|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Risk | Response strategy | Objectives | Likelihood | Impact | Risk Level |
| SQL injection attacks | Use prepared statement for any database queries | To prevent SQL injection attacks | High | High | High |
| Unauthorized database access | Have a strong password for connection to the database. Constantly change the password at least once every month and do not keep all your passwords from other accounts the same. | To prevent unauthorized access to the database | High | High | High |
| System or application failure | Use SonarQube for code quality, checks code quality. | To stop the system from failing due to poor code | Medium | High | low |
| Loss of code | Push the versions of the code to a repository on GitHub | You can easily pull your work down if code is lost. | low | high | low |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Risk | Response strategy | Objectives | Likelihood | Impact | Risk Level |
| Human error | Restrict access and give user permissions | To stop users without the correct permissions updating or deleting records | medium | medium | medium |
| Unsecure connection | Encrypted connections | To make sure all connections to the database are secure and encrypted | medium | medium | medium |