## Risk Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | Likelihood (L) | | |
| 1 - Low | 2 - Medium | 3 - High |
| Severity (S) | 1 - Very Low | 1 | 2 | 3 |
| 2 - Low | 2 | 4 | 6 |
| 3 - Medium | 3 | 6 | 9 |
| 4 - High | 4 | 8 | 12 |
| 5 - Very High | 5 | 10 | 15 |

***Figure*** *A risk matrix showing how Risk Impact can be derived from estimates of Likelihood (L) and Severity (S).*

### Risks

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Risk | Raw Risk | Mitigation | L | S | Residual Risk |
| Hard disk failure | 1 x 5 = 5 | Proactive: Make use of cloud storage for key files relating to the project wherever possible. Maintaining copies of files across multiple storage platforms (e.g. both OneDrive and GitHub) will further reduce the risk of losing significant amounts of work. | 1 | 1 | 1 x 1 = 1 |
| Poor time management | 3 x 4 = 12 | Proactive: Refer to the project Gantt chart, deliverables and milestones to understand whether the project is ‘on-schedule’. Maintain a progress log & aim for transparent communication of progress with the project supervisor. | 2 | 2 | 2 x 2 = 4 |
| Poor project planning | 2 x 3 = 6 | Proactive: Try to break tasks down until they are shorter than two weeks in duration. Discuss these tasks with the project supervisor and agree on clear milestones to indicate progress. | 2 | 2 | 2 x 2 = 4 |
| Final product fails testing due to bugs | 2 x 5 = 10 | Reactive: Since an iterative development methodology is in place, select a previous iteration of the software to be used as the final version. Fix the bug if sufficient time is available. | 2 | 2 | 2 x 2 = 4 |
| Insufficient documentation for use of the software | 2 x 5 = 10 | Proactive: All released iterations of the software must include a clear explanation of the software functionality and directions for use. This information should be in the form of a ‘README’ file (.txt or .md), and/or available within the software interface. | 1 | 3 | 1 x 3 = 3 |

(continues on the next page)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Risk | Raw Risk | Mitigation | L | S | Residual Risk |
| None of the software iterations produce valid solutions to the MSA problem | 1 x 4 = 4 | Proactive: Ensure that producing valid solutions to the MSA problem is one of the first requirements to be satisfied by a software release. This should mean that a functional tool is always available to fall back on, while following iterations can aim to improve the performance and solution quality. | 1 | 4 | 1 x 4 = 4 |
| Personal circumstances (e.g. hospitalised) disrupt productivity | 1 x 5 = 5 | Reactive: Communicate these circumstances with the project supervisor as soon as possible. Discuss how the project plan can be adapted if necessary and get in touch with the student services. | 1 | 4 | 1 x 4 = 4 |
| Project supervisor becomes unavailable | 1 x 4 = 4 | Reactive: Discuss this circumstance with the module lead if this situation arises. Not sure what to put here | 1 | 2 | 1 x 2 = 2 |
| Social restrictions due to an epidemic impact ability to work effectively | 1 x 3 = 3 | Reactive: Work remotely using cloud services. Check whether completion of all primary objectives is still feasible and discuss making revisions to the project plan if necessary. | 1 | 2 | 1 x 2 = 2 |
| Scope creep | 2 x 3 = 6 | Proactive: Work with the project supervisor to create a comprehensive set of primary and secondary objectives for the project. Ensure that any work taken on aligns with these pre-defined objectives. | 1 | 2 | 1 x 2 = 2 |
| External data sets (for testing) become unavailable | 1 x 3 = 3 | Reactive: Generate synthetic test data for testing by writing a script or create a set of simple testcases by hand. Test the software using this data instead and communicate this compromise. | 1 | 2 | 1 x 2 = 2 |