Risk assessment and mitigation

Assessment 2

Team 10: Hard G For GIFs

Dragos Stoican

Rhys Milling

Samuel Plane

Quentin Rothman

Bowen Lyu

Change Report

Introduction to Risk Format and Level of Detail

We have identified risks that we believe are relevant to this project by going through a number of sources.

1. Researching general software development risks.
2. Discussing with other people who have had previous experience.
3. Discussing hypotheticals amongst ourselves and evaluating them.

We chose to display the risks in a tabular format. Each risk has its own unique ID to identify it from one another.

The risks are separated in three categories: product (regarding quality/completeness of the game), project (regarding the project resources and schedule) and business (regarding the organisation procuring/developing the software). These types were inspired by research into risk types in software engineering.

We use the Low-Moderate-High scale for measuring the severity and likelihood of each risk. It's a simple and clear format for presenting risks applying to a small-scale project such as ours.

- Low - Very unlikely to happen | Not much damage
- Medium - It could happen or not, depends | A bit of damage, which may affect deadlines. Should move swiftly to fix
- High - High chance of this happening, pretty common | A large sum of work will be needed and will definitely affect the deadline. Every group member should focus on it to recover, perhaps involving staff.

In terms of the level of detail, we have provided a coherent description of each risk which sets out what it entails and how it would affect the project and team, as well as how to mitigate it in the event of its occurrence.

We have given each risk an owner as recommended and a backup owner as this was a simple way of reducing the chance of a risk resulting in a problem. On some risks, the entire team looks out for it in their own work and reports anything if there is an issue.

Risk Assessment Table

ID	Type	Description	Likelihood	Severity	Mitigation	Owner	Backup Owner
R_MIA	Project	Team member unavailable for remainder of project	L	H	Ensure knowledge and skills are shared between team members, and form subteams to work on tasks	All	N/A
R_GITHUB	Project	Github server goes down so we cannot access the repository.	L	H	Back up files in local repositories	All	N/A
R_REQUIREMENTS	Project / Product	Errors or changes in user requirements	L	H	Use agile methods and frequently review requirements and how we are meeting them, meet with client often	All	N/A
R_PRODUCTIVITY	Project / Product	Overall productivity affected by external factors e.g. COVID-19 situation	M	M	Make sure work is spread evenly and work is done in teams or subteams with support on tasks offered from all team members and set achievable, sustainable timeframes	Quentin	Sam
R_OBSOLETE	Business	Software benign used becomes obsolete or unusable	L	L	Have backup pieces of software which can be used on all work and ensure that they are ready to be used in this event	Dragos	Sam
R_ESTIMATION	Project	Incorrect estimate of project completion date resulting in overrun	L	H	Review schedule and plan as a team, keep constant track of what team members are working on, and how much work is left to do. Make specific duration estimates for each task	All	N/A
R_CODE	Product	Poor quality code or clashes in code produced by different team members	L	M	Make sure code meets software and user requirements, and review often. Test code frequently and appoint reviewers for different sections. Use continuous integration of code and lots of documentation to ensure this doesn't occur	All	N/A
R_SCALABILITY	Product	Code doesn't cater to changing requirements and isn't reusable by other team members	M	M	Write simple and efficient code with clear documentation, frequently test and review code	Dragos and Sam	Rest of team

R_WEBSITE	Product	The Website isn't up to date/is obsolete	L	M	Make sure that multiple people have access and understand how to code the website in order to properly maintain it to reflect our progress	Rhys	Quentin

Bibliography

[1] I. Sommerville, Software Engineering, Pearson Education, 2008, pp. 74-98. [2] K. Schwaber and J. Sutherland, The Scrum Guide™, 2017 pp. 1-19