Risk assessment and mitigation

Cohort 3 Team 7 (Yetti)

Bak, Bartek
Burberry, Katelyn
Collins, Lucy
Ganley, Joe
Keegan, Josh
Kirkham, Katharine
Mayall, Daniel
Sawdon, Theo

As developing a system can be a very versatile process, prone to changes and adversities, being able to predict and mitigate risks is very important. If an issue arises that could have a big enough impact, it could completely halt development of the project. We are working within a certain timeframe, with a hard set deadline, and so this happening could be detrimental, and could result in us not meeting our targets in time, which could be a great hindrance on the customer, as they may require the software to be able to function properly by a certain point. Being able to predict and plan around risks is imperative for assuring a smooth development process, and making sure that the project is complete on time.

We could start by trying to imagine every possible scenario, and thinking of what possible setbacks we might encounter (risks). From this we could then estimate the likelihood of this happening, and the impact it could have on the development. If the risk is likely to happen, or could have a big impact on the project, we should plan around this, creating plans to ensure it doesn't happen, and contingencies, just in case it does.

To do this, we need a way to effectively organise and plan for risks. As such, we have decided to do our risk register in the form of a table. With this, we can organise what might happen and what we are doing about it, in a very clear, concise and easy to understand manner. Using a table helps us to group which actions are associated with which risks, making mitigating these risks much easier. Overall, this format of tracking risks should mean they are much easier to track, and by extension easier to deal with. As a result, any issue that arises will be less likely to negatively impact the team and the development of the project.

Key: L - Low, M - Medium, H - High

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
Team member stops contributing (R000)	L	H	Certain aspects of the project simply won't get done, as everyone has already been assigned certain tasks to do.	Conduct weekly check-ins to ensure everyone is coping. Also assign a shadow member to ensure the work is being done, and to help out where possible.	Redistribute the required tasks fairly across the rest of the team.	Everyone
Remote repository goes down (R001)	L	H	No one will be able to access files related to the project i.e. code or planning documents.	Keep backups of the repository across a few locations.	Find the most up to date local copy on someone's machine, work on that in the meantime, then consolidate changes when the repository is back up.	Everyone

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
Development software in use stops working (R002)	L	M	Development on the implementation would completely halt, and those coding would not be able to effectively work.	Ensure research has been done into many different options just in case we need to swap.	Try to use a different piece of software that is known to support the current code base.	Implementation team (Lily, Alex, Taro, Fares & Guilia)
Hardware in use stops working (R003)	L	L	All forms of contribution towards the project would stop completely, meaning we may miss the deadline for the product.	We should each have access to backup pieces of hardware that we can fall back on if any machines fail.	Make use of publicly available hardware in the meantime (i.e. library computers, university machines etc)	Everyone
Customer changes their mind on a certain aspect of the product (R004)	M	M	We would have to stop development on anything that doesn't meet the new requirements, and lose any progress and time spent on these now unnecessary components.	Keep regularly meeting with the customer to ensure any changes to their requirements are caught quickly, so no more time is wasted.	Immediately stop production on those parts that do not fit the new criteria, and start planning on how to work to new constraints. Once a solid plan is formed, immediately start working towards these new goals.	Everyone
Feature doesn't fit what the customer envisaged (R005)	H	M	Similarly to if the customer changed their mind (R004), we'd lose any progress made towards this component of the project. This could have an impact on our ability to meet the customer's deadline. With	Regularly meet with the customer, and double check with them to clarify exactly what they want (even if the questions may seem redundant). Use prototypes and mockups for new features	We would take the customer's feedback into account, clarify exactly what they want and how they want it, immediately start planning how to do this instead, and start production on the newly	Everyone

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
			new features like leaderboards and achievements, ensuring alignment is critical.	like achievements and leaderboards.	clarified requirement.	
Development library in use for the development loses support and stops working (R006)	L	M	Just like if the software itself stops working (R002), development of the code would stop, and we'd lose some progress made on the coding.	Maintain a decent knowledge on different libraries, so we can swap over if need be.	As most libraries follow a similar paradigm, refactoring code would not be too much of a challenge. We'd have to spend time changing what has already been made to work with this new library.	Implementation team (Lily, Alex, Taro, Fares & Guilia)
Another company takes legal action against us for the project (R007)	L	M	This could have financial and academic impacts on the project if a company pursues action for license violations. In a university context, impact is primarily on grades rather than business operations.	Meticulously find the licenses for any asset or software used to ensure we are able to use it as we wish. If any do not fit our use case, do not use it.	Cease production immediately and conform with any requests (if justifiable) from the other company. Replace problematic assets with properly licensed alternatives.	Everyone
The quality of code produced is poor due to certain constraints (R008)	M	L	This could lead to a poor running product for the customer. In a learning environment, the impact is lower as this is	Host regular code review meetings, so we can all have input on what might be best. Follow coding standards and	Allocate more people onto the implementation team, and refactor all the code to meet a certain much better standard.	Implementation team (Lily, Alex, Taro, Fares & Guilia)

