|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Response strategy | Likelihood | Impact |
| Loss of relevant data | Keep backups and take fresh back ups regularly | Moderate | Moderate |
| SQL Injection | User inputting SQL into the system. An issue will become more likely when user input is added to the table. | High | High |
| User data theft | All data holding personal information on users should be hashed when entered into the table to ensure privacy from both hackers and staff accessing the database. | Moderate | High |
| User damaging database structure with invalid inputs | All input points should have the adequate protections and validation to keep the database structure and set up functioning as intended (inputting negative numbers when not applicable, entering a string of a vast length in string columns) | Moderate | Moderate |
| Repo mismanagement and automated update failures | Keeping multiple back ups to negate any erroneous updates sabotaging progress on the construction/update of the end system | Low | High |
| Illness | Plan to make up time around the illness, where time has been lost. | High | High |
| Database net connectivity failure | Focus on constructing the database connection first, iron out problems for project moving forwards. | Moderate | High |
| Database hack/hijack/ransom | Employ industry practice standards of security, not just a good password, but keeping it protected from prying eyes. Currently hack rates have increased globally due to more remote work being conducted leading to less secure packet transfer and less secure system set ups. | High | High |