Code Review - Finders Keepers

Backend code review

For the backend review, we examined the User Handler, the Item Handler, and tests associated with those two modules. While the backend as a whole is well-designed and well-coded, there are a few outstanding issues to be addressed regarding security and documentation.

First, on the security front, the backend has few safeguards against external attack. SQL calls are frequent across almost every function, and they are all vulnerable to SQL injection attacks due to their naive string-concatenation construction. This could result in the compromisation of user data or database integrity, including but not limited to the stealing of user's inventories and emails, and the damaging or destruction of database contents.

Additionally, and perhaps more concerningly, the backend provides no facilities for user authentication or verification, meaning that anyone could access or modify the contents of the database simply by accessing the api URLs in their browser. This poses a potential security risk.

Besides these security issues, the back-end code in these classes is mostly very well documented and structured. The one exception is in the tests, which have a few relatively minor issues. The tests have limited or no documentation, which poses challenges for determining the exact purpose and intended behavior of tests. There are also several redundant tests that either test the same or very similar functionality repeatedly, or otherwise use large chunks of code that could easily be eliminated by splitting it into smaller, parameterized functions that could serve multiple purposes and make future test construction easier.

What we learned and what was changed

Our revisions mostly focused on improving the backend based on the comments given in the code review. The code review focused on the UserHandler and the ItemHandler, but the issues raised applied to all tests and to all handlers dealing with search functionality, so we went ahead and edited all files accordingly.

We learned that breaking code down into smaller, well-organized, reusable chunks and writing clear, simple documentation is essential for helping future programmers understand the code. Because of this, we added some brief documentation to the back end tests to make them more readable and provide some insight as to what each test class and test method was doing.

We also tackled the issue of sql injection throughout our backend code. We learned that there was a more proper way to create query strings by utilizing format strings. We went through all of our handlers, and wherever there was an execute statement we changed it to utilize a proper format string. Now it will be much more difficult to maliciously alter our database. Additionally, utilizing format strings allowed us to reformat our code in areas where string concatenation caused verbose if else statements, so we were also able to reduce the number of lines of code while making the queries more readable as a result. We were also able to catch some areas where proper error handling needed to be added in order to deal with the case where a user tried to update a non existent item. Overall, refactoring our backend code base based on the security concerns raised in the code review allowed us not only to cover those security holes, but also made our code easier to deal with and more fault tolerant.