Risk Assessment and Mitigation

Group 26

devCharles

Ross Holmes
Sabid Hossain
Thomas Pauline
Joel Paxman
Andrey Samoilov
Louis Warren

Part a)

Before blindly adding features to the previous team's code base, it is vital to consider the risks that are involved in the project as a whole. Since we were inheriting the project from team 32 we also inherited their risk documentation. After having a look into that document we realised that it was lacking many of the risks we had previously identified in assessment 1 and so within one of our first group meetings we appended all risks we had previously identified and agreed were still relevant to assessment 2 to this document.

Once this was complete we then needed to discuss possible risks that may affect us in assessment 2. As such, each team member was given the opportunity to voice their opinions, allowing an extended list to be developed. We then analysed each risk to determine its likelihood and severity discussing them so the team can be in agreement with each one.

Importantly, each risk needed a mitigation plan to limit its impact on the overall project. Completing this step required the team to collaboratively determine what the best strategy for the risk would be. Consequently, it allowed members to look at the problem from different perspectives, ensuring that the members are critically evaluating the plans.

Following the continued development of the risks, we needed to correctly and effectively format the risk register so it's easily accessible and readable. Therefore we discussed whether we should use the system used by team 32 or whether we should use our own system developed in assessment 1. After some back and forth we decided that it would be best to use our own system as it was clearer to us which risks held what kind of importance.

The risk assessment and mitigation document is therefore presented in the risk register, which is formatted as three tables:

- **Project** - these are risks that may affect the project's schedule or the resources used by the project (including team members).
- **Product** - these are risks that would affect the product quality or its ability to be fully completed e.g. tools in the project having bugs outside of our control.
- **Business** - these are risks that would affect the team's ability to procure/develop the software e.g. obsolete technology. - this was newly added to the document

Each risk within each table is given:

- **ID** - Each risk has a unique identifier to allow for simple referencing and identification.
- **Description** - This is a description of the risk.
- **Likelihood** - This is how likely the risk is to occur - i.e. low, moderate, high.
- **Severity** - Should the risk happen this is an estimate of how much of an effect it would have again given as either low, moderate or high (Variable shows unknown impact).
- **Mitigation** - These are the steps we are taking to mitigate the risk, or steps to be taken in case of the risk occurring.

Owner - This column indicates who is responsible for mitigating the risk and reporting an issue to the group if there's a problem. This prevents risks from repeating (All team members are made aware of risks which also means that in rare cases where the owner doesn't see the risk first the issue can be still found quickly).

Something significant we also did was to assign at least two people to all tasks deemed very important and we also (for these tasks) assigned a shadow role (somebody who had no responsibility for the task but was made aware so that they could provide aid for any issues). This meant our 'bus factor' was never below 2 and with important tasks never below 3. This meant we could be prepared for as many issues as possible and meant very few unexpected issues occurred.

We also decided to add a paragraph about any risks that occurred during assessment 2, documenting what happened and how we fixed the issues as they arrived.

Part b)

Project

Risk	Risk ID	Likelihood	Severity	Mitigation	Owner
A group member misunderstands their task and completes the wrong thing	R1	moderate	moderate	We will meet regularly and stay in contact through messaging to ensure we all are clear on what we are doing, and if a different task gets completed, we just mark that one as done. We will also break tasks down into small parts on a kanban board to make it easier to keep track of what needs doing.	Louis
A member leaves the project	R2	low	high	Our regular meetings should make sure everyone is fine and working on the project. But if this should happen, we will be in contact with the lecturers and module leader.	Louis
The network fails before a member has committed their work and therefore they cannot keep up to date with the remote code which makes it challenging to merge their changes in	R3	low	moderate	We make sure we commit often and try to keep pull requests a reasonable size, not too big. But if this occurs, then we can review the merge conflicts together and work out the best way to resolve it. We will also try to make sure that our pieces of work are self contained and have minimal areas that may have merge conflicts.	Joel
A member becomes unavailable for an extended period of time and cannot communicate this	R4	low	moderate	Regular communication should mean that we all know where we are at, and if someone becomes unresponsive, we will notice this. Attempts will be made to contact the member using alternative forms of communication, and if that does not work we will let the lecturers and module leader know, while divvying up the work between the rest of us.	Ross
A member is	R5	low	high	Our regular meetings to assign work	Tom

overworked and becomes burnt out as others are doing their workload				should help ensure work is assigned evenly. We will also try to make sure that there is a second person assigned as backup, so that if people don't do their main task, there is still someone who can do it and it does not all fall on one person.	
We misjudge the time it takes to complete the project, and do not finish it before the deadline	R6	moderate	high	We make a schedule of when certain pieces of work should be done so we have a rough way to measure progress. Then throughout the project with regular meetings we can see what parts are on schedule and what we need to catch up on. There is also buffer time at the end to help mitigate this risk.	Sabid
Tasks given in the same week are split up such that they heavily rely on the completion of tasks that need to be completed in the same week	R7	moderate	moderate	Tasks that clash with each other must be given to the same person to complete and therefore can be completed in order. If the tasks are too heavy for one person to complete in one week then the tasks must be split into two weeks instead.	Joel

