Risk Assessment & Mitigation

Assessment 2

Group 3:

Ben Howard <bh1219@york.ac.uk>
Cai Hughes <cabh500@york.ac.uk>
Harry Richardson <hr1040@york.ac.uk>
Ivan Ndahiro <in597@york.ac.uk>
James Sutton <jis509@york.ac.uk>
Yuzhao Liu <yl5164@york.ac.uk>

Group 6:

Igor Smollinski <is942@york.ac.uk>
Pranshu Dhungana <pd861@york.ac.uk>
Zack Tyler-Kyle <ztk500@york.ac.uk>
Phoebe Russell <pbr508@york.ac.uk>
Sanjna Srinivasan <ss3264@york.ac.uk>
Sam Savery <sgs527@york.ac.uk>
Ewan Hutcheson <@eh1776york.ac.uk>

Risk Management Process

This document breaks down the issues that we anticipate encountering throughout our project. All entries have been agreed upon by all team members before they were added. Although many potential problems could be identified at the beginning of the project; any foreseeable risks encountered during the run of the project will be added as soon as they are feasible.

We have agreed that to best manage this document and what it represents we should revisit it regularly during meetings allowing us to update, append and discuss it. Each group member will also be expected to track the risks they own and report back on them to the group during these meetings.

For higher likelihood risks the owner is also responsible for taking preemptive measures to mitigate/prevent an occurrence. These steps may take more work such as completing backups of shared documents. In these cases, we hope that going over currently listed risks again will ensure these tasks are always completed.

Risk Monitoring

In addition to identifying risks, it is essential to constantly monitor the risks to ensure if they are being mitigated, and identify any potential new risks along the way. Owners of each risk need to constantly reassess their severity and likelihood and report on them to the rest of the team members. Many times, during the development of a relatively big project like this one, we can come across a number of risks during the implementation of the project. One of the risks we came across after inheriting team 3's code was the memory leak it was causing. Our team identified it during the initial testing stages of team 3's code, and mitigated it. During the final stages of implementation, we identified another location in the code which was causing a memory leak, and fixed it as soon as possible. During our regular discord meetings, we discussed all these new risks identified, and if we have resolved them fully. Adopting SCRUM ensured that we had a proper system put in place to conduct risk monitoring and report on the status of each of the risks.

Risk Register

Fields of note

Type:

We have split our risks into four different types based on their causes and effects. This is to allow us to better track what risks could happen at different stages. We can also use these categories as a shorthand for what can be prevented/mitigated. They are as follows:

- ➔ *Implementation* - Risks that could affect our ability to create the product.
- ➔ *Product* - Risks that relate to a change in what we are working towards.
- ➔ *Technology* - Risks that could be triggered by the technology we are using.
- ➔ *Team* - Risks caused by people within our team + the customer.

Likelihood:

How likely an occurrence of this risk is. Also affects how actively we need to monitor/work to prevent it. Categorised as follows:

- ➔ *High* - The risk will most probably affect us during the project and must be prevented/accounted for
- ➔ *Medium* - The risk occurring is not entirely unlikely but we should be prepared for it

- ➔ *Low* - The risk likely won't come to fruition during our project but is still worth monitoring just in case

Severity:

Describes how much an instance of a risk will affect our project. Will also affect how much planning we should do for the event it does occur.

- ➔ *High* - The risk will have a detrimental impact on the development of the project causing long-lasting or substantial loss on the final product.
- ➔ *Medium* - The risk will have a noticeable impact on the development of the project causing short-term or minor long-term consequences.
- ➔ *Low* - The risk will have a negligible impact on the development of the project creating minor short-term consequences.

Owner:

Shows who is taking responsibility for managing a risk. In the event that the management must be done individually or collectively, the owner is listed as *everyone*.

ID	Type	Description	Likelihood	Severity	Mitigation	Owner
R1	Team	The loss of or unavailability of a team member	Medium	High	Allocate multiple people to the same work or make sure everyone's work is up-to-date and can be viewed on google drive	Zack
R2	Technology	A team member's laptop isn't working	Medium	Medium	Use of university machines and if necessary relocation of work until fixed	Everyone
R3	Technology	A local repository becomes corrupted	Low	Low	Clone repository again	Everyone
R4	Technology	GitHub goes down	Low	Medium	Use repositories stored locally and consider different version control system	Igor
R5	Implementation	Bad push updates which might cause errors for the best of the project	Low	Medium	Make sure each potential merge and commit is peer-reviewed by the team. Ensure there is a way to revert changes by using branching and also backups.	Igor
R6	Technology	Google Drive goes down	Low	Medium	Back-up on the local drive.	Pranshu
R7	Technology	Losing data	Low	Medium	Back-up online and have multiple copies on other systems.	Everyone

R8	Implementation	Someone steals our work	Low	Low	Contact CS exam team	Igor
R9	Team	We lose contact with the customer(s)	Low	High	Email CS teams to ask what happened	Zack
R10	Product	The requirements change due to either issues in the product or the client changes their mind about parts of the product.	High	High	Add new changes to the requirements table and make sure that Architure, Risk and Implementation sections are updated with new changes. Leave the description on every function implemented which will make the re-edit faster.	Everyone
R12	Implementation	Implementation of unnecessary features	Medium	Low	Delete these features, Reconsider time spent on what	Everyone
R13	Team	We fall behind schedule (e.g. spend too much time on specific features)	Medium	Medium	Assign other members to / shift the focus to more important aspects of the project.	Phoebe
R14	Technology	The software used doesn't have good support for technical issues	Low	High	Use a different piece of software or ask for help from someone with experience with the software.	Pranshu
R15	Product	Unexpected behaviours we haven't accounted for	High	Medium	Attempt to resolve or patch the issue.	Everyone
R16	Product	An accessibility need arises that we hadn't accounted for	Low	Medium	Attempt to implement features to account for the need.	Everyone
R17	Implementation	Inheriting code that is not well documented.	Medium	Medium	Try to interpret and edit the comments. If the other team is available ask for clarification on comments	Zack
R18	Implementation	Inheriting poorly structured and written code	Medium	Medium	Try to interpret and fix the code. If the other team is available, ask for clarification on code. Set up Github workflows to run automated tests to identify any immediate bugs within the code. If a bug is identified, look at external resources to	Igor

					try and fix it	
R19	Product	Inheriting poorly written deliverables (Requirements , Architect, Risks and Methods)	Medium	Medium	Try to interpret and change the deliverables. If the other team is available ask for clarification on the deliverables	Sanjna, Phoebe
R20	Technology	Shifting from GutHub repository (GHC) to GitKraken and making sure that branches are integrated properly after switching.	High	Low	Make a copy of the ripstory before switching platforms so you can revert the change if something goes wrong without harming the product.	Igor
R21	Technology	GitKraken servers going down	Low	Medium	Use repositories stored locally and consider different version control system	Igor
R22	Team	Clients finds the experience of using the product no enjoyable and doesn't find it fits there requirements for the product is ie user experience, usability, if the product has met requirements ect	Medium	High	Go back to the client and ask what parts they did not like about the product and try to change the product back to their liking. If this happens at the end of the product cycle then make notes of future changes that people could to improve the product	Everyone
R23	Implementation	Inheriting code that contains memory leaks.	Medium	High	Try to find and fix the memory leak in the code. Ask the other team where they might think where the memory leak is and how to fix it. If not then go to other resources to try and find a fix to the memory leak.	Pranshu