Risk Assessment

Cohort 3 - Group 10
Backlogged

Ben Walker
Chris Williams
Gavindu Tissera
Josh White
Nana Twum Oviarobo
Tamerlan Urazbayev
Tymur Topala

Risk Management Process

Our risk management process involves the categorization of issues 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 clients specifications and needs.

Upon reflection of each of the risks as a group and discussion about their impact, they are assigned a category exhibiting their implications before and after appropriate prevention / mitigation has been taken. These can be seen below:

- **High:** risks that are of greatest impact to the project will be assigned this category. This will be reserved for events that threaten the integrity of our software or documentation.
- **Medium:** 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:** risks that have easy mitigations and provide little to no significant impact will be appointed to this category. Issues such as meeting deadlines or short-term illness are likely to reside here, as these are relatively simple to manage.

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.

Risk assessment table

ID	Type	Description	Risk Rating before	Mitigation	Risk rating after	Owner
R1	People	Game Logic Designer stops showing up/participating	Likelihood: Low Severity: High	Allocate more than one person to work on the Game Logic and another team member to shadow the main contributors, so if necessary they can pick up where the previous main contributor left off. 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 game logic.	Likelihood: Low Severity: Medium	Ben. Tamerlan , Chris
R2	People	Secretary is unwell or does not show up	Likelihood: Low Severity: High	Upon discussion of roles, a secondary Secretary was appointed to reduce the bus factor, were the primary Secretary to fall unwell or be unable to attend a meeting.	Likelihood: Low Severity: Low	Chris, Josh
R3	People	Build/version control maintainer stops showing up/participation	Likelihood: Low Severity: High	Have several people on maintaining the GitHub repo and have everyone on the project at least have an idea of how the version control works. We can then allocate new team members to manage the version control and build of the project, however this would pose at least some sort delay on the project timewise.	Likelihood: Low Severity: Medium	Ben, Tamerlan
R4	People	Team members stop showing up	Likelihood: Low Severity: High	Have several people in various roles throughout the project and have team members to shadow the main contributors - already done across multiple aspects. As this has already been accounted for in the various roles, the mitigation has been scaled.	Likelihood: Low Severity: Medium	Everyone

R5	Product	Not fully meeting the product requirements in time	Likelihood: Medium Severity: High	Schedule more meetings and be realistic about the time it takes to get a feature completed (Software developers tend to underestimate software estimates [1] We will also likely schedule frequent sprints as well and have our own deadline to plan around.	Likelihood: Low Severity: Medium	Everyone
R6	Project	Losing locally stored project data	Likelihood: Low Severity: High	We are using GitHub to manage version control and project backup. Project repository Website repository We can also introduce several different branches for features and Merge them individually into the master branch and resolve any possible merge conflicts or roll back to a previous version of the project if need be.	Likelihood: Low Severity: Low	Tamerlan
R7	Project	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	Everyone
R8	Project	Merging bad code to source	Likelihood: Medium 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	Tamerlan
R9	Product	Client unhappy with Product direction	Likelihood: Medium Severity: High	Keep in contact with clients with meetings and through emails to make sure the project is staying inline with their goals.	Likelihood: Low Severity: Medium.	Everyone

R10	Project	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	Everyone
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	Everyone
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 to be ran on lightweight hardware (8-bit, Top Down)	Likelihood: Low Severity: Medium	Everyone
R13	Project	Code does not conform to Google style guide	Likelihood: Medium Risk: Low	Set up a GitHub actions and development scripts to automatically check code style when a pull request is made	Likelihood: Low Severity: Low	Ben

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)