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
	🟨	🟩	expected and can be addressed through iteration and feedback.	best practices from the start.		
An unexpected problem arises (R009)	L	M	As this problem is not within the planned risks, the effect could be larger as we will need to establish a plan, slowing down the project	Proactively add possible risks to the register as we think of them	Develop a plan to mitigate this new risk, and execute it in a timely manner, with input from the whole team to ensure it is the correct approach.	Risk Management Team (Alex)
Deployment goes poorly (R010)	L	L	The software won't be published properly, which could lead to issues with the client if they need to hit certain deadlines.	Ensure the deployment is effectively planned.	Analyse what went wrong, and try again. Possibly using a different method.	Everyone
Leaderboard data persistence issues (R011)	M	M	Leaderboard data could be lost between game sessions, frustrating players who lose their scores and rankings. This undermines the competitive aspect of the game.	Implement robust local file storage with proper serialization. Test save/load functionality across different scenarios (crashes, normal exits). Create backup systems for score data.	If data loss occurs, implement a recovery system from backup files. Add validation to prevent corrupted save files from causing crashes. Inform users and offer fresh start with apology message.	Implementation team (Lily, Alex, Taro, Fares & Guilia)
Achievement system bugs (R012)	M	L	Achievements might unlock incorrectly, not unlock at all when earned, or unlock	Design clear achievement criteria with well-defined trigger conditions.	Debug and fix the specific achievement logic. Create admin tool to manually adjust	Implementation team (Lily, Alex, Taro,

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
			prematurely. This reduces player motivation and creates confusion about game progression.	Implement comprehensive testing for each achievement. Create achievement validation system to prevent false unlocks.	achievement status for affected players. Issue patch update with fixes and explanation.	Fares & Guilia)
Restart game functionality fails (R013)	L	M	The restart button might not properly reset game state, causing carryover of previous game data, crashes, or requiring users to manually close and reopen the application.	Implement proper state management with clear initialization and cleanup methods. Test restart functionality extensively in different game states (mid-game, end screen, pause menu).	Identify which game state variables aren't being properly reset. Implement a complete state reset function. Add fallback option that forces complete application restart if soft reset fails.	Implementation team (Lily, Alex, Taro, Fares & Guilia)
Leaderboard display issues (R014)	M	L	Leaderboard might display incorrectly sorted scores, show duplicate entries, or have formatting issues that make it hard to read. This reduces trust in the scoring system.	Implement proper sorting algorithms with unit tests. Design UI mockups before implementation. Test with various data scenarios including edge cases (ties, very high scores, many entries).	Fix the sorting or display logic immediately. Add data validation to prevent duplicate entries. Consider temporary simplified display while developing comprehensive fix.	Implementation team (Lily, Alex, Taro, Fares & Guilia)
Single map becomes repetitive (R015)	H	M	With only one map instead of multiple maps, players may find the game repetitive quickly,	Add variety through randomized event placements and varied gameplay	If feedback indicates the single map is too repetitive, consider developing additional map	Architecture, Implementation teams

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
			reducing engagement and replayability. This conflicts with having a leaderboard that encourages repeated play.	scenarios within the single map. Implement different difficulty modes that change enemy behavior or time constraints. Design achievements that encourage different playstyles.	layouts as post-release content. Alternatively, implement procedural generation elements to create variation within the existing map structure.	
Specification completion delays (R016)	M	M	Requirements or other deliverables not completed on time, causing bottlenecks for dependent work streams. Implementation cannot start without clear specifications.	Set internal deadlines ahead of actual deadlines. Break specifications into smaller tasks with checkpoints. Assign clear ownership of each specification section.	Reallocate resources to specification work. Deliver minimum viable specification to unblock implementation. Work in parallel where possible with preliminary specifications.	Requirements team (Jacob, Taro & Fares)
Learning curve for new technologies (R017)	M	M	Time taken to learn Java 17, LibGDX, or game development frameworks could delay implementation and reduce code quality. Team members have varying levels of experience.	Allocate time upfront for tutorials and experimentation. Pair inexperienced members with those who know the technology. Create internal knowledge sharing sessions and documentation.	Simplify technical approach or use more familiar alternatives where possible. Focus on delivering core features first. Seek help from module staff or external resources.	Implementation team (Lily, Alex, Taro, Fares & Guilia)
Underestimation of task complexity (R018)	H	M	Tasks take significantly longer than estimated, causing	Add buffer time to estimates (multiply by 1.5-2x). Break down tasks into	Prioritize must-have features over nice-to-have. Negotiate with	Method Selection and Plannin

Risk (ID)	Likelihood	Impact	Consequences	Mitigation	Contingency Plan	Owner
	🟥	🟨	deadline pressure and potentially incomplete features. Common in game development due to unforeseen technical challenges.	smaller chunks for better accuracy. Review estimates as a team. Track actual vs estimated time to improve future estimates.	customer on scope reduction if necessary. Increase development time by reallocating team members or extending work hours.	g team (Lily)