Assignment 2
Requirements, risks, and design

Team

Nicholas Wong – njwon4@student.monash.edu
Wanyu Yin – wyyin1@student.monash.edu

FIT3140 Advanced Programming

Overview

For this risk document we will be measuring the Impact and Likelihood tables found in the lecture notes for FIT3140. They are both on a scale from 1 – 5 as shown in the two tables below.

Impact	**Color Code**	**Description**
5 – Extreme		• May result in project failure • Budget overrun could exceed 50% • Project late by more than 50% • Could affect the ability of the organization to continue functioning
4 – High		• May result in significant impact on expected features, functionality or quality • Budget overrun exceeding 25% • Project late by more than 25%
3 – Moderate		• Significant effects on the project are unlikely • Budget overrun exceeding 10% • Project or subsystem late by more than 10%
2 – Nominal		• Does not require monitoring or review • Budget overrun exceeding 5% • Project late by more than 5%
1 – Minimal		• Little or no impact on any aspect of the project • Should be reviewed quarterly

Likelihood	Color Code	Description
5		Very likely • 91-100%
4		Likely • 61-90%

3		May occur • 41-60%
2		Unlikely • 11-40%
1		Very unlikely • 0-10%

Risk Categories

There are five main areas of risk that relate to this project. They are:

- Team members and management risk
- Resource management risk
- Hardware environment risk
- Software environment risk
- Requirements risk

Team Members/Management Risk

Identification	**Likelihood**	**Impact**	**Monitoring**	**Avoidance**	**Mitigation**
Team member gets sick	2	3 – Not likely to significantly impact the project, but might cause delays in delivering requirements.	Regular review of how each team member is feeling in a stand-up before work. Thorough health checks are not required.	We will maintain our mental and physical health by eating well, sleeping well, and not overworking to the point of exhaustion.	If the illness is not severe, the team member can take a rest for a day or two but get back to work as soon as possible. To help minimize the impact, the other team member should continually update the sick member on the progress of the project made in their absence. For severe illness (very unlikely),

					the team will need to find a replacement or complete the project solo.
Team member drops out of the unit	2	5 – Will severely affect the project. Can make the project significantly late, or cause an outright failure to deliver what is required.	Team members are thinking about dropping out of the unit.	Hard to avoid completely, but team members need to think and plan carefully before enrolling in this unit to minimize the chances of dropping out.	The team member that drops out should help in finding a replacement team member. Alternatively, the team could merge with another team. As a last alternative, ask the tutor if it can be done solo.
Team member task/time management issues	4	3 – Will slow down the progress of the entire project. Not likely to cause outright failure, but can delay deliverables.	Using Trello as an online sprint board and Git with version control to track the progress of each team member.	At the start of every meetup day (Monday, Wednesday, Friday) we have a short stand-up to describe what each team member has done, what they will do, and if they have any difficulties.	If a team member falls behind in their work, they need to seek help as soon as possible. The other team member needs to be notified and can dedicate time to help with the backlog of work.
Tutor or lecturer unresponsive	1 – Classes on Monday, Wednesday, and Friday make this very unlikely	3 – Since some problems are hard for students to solve themselves, this issue can slow down the progress of the project. It is unlikely to cause complete failure, but can impact the quality of the work or delay deliverables.	The tutor or lecturer has not replied for more than two days.	We need to start the assignments as soon as possible in order to discover problems as early as possible. This will leave enough time to seek help from the tutors or lecturer	If a tutor is unresponsive, then it might be better to seek another tutor for help

Team member conflict	1	4 – Will significantly slow down the progress of the project and affect the quality of the work. Has a chance of leading to complete project failure if not dealt with.	A team member is being overly aggressive, controlling, or otherwise causing conflict.	We should strive to maintain a fun and productive working atmosphere	If conflict arises or could potentially arise, we need to talk it out and handle it like adults. If the conflict is unresolvable then the we need to consult the lecturer as soon as possible.
Team members unable to meet up	3	2 – Will not slow down the progress of the project significantly.	A team member does not show up to class or meetings	Plan the meeting time in advance and ensure everyone is able to make it.	Plan another meeting time. If one still cannot meet up during the day, then use Skype of Google+ Hangouts to communicate effectively.

Resource Management Risk

Identification	**Likelihood**	**Impact**	**Monitoring**	**Avoidance**	**Mitigation**
Time	4	5 – Since time is the primary resource for this project, mismanaging this resource can make it severely late. This can lead to outright failure.	We will be using Trello as an online sprint board and version control using Git to track the progress of each team member and the team as a whole.	To avoid time resource issues, we need to keep track of the tasks that have been done and what still need to be done in the current sprint. Careful sprint planning needs to be done before each sprint, including estimating the difficulty of individual tasks. At the	If the team has fallen behind in a sprint, then they need to make up for lost time as much as they can. This may include working for longer. The goal is to create a working product by the end of the sprint, so lower priority functionality can be cut. At the end of the sprint, a Retro will be conducted to discuss what went wrong with time management,

				start of each work day, a stand-up will be done so that each team member can talk about what they've done, what they plan to do, and if they had any problems.	and how it can be improved for next sprint.

