Group 25 Group C – Quality Assurance

# Code Review

We had scheduled weekly meetings every Monday where we spent time on code review. During these meetings, we would say what we had been working on since the Tuesday seminar the week before. We then took turns in reviewing one another’s changes. Since we all shared a similar level of expertise, those who attended the meeting took turns in looking over the progress we had made and altering/ helping where each individual could. The focus of this feedback primarily centred around code maintainability in addition to merely fixing obvious bugs. Our meetings cultivated an environment conducive to open dialogue we had the opportunity to discuss our concerns and make comments face to face via in-depth conversations in order to prevent miscommunications and confusion.

For example, during a code the member of the group responsible for functionality adding a new doctor to the database was having trouble adding the doctors in the database to the dropdown. This was resolved during a peer review where it was realised that the attributes in the schema table were different to that of the user stories. Through this, the two members could collaborate to have aligned attributes for the doctor information. This then made future functionalities such as adding a new patient and new booking easier to implement as there was a solid framework we could use as a guide.

# Refactoring

As discussed previously during code reviews, our objective in refactoring prioritised readability and maintainability throughout the development process. We ensured that any inefficient code or unconventional code was altered and rectified as soon as possible. For example, separating GUI and functionality into separate classes; removing unnecessary whitespace to enhance code cleanliness; adhering strictly to Java naming conventions for consistency.

An example of us refactoring our code would be when a member of the group showed their GUI implementation of the days of the week checkboxes for doctor availability, another member of the group refactored that section of the code using a for loop for more efficient creation of the checkboxes, rather each day of the week being declared one by one.

# Issue Tracking

At the beginning of the project, each group member was assigned to a particular functionality that we would individually be responsible for maintaining and testing. This aided query resolution as it facilitated more efficient communication when any questions arouse regarding particular classes. In addition to this we utilised version control through leveraged GitLab commits to track edits, enabling who was making changes and when they were made.

For example, during Sprint one the functionality of the authentication system was created by a certain member of the group. Another member of the group was unaware of the log in details required to get into the system. Since GitLab showed them who had implemented this feature, they were able to reach out to said member and get the information they needed about the authentication system.

To search for any vulnerabilities that might lead to data breaches, additional code analysis was done during the code refactoring processes. This is important to us because only those who need it can access the data stored in databases and Java code. This includes known weaknesses like keeping passwords and usernames in plaintext. It's also important to note that in order to prevent the user from using SQL Injection to access sensitive data, we employed prepared statement queries.