Testing

Testing plays a very crucial role in ensuring a bug-free product delivery and an error-free user experience. Complete testing of a web-based system before going live can help address issues before the system is deployed to clientele. Several testing methods have been and continue to be used to enforce and measure the functional qualities of the system.

Selenium is software that enables automated testing of user-created ‘test cases’ to detect syntax errors, semantic errors and logical errors. Selenium is reliable and easy to use. With the chosen framework consisting of PHP and HTML, Selenium was an appropriate and efficient tool to use for testing as its tests were written to return ‘success’ or ‘failure’ (there was no ambiguity).

During testing, some previously unknown bugs were found. The team members conducting the testing passed their results onto the CS team who were able to promptly fix the errors in their code using the testing feedback. One such early bug that was found and fixed was an inability to log in using student accounts. After this error was found and the CS students were alerted to its location it was a very quick fix. The test was ran again after the fix and came back as a success.

**System Integration Testing:**

This black-box testing technique tests the system as an integrated whole, evaluating the system’s compliance against specified requirements (outlined as user stories in the specifications). It tests the required interactions and verifies that all related systems can operate in coordination with other systems in the same environment. This was especially important to test once the php front-end of the application and its SQL back-end were brought together and when code that had been worked on separately by the CS students had been uploaded to the same main branch of the repository.

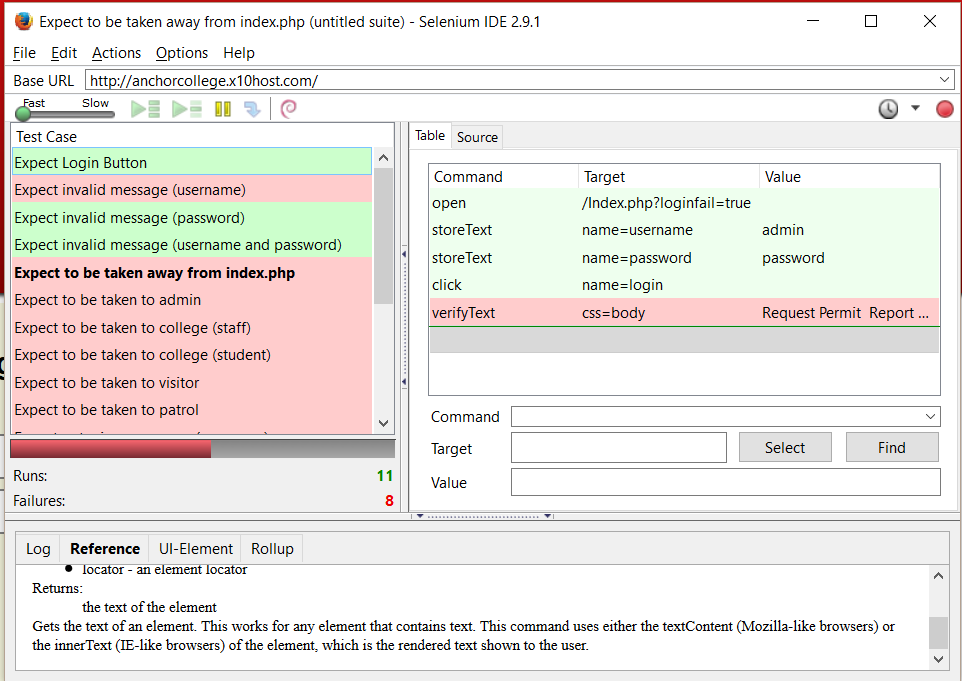
**User acceptance testing:**

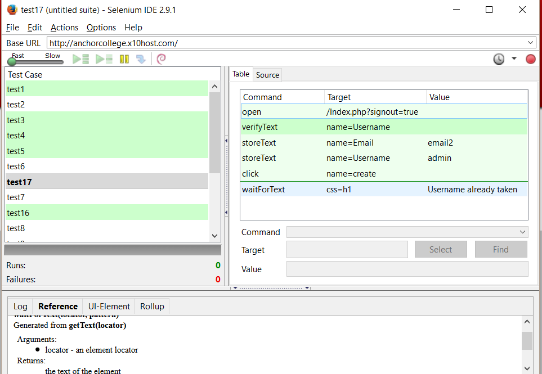
In Scrum, UAT refers to the functional testing of a user story by the development team during a sprint. Actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to user story specifications.

We integrated SIT and UAT by taking on a role of a user to test user stories in the system aided by Selenium, in order to ensure that the system could meet the requirements set by the specification document. It through this testing that we confirmed our completion of each user story outlined in releases 1 and 2.

