Risk Assessment and Mitigation Report

When developing this project, there are a few different types of risk, such as technical, business and operational risk. During development, there are a few ways which we can use to mitigate challenges.

Risk Identification

We began by identifying potential risks across different categories: technology, people, requirements and estimation. This involved stakeholder analysis, technology assessment, and analysis of external dependencies. We also eliminated risks with either a very low likelihood or very low damage to the project.

Risk Analysis and Prioritisation

Each risk must be assessed based on likelihood of occurrence and impact on project. Risks are then ranked so we can prioritise those with the highest potential damage.

Risk Register

Maintained throughout the project. Serves as a dynamic document for recording the likelihood of the risk, impact and strategies for mitigation. Frequently updated to reflect new risks or changes to the project.

Mitigation and Monitoring

For every risk, we will develop appropriate mitigation strategies. We will also ensure regular monitoring of risks and effective communication with the team. Redundancy is also important, we will make sure there is no single point of failure within the team for each critical task, and no important file exists on one machine only. Owners of risks will need to update their likelihood and impact regularly.

Agile Approach

The project is broken down into smaller iterations, which allow us to check for risks at each phase. By doing this, we can address challenges in smaller, more manageable pieces.

RISK REGISTER

The risk register's format consists of columns for:

- Risk Description
- Likelihood
- Impact
- Mitigation Strategy
- Owner
- Current Status

Likelihood and Impact Measured using simple terms:

- High
- Medium
- Low

Risk ID	Risk Description	Likelihood	Impact	Mitigation	Ownership	Status
1	Delayed software development due to team skill gaps in game engine usage	Medium	High		Project Manager	Open
2	Bad integration between buildings affecting gameplay and satisfaction	High	High	Test integration early on to avoid bigger problems later		
3	Communication breakdown between customer and developers	Medium	High			
4	External dependencies of third-party tools	Low	High	Identify backup tools and libraries in case the main tool fails		
5	User dissatisfaction due to unbalanced event responses	High	Medium			

6	Insufficient time for proper user testing before launch	Medium	Medium	Allocate additional time and the end of each iteration for user testing		