R14	People	Delays and errors arise after project handover due to unfamiliarity with tools, methods, and document structures used by the initial team.	Likelihood: Medium Severity: Medium	Thoroughly review all documents before selecting to overtake projects to identify unfamiliar tools and methods used by the initial team. When making modifications to the original documents, ensure full understanding of the structure so these changes can be made appropriately.	Likelihood: Low Severity: Low	Abbie

				Account for possible delays when the overtaking team familiarises themselves with the new tools/ approaches.		
R15	People	Delays due to misunderstanding project components transferred from the previous team - such as code or .	Likelihood: Low Severity: Medium	Communicate with the previous team in the case of any misunderstandings before errors are made.	Likelihood: Low Severity: Low	Adam

References

[1] Bent Flyvbjerg, Alexander Budzier, *"Why your IT project may be riskier than you think "* , hbr.org https://hbr.org/2011/09/why-your-it-project-may-be-riskier-than-you-think (accessed 9th November 2024)