**Continuous Integration:**

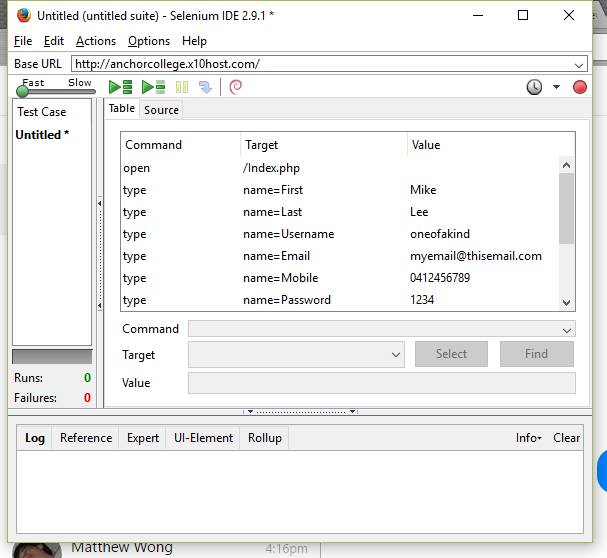
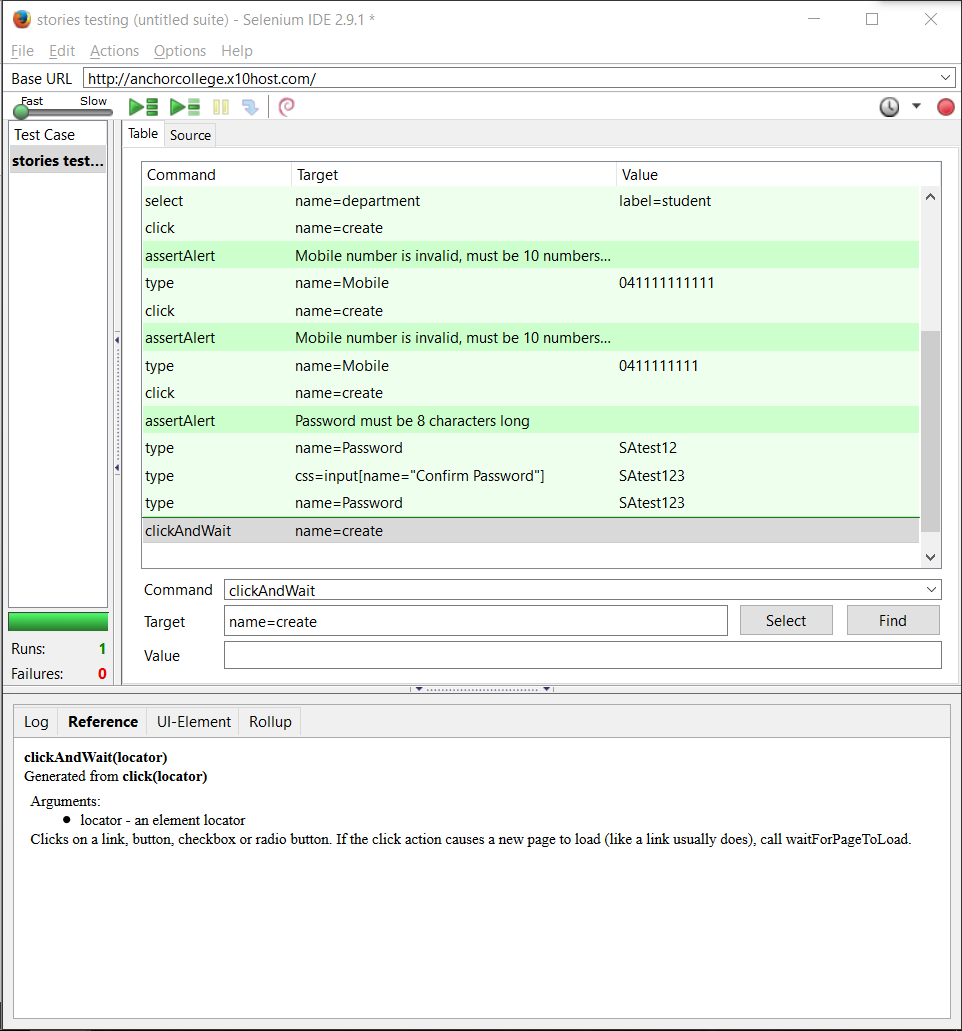
CI is a development practice that requires integrating one’s new or changed code into the master code repository at least once a day. We used the X10hosting service to support this process as all developers had remote access in order to upload code to the shared domain.