Appended

Risk	Risk ID	Likelihood	Severity	Mitigation	Owner
Team member's knowledge of the codebase is not enough to contribute	R8	high	moderate	Arrange a meeting to discuss the codebase with the team member	Andrey
A file is accidentally deleted or corrupted	R9	low	high	Keep updating the remote repository with github, keep extra copies of our important documents in our personal computers that we regularly update	Ross
Discord, our main messaging tool, goes down	R10	low	high	Make sure we have multiple ways of contacting each other (e.g. via e-mails)	Joel
A team member does not have the correct version of our dependencies, and therefore cannot contribute to the implementation	R11	moderate	low	Make use of centralised dependency management (Gradle) and make sure everyone's base JDK is the same version	Andrey
A team member's computer breaks	R12	moderate	moderate	Make sure everyone has access to another computer they can work on if necessary (e.g uni computers)	Tom
A group member commits directly to main, losing changes made by others or introducing merge conflicts	R13	Low	Variable	Lock the main branch to only accept github pull requests from other branches.	Andrey
Fail to clarify everything during a client meeting	R14	low	moderate	Contact the client to further clarify details.Prepare questions in advance in order to be sure of having every required detail.	Sabid
Team member's computer crashes losing unsaved changes	R15	low	moderate	Regularly commit changes to the local source control system	Sabid
Requirements being added during Project progress (Scope Creep)	R16	low	moderate	Get a clear understanding of what the client wants at the beginning of the Project	Everyone

Product

Risk	Risk ID	Likelihood	Severity	Mitigation	Owner
The project becomes less maintainable due to bad code quality (e.g. big classes, confusing package names, bodge fixes)	R17	moderate	moderate	When working on code, we will create pull requests to encapsulate tasks undertaken by team members. These pull requests can be reviewed by other team members to ensure code quality. We can discuss decisions on code either in the pull request or in person / via discord. The CI pipeline should also help identify bad code style.	Andrey , Ross
Assets that are being used are not allowed to be included in open source projects	R18	low	moderate	Before using any assets we will check the licence for usage rights. If that is clear and says that we can, we will use it. If it says we cannot use it, then we will not. If it is unclear, then we can either look for something else, or we will get in contact with the asset creator to ask for permission.	Louis
A library being used becomes unavailable or deprecated in the middle of the project	R19	low	high	We will try to choose popular open source libraries that have a large community around them to help avoid this risk. However, should this still happen, we can check to see if the last released version is sufficient for our use case, which it should be, or we have a look for alternatives that would also work.	Ross

APPENDED

Product Table

Risk	Risk ID	Likelihood	Severity	Mitigation	Owner
The implementation has a progress halting issue	R 20	low	moderate	Help the team member resolve the problem	Andrey
The game runs slowly on certain computers	R 21	low	moderate	Test the game on multiple computers with different specifications, and make changes if necessary	Joel
The libraries used in the implementation don’t have/don't have enough documentation, and the	R 22	high	moderate	Check documentation of libraries prior to picking them to be used in the project, if that's not possible look for alternative material (e.g. open source	Ross

team is struggling to implement them				projects using the library)	
The game doesn't resize well to some window sizes	R 23	moderate	moderate	Test the game in various common window sizes and make changes if necessary	Andrey
Part of our code accidentally infringes upon someone's copyright	R 24	low	Variable	Depending on how the infringing code is licensed, the code can be kept as is with a copyright notice, or removed entirely if the licence is not compatible with the project	Tom
Architecture does not support a required feature	R 25	moderate	high	Consider all possible architectures with their advantages and drawbacks.	Andrey , Louis, Ross
Hosting service becomes unavailable	R 26	moderate	moderate	Prepare alternative hosting methods, including private servers.	Tom
Compromising on design to complete function as quick as possible	R 27	low	moderate	Regularly check non-functional requirements during development.	Joel
Lack of support for users post completion	R 28	high	low	Design simple to understand interface and provide clear instructions	Tom
Bugs in code that are difficult to detect but appear with frequent use	R 29	moderate	moderate	QA / test to decrease frequency of issues	Louis, Ross

APPENDED

Business Table

Risk	Risk ID	Likelihood	Severity	Mitigation	Owner
The software does not perform well on client's computer	R 30	moderate	high	Make sure we use optimised libraries and the codebase is overall efficient with its resources. Clarify with client about specifications of machines expected to run the game	Everyone
University pauses teaching due to industrial action	R 31	moderate	high	Access alternative teaching material	Everyone

Risks Occurred:

- Several times during assessment 2 R1 occurred, this generally occurred when a team member was unsure how to proceed with a task and so would realise later on that the work must be redone. To mitigate this we encouraged everyone in the team to ask for help if they are unsure about a task, which meant after the first occurrence this rarely ever happened again.

- 20/02/2023 R8 has occurred. This was due to the fact that Sabid, Joel and Tom did not contribute much to the code side of assessment 1 and therefore struggled to understand how team 32's code base worked. Therefore we created a senior/junior developer role system (as described in method selection and planning) so that we could work on code in pairs. Allowing Sabid, Joel and Tom to learn faster and get stuck into some testing. This greatly reduced the impact R8 had on our project and hopefully means the likelihood of this happening again is close to zero.

- 02/05/2023 R9 occurred. This was due to the fact that a team member accidentally deleted half the documents. This meant all of the testing report, change report and risk assessment and mitigation were completely and irrevocably deleted. However the team member had saved these files on his local computer several hours later, and during those hours nobody worked on those specific files, meaning that no progress was lost. To continue to mitigate this risk in the future, a new mitigation plan was added, specifically: "keep extra copies of our important documents in our personal computers that we regularly update". The whole group has been recommended to do this, which should now have a significantly smaller impact since all work cannot be lost with just one click.