

# Risk Assessment & Mitigation

## Risk Rating

To determine the overall damage to the project that a risk may have, we applied the likelihood and impact of the risks via a risk matrix to abstract a 'risk rating', ranking them from Low, Medium, High and Critical:

Likelihood	Impact				
	Negligible	Minor	Moderate	Major	Severe
Very Likely	Medium	Medium	High	Critical	Critical
Likely	Low	Medium	Medium	High	Critical
Somewhat Likely	Low	Medium	Medium	High	Critical
Unlikely	Low	Low	Medium	Medium	High
Very Unlikely	Low	Low	Low	Medium	High

### Impact Definitions

- Negligible - The impact of the risk will only delay the project by a small number of hours
- Minor - The impact of the risk will delay the project by 1-2 days
- Moderate - The impact of the risk will delay the project by 3-5 days
- Major - The impact of the risk will delay the project by 1-2 weeks
- Severe - The impact of the risk will delay the project by 2 or more weeks

### Likelihood definitions

- Very Likely - Occurs once a week
- Likely - Occurs once in two weeks
- Somewhat Likely - Occurs once in three weeks
- Unlikely - Occurs once a month
- Very Unlikely - May occur once during the project completion time

## Risk format justification

The risks have been formatted into a table consisting of 3 columns; Risk name and description, Mitigation, and Overall Risk. The rows are sorted into descending order of overall risk rating so risks that would have the biggest impact are clearly visible and distinguishable. Risks are ranked by evaluating negative impact to project and likelihood of possibility. The table suggests mitigation strategies, that would help effectively solve potential problems.

# Risks

Risk	Mitigation	Overall Risk
Ineffective communication between group members	GitHub and Slack allows effective communicate between group members. Failure to communicate will have a reason. Schedule a special meeting with an elected mediator to allow poor communicators to explain their issue(s) and re-integrate with the team.	Somewhat Likely/Severe
Project team misunderstood requirements	Following Scrum methodology frequent team meetings are held and requirements are often reviewed. Team maintains consistent communication via Slack and through face-to-face means.	Somewhat Likely/Severe
Team members missing important team meetings	Communicating via Slack, ZenHub. Team members are provided with all necessary information that they missed. If that person is still missing team meetings, the problem will be reported to lecturers to be solved via official channels.	Likely/Major
Inadequate architecture, performance and quality	Simulation created and benchmarking results are evaluated. Prototype is created to allow tuning of some required details.	Unlikely/Severe
Deletion of project code	GitHub minimises the likelihood of complete loss of code, regular backup via forking to personal repositories increases redundancy and cloning of the repository to an offline location before major changes further reduces risk.	Very Unlikely/Severe
Final build has low quality	Prototype developed to test functionality. Disciplined development process is used. Technical reviews are used on all requirements, designs, and code. Test planning assures all functionality will be covered by system testing. System tests are performed by independent testers.	Unlikely/Severe Impact
Poor project planning	Gantt charts alongside our agile development model (Organised by ZenHub for GitHub) allow us to establish a clear overview of bottlenecks in the project and then refactor the plan as we go.	Somewhat Likely/Major
Certain team members lack of specialized skill required by the project	Inexperienced team members should find time and resources to improve needed skills. Different tasks should be allocated to team members that have skills to accomplish it.	Likely/Moderate
Project team need to acquire new skills for	Work plan is rearranged so it includes some time to gain required skills. No one is left struggling feeling	Likely/Moderate

the project and it leads to low productivity	they can't work on the tasks.	
Poor productivity	ZenHub for GitHub is used. It shows visual display of effort that people put in. It indicates team members that don't contribute enough. Therefore it allows to locate and solve an issue fast.	Somewhat Likely/Moderate
Member of team is sick and can't participate in further work	Program code is submitted in GitHub therefore the rest of a the team can continue working. No member will ever have a 100% share in a mission critical task. Frequent commits and clear comments will allow members to compensate for illness or absence.	Unlikely/Moderate
Team member is underperforming	Team is rearranging workload and providing any help if it is necessary, possibly through mediation meetings	Unlikely/Moderate
Lack of effective project management	Agile software development technique is going to be used to manage product development	Unlikely/Moderate
Project progress not monitored closely enough	Frequent team meeting are held to review progress. Gantt and burn-down charts are used to give a close view of project progress	Unlikely/Major
Project milestones not clearly defined	Milestones analysed within the planned scrum meetings, milestones can be re-arranged and definitions updated within a week of issues arising.	Somewhat Likely/Moderate Impact
Inexperienced project managers	If needed different project leader is elected by the team members	Unlikely/Major
The wrong software functions are developed	Customer requirements are analysed. User survey reviewed. Prototype tested. Discuss and resolve software issues in meetings.	Somewhat Likely/Moderate
Continually changing requirements	Changes are accepted as a fact of software projects. Prioritisation sessions are scheduled that allow changes to proceed.	Very Likely/Minor
Incorrect system requirements	Task descriptions are reviewed frequently. Note that while our requirements are correct in the context of our current concept, that concept may change in the future.	Very Likely/Minor
Gold plating (added features are not useful)	Requirements are frequently reviewed and tasks are worked on in order of priority. Prototype is created and tested.	Unlikely/Moderate
Functionality is complex to implement	Team meeting is held to discuss the problem and find solution. Research is done on programming techniques or available software that would allow to solve the problem.	Somewhat Likely/Moderate
Problems to	The GitHub pull request system combined with a	Somewhat

integrate separate pieces of code	potential sit down meeting with all developers affected by the conflict in question will likely ensure a smooth development process.	Likely/Moderate
Overriding each other's work	Over-writing other's work should only occur after proper code review enforced by the pull request system. This review process allows us to ensure any changes removing or changing others' work to be intended and functional.	Unlikely/Major
Internet access is compromised for a prolonged period of time	Whenever a pull request is approved, all team members should clone the repository's master branch so that they can work on the game without requiring internet access. 4G data-plans can be activated if necessary to acquire internet connections.	Very Unlikely/Severe
Another user gains unauthorised access to the GitHub repository	Take care to avoid sharing login credentials for any of the university/GitHub accounts that can access the project's resources.	Very Unlikely/Severe
Team member's local copies of project files are compromised after having been worked on heavily (prior to committal)	Submit commits immediately after each task is completed so that all completed work is backed up frequently. Use Google Drive to write up documentation and automatically save it on Google's servers.	Unlikely/Major
Created software code has a bug	Software code is saved on GitHub. It allows to access previous version of code and locate where the bug was introduced. A possible integration of Travis CI will allow immediate testing and debugging of code.	Very Likely/Minor
User interfaces do not fit needs	Prototype is created, scenarios are development. Customer description reviewed.	Likely/Negligible
Inadequate estimation of required resources	Frequent meetings are held. If needed additional resources can be allocated. Tasks can be divided between more group members if the velocity of one member is dropping due to tackling a large task alone.	Unlikely/Minor