

Risk Assessment and Mitigation

Cohort 1 Group 2

Yusuf Almahmeed
Jane Boag
Charmaine Chamapiwa
Charlie Coyne
Anupam Cunden
Bertie Kerry
Leo Xu-Latti

This page does not count toward the total page number

Risk Management Process

Our team have used quite a simple risk management process, suitable for the scope and given nature of the project. Responsibility for risk management is shared amongst the group, ensuring we have no single point of failure. This involves identifying and managing risks making sure none impact the project drastically. As we progress through our task list, we regularly consult the risk management table. We check to see if the current point of the project would be the appropriate time to implement one of the mitigation strategies. This allows risks to be addressed quickly and for the project to adapt and evolve.

Our risk register is split into eight columns:

1. **ID:** A unique identifier for each risk.
2. **Risk:** Describes the particular risk or scenario of concern.
3. **Type:** Risk category (project, people, product, technology, business).
4. **Impact:** The possible outcome should that risk occur.
5. **Severity:** A score from 1 (low) to 5 (high) showing how much that impact would affect the project
6. **Probability:** A score from 1 (low) to 5 (high) indicating the likelihood of the risk occurring.
7. **Mitigation:** A short explanation of how the risk can be averted, or otherwise lessened.
8. **Ownership:** Indicates the group member responsible for managing and mitigating this risk. This is assigned to either the programming half of the team, or the documentation half of the team.

Risk Register

| ID | Risk | Type | Impact | Severity (1-5) | Probability (1-5) | Mitigation | Owner-ship |
|----|--|------------|--|----------------|-------------------|---|---------------|
| R1 | Poorly written code that over-exerts users hardware. | Product | Worsened user experience. | 2 | 2 | Thoroughly test code to ensure it works on all systems well. | Programming |
| R2 | A poorly designed game. | Product | Users become bored. | 2 | 4 | Playtest the game regularly and gather user feedback. | Documentation |
| R3 | Copyrighted content is included into the game. | Product | The (fictional) company is sued. | 4 | 3 | Check assets licensing before using them to make sure only the correct ones are used. | Documentation |
| R4 | The developer falls unwell. | People | Development falls behind, project ships unfinished. | 3 | 3 | Implement a structured plan with redundancy in place, and make sure other team members are able to cover other tasks. | Both |
| R5 | Two developers disagree on a course of action. | People | Development falls behinds, project ships unfinished. | 3 | 4 | Ensure there is some mechanism to resolve conflict and pick a choice. | Both |
| R6 | File loss | Project | Loss of work, rework needed. | 4 | 1 | Regularly save, have backups. | Documentation |
| R7 | Software bugs. | Technology | Poor user experience, frustrated users, negative | 3 | 2 | Regularly test to find and remove bugs. | Programming |

| | | | | | | | |
|------|--|-------------|---|---|---|---|---------------|
| | | | reviews. | | | | |
| R8 | miscommunication | People | Missing work/code. | 3 | 2 | Weekly meetings, maintaining group chats whenever decisions are made and task allocated, | Both |
| R9 | Client unavailable. | Project | Misunderstanding of requirements and not given clarification. | 3 | 1 | Regular client meetings with a pre-defined list of questions needing clarification. | Documentation |
| R 10 | Software/Hardware incompatibility. | Tech-nology | Poor/no performance on user devices. | 4 | 1 | Early access to customer hardware, multiple tests on multiple platforms. | Programming |
| R 11 | Unfamiliarity with tools. | People | Slow progress, delayed game development | 2 | 2 | Use commonly known tools that the team are familiar with or provide tool training if this is not possible, | Programming |
| R 12 | Flaky Libraries, libraries used become unsupported | Tech-nology | Need to rewrite code or find alternative libraries. | 3 | 2 | Monitor library updates, choose libraries that are maintained, use functional programming for easy replacement. | Programming |
| R 13 | No longer demand for | Business | Less sales due to less | 2 | 2 | Continually gather user | Documentation |

| | this genre of game. | | interest. | | | feedback from user research. | |
|------|---------------------------------|------------------|---|---|---|--|------|
| R 14 | Client's requirement changes. | Project/ Product | The game may need reworking or redesigning leading to delays and more work. | 3 | 3 | Regular and clear contact with the client to always know what they want. | Both |
| R 15 | Team member leaves the project. | People/ Project | One less developer means that other members may have to increase their workload or learn new skills to replace the member who left, resulting in slower progress. | 3 | 1 | Share knowledge within the group so everyone knows what everyone is doing. Delegate tasks equally, not just one person left to do the work of the development. Use a shared drive to store files so everyone has access to them. | Both |