Software Tests, Bugs & Fault Reporting

In regards to software tests, our team will follow the following steps to ensure bugs and fault are accurately picked up and found.

* Developers will begin by testing the core functionality of the software. This will simply involve making sure core functionality such as buttons or input fields work as they’re supposed to. Testing with extreme or borderline values does not necessarily need to be done here. Any issues will be recorded using the gitlab issue tracker.
* Other members of the team will work as reviewers on the developer’s code. More of this is detailed in the version-control document; but essentially code review will work as such: the reviewers will review around 400 LOC (lines of code) per session, highlighting any possible mistakes in the code, and any improvements that can be made, before allowing the developer’s work to be merged into the master branch of the software. Reviewers will leave comments on the piece of work being reviewed, and will inform the group that they have reviewed the work and have left comments. This will ensure our work stays at a high quality.
* Developers will write unit tests for the methods or classes. They will test how these methods & classes work across a range of inputs, namely valid and invalid ones. They will also test with borderline/boundary values to test to see how the methods they have written cope with such inputs. Using the gitlab issue tracker, issues will be recorded for the sake of the developer going back to rectify the issue, or so other members of the team may attempt to rectify it.
* Results from test should be noted in a document with an appropriate title matching the unit or component being tested.

# Tests

Where appropriate, values and results should be noted in a document matching the unit/component being tested.

# Authentication

* Testers will first test to see if all forms of users can log in – whether it be the employee, the manager, or the director. Regardless of whether the user is successfully authenticated or not, results and values used should be noted.
* Testers should test whether or not users can be logged out of the system.
* The authorisation check and the authorisator must work with each type of user in the system.

## Personal Details

* Testers must test if Users can read and amend their personal details.
* Testers must test if Users can amend personal details.
* Testers must test if Hr Employees can create personal details for employees.
* Testers must see if HR Employees can amend personal details for employees.
* Testers must see if HR Employees can read personal details for employees.
* The authorisation check and the authorisator must work with each type of user in the system.

## Review

* Testers must see whether Employees can create new review records.
* Testers must see whether Directors and Managers inherit the capabilities of Reviewers.
* Testers must see whether Reviewers can read past completed review records.
* Testers must see whether Reviewers can Amend review records
* Testers must see whether Reviewers can Read review records
* Testers must see whether Reviewers can Perform reviews
* Testers must see whether HR Employees can Perform Reviews
* Testers must see whether HR Employees can Allocate Reviewers to the HR Database
* The authorisation check and the authorisator must work with each type of user in the system.
* Records should be stored in the HR Database.