IMS Project // Risk assessment & matrix

Author: Cameron Ofoluwa

	Severity: 1	Severity: 2	Severity: 3
Likelihood: A	A1	A2	A3
Likelihood: B	B1	B2	B3
Likelihood: C	C1	C2	C3

Impact Risk	Risk Statement	Response strategy	Objective	Likelihood	Impact	Risk Level
GitHub	Any source code pushed to GitHub could potentially contain information that hackers would find useful when trying to a maliciously alter the project. The source files could potentially contain hard-coded login credentials which could allow for data leaks.	Use stronger passwords and usernames than just “admin” or “root” and keep them regularly updated.	Reduce the likelihood of hacking and data leaks.	B	3	B3
Poor management	Maintaining poor management throughout the project lifetime can result in a dire outcome for the whole project. Whether this be running out of time or mixing up priorities with tasks needing to be completed.	Work with a project management dashboard such as Jira to help keep management to a professional level.	Maintain a high level of management throughout the project to meet deadlines.	A	1	A1
Unlearn, yet required skill	Features & parts of the programme may require certain skills that I do not know, nor have the time to learn (as the project will not be finished in time).	Ensure to use strategies and technologies that I know prior to the project, or only ensure I can fit learning into my Sprints to make sure the project does not go over.	Ensure the programme is worked on by the correct person(s) with the correct knowledge.	B	2	B2

Unavoidable risks	Unavoidable, real world risks that would affect the project and potentially stop it from being finished on time such as power cuts and hardware failure.	Have multiple options in terms of hardware (PCs, laptops, mobile phones) and have more than one spot in a different geographical location in case anything was to happen such as a power cut or internet maintenance.	Ensure I have multiple backup ways to access project files to ensure I can complete the project by the deadline.	C	2	C2
Poor code & technical risk	Poor code can lead to technical failure throughout the programme leading to poor performance and a bad end product the client is not happy with.	Use proven coding strategies, tactics and practices to ensure a high quality of code. Use programmes such as SonarQube to help clean up code and Junit to help testing the programme to ensure it runs smoothly with no errors.	Improve the quality of code written for the programme, ensuring the likelihood of the project working as it should.	A	1	A2