Hardware Environment Risk

Identification	**Likelihood**	**Impact**	**Monitoring**	**Avoidance**	**Mitigation**
Laptop Failure	1	5 – Lose part of or the entire project. This will severely impact the project and possible cause complete failure.	N/A – There is little we can do to monitor the status of our hardware without incurring unnecessary overhead. Therefore we will not worry about monitoring.	We will be storing the majority of our work online using Google Drive for documentation and Git for code. This means we need to continually update, commit, or push our work so the online copies are as current as possible.	Since the project data will be backed up online and on the other team member's computer, this will not be a big concern. If the laptop is not repairable, then work can be done using the University lab computers.
Loss of data (hardware)	1	5 – Lose part of or the entire project. This will severely impact the project and possible cause complete failure.	N/A – See above	See above	Since the project data will be backed up online and on the other team member's computer, this will not be a big concern.
Lack of internet connection	2	3 – Cannot update team members or save new work	N/A – See above	Pay the home internet bill and the University fees on time. Most of the work	As a last resort, using mobile hotspots or go to free-wifi spots to get connection.

		online. Can slow down the progress of the project, but unlikely to cause any significant impacts.		will be completed either at home or at university.	

Software Environment Risk

Identification	Likelihood	Impact	Monitoring	Avoidance	Mitigation
Loss of data (software)	3	5 – Lose part of or the entire project. This will severely impact the project and possible cause complete failure.	Need to constantly check that local data copies are not corrupt and are up to date. Also need to make sure that corrupted copies are not pushed onto the online copies.	We will be storing the majority of our work online using Google Drive for documentation and Git for code. This means we need to continually update, commit, or push our work so the online copies are as current as possible.	Since the project data will be backed up online and on the other team member's computer, this will not be a big concern.
Kivy issues	4	4 – issues include being unable to implement specific functionality (like drag and drop) and can significantly delay the project.	The team need to keep working towards the goal of the current sprint. If a member has been stuck on a specific feature for two or more coding sessions then we have an issue.	We will be doing a number of pre-emptive coding spikes to avoid potential Kivy issues. We will also do a stand-up at the start of every work day to raise any issues or blocks that we are experiencing	If we come across any significant Kivy issues, we need to identify them and tackle them early. This includes referring to resources like the Kivy docs, Stack Overflow, tutors or lecturer. We can also do additional coding spikes on an ad-hoc basis during or in between sprints.

Issues from lack of experience with Git	3	3 – Possible loss of data or the need to rollback due to a mistake can cause delays. However, this is not likely to significantly impact the project overall.	Git does not perform as we expect it to, or we are unable to understand what is going wrong	Reading the Atlassian Git “Getting Started” tutorial will give a good foundation to understand Git. https://www.atlassian.com/git/tutorials A technical spike will also be useful.	If any issues occur, we will refer to the Git documentation. Other resources include Stack Overflow, online tutorials etc. Issues need to be identified and dealt with early.
Git merge conflicts	1 – since there are only two people in the development team, the likelihood of needing to merge/rebase is very small	4 – Potential loss of code can cause significant setbacks if merging/rebasing is done incorrectly. This can significantly affect the project by causing delays.	When Git raises a merge conflict when updating a local copy of our code, we need to proceed with caution	Simple practices like messaging each other when modifying a common file, continuously committing, pushing and pulling, etc. should be enough to avoid most common Git merge/rebase issues.	If a conflict arises, then we need to notify the other team member immediately. We then handle the conflict carefully and collaboratively with the other team member. If mistakes are made, as a last resort we can rollback to a previous version with minimal data loss (assuming committing/pushing has been frequent)

Requirements Risk

Identification	Likelihood	Impact	Monitoring	Avoidance	Mitigation
Not correctly completing requirements from requirements specification	4 – The requirements given to us are incomplete by nature	5 – If we do not come up with a relatively complete set of requirements, then this will severely affect the quality of the project. It will make the difference between a successful project and a failure.	We need to detect incomplete or incorrect requirements as early as possible. This involves collaborating with the product owner continuously and asking lots of questions.	Continuous contact and collaboration with the product owner during the requirements gathering stage AND during development / testing should help to avoid incomplete or incorrect requirements.	If we find an incomplete or incorrect requirement in the specification, then we need to sit down with the product owner as soon as possible to mitigate risk early. The longer we leave it, the greater the impact will be.
Changing requirements	2 - given the nature of this university project, significant changes are not very likely to occur (as Robyn explained in the lecture)	5 - Depending on how late in the project this happens, the impact has the potential to be very significant. Delaying the project deadline and putting further budget on the time budget in the worst case scenario.	Monitoring the requirements will be done using the Trello backboard to track the progress of each task. Changes will come from the product owner	Avoiding change in requirements completely is hard to do. The best we can do to avoid changes is to keep in contact with the product owner and keep checking that the project is in line with the vision statement	Different methods of mitigation will be taken depending on the size of the requirements change. Generally, smaller changes may can be accommodated to the best of our ability. Larger tasks will not be as easy to do. If the change is not in line with the vision statement, then we need to raise this with the product owner and negotiate as appropriate.
Misunderstanding	3	4 - Misunderstandings	Requirements will be monitored using the	To avoid misunderstanding we	This is similar to any change in requirements. We need to talk

requirements		and assumptions will make the project late and affect our budget negatively	Trello board.	need to ask lots of questions to the product owner about the requirements in enough detail that they become unambiguous as much as possible	with the product owner as soon as possible to identify and minimize the impact on the project.
Requirements creep	2	4 - Runs the risk of making our application bloated and either increasing the iteration/sprint duration, or decreasing the amount delivered at the end of each sprint. Has the overall effect of delaying the deadline	The product owner gives us requirements in the middle of an iteration that they want implemented in the same iteration	During iteration planning, we will follow the convention that no new stories will be added during the iteration.	If the product owner gives additional requirements during an iteration, we need to notify them that the requirement has been acknowledged, but will be acted upon after the iteration has come to completion.