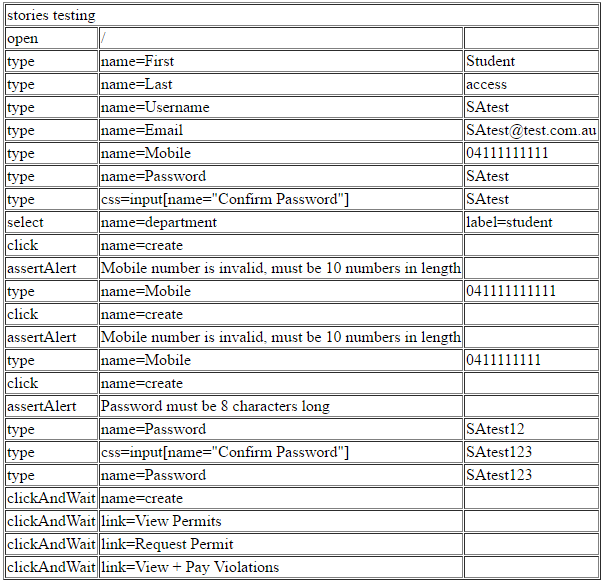
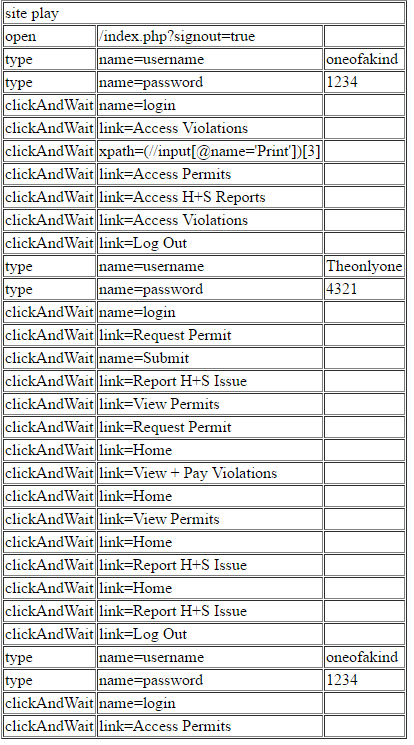
*Figure 1: Example of a suite of test cases. Figure 2: Example of refined test cases*



There are 26 user stories in Release Plans 1 & 2. We tested 24 of them using Selenium with a 100% success rate and the final 2 manually, also with a 100% success rate.

**Automated Testing:**

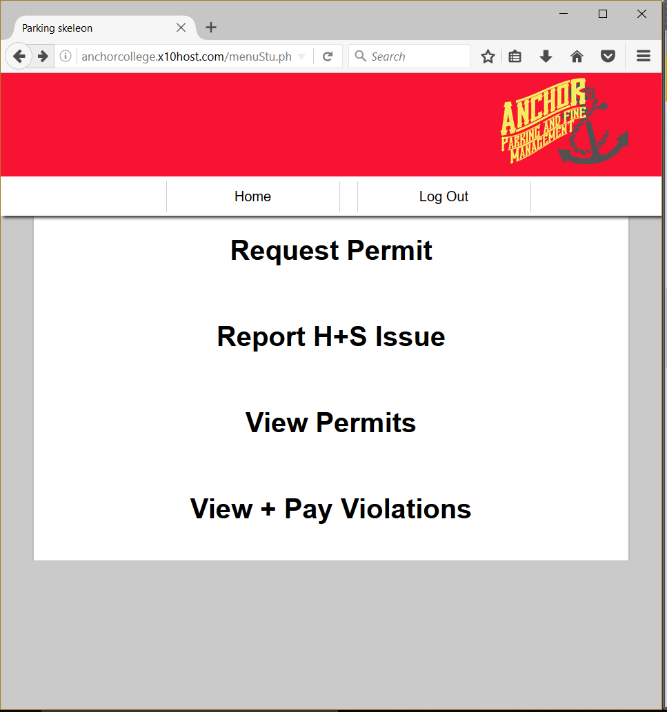
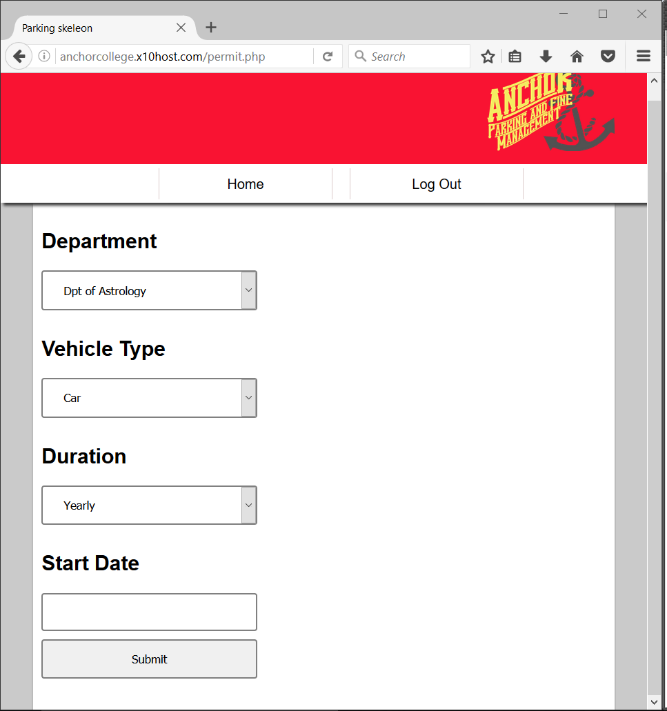
*Figure 3: Initial Test suite Figure 4: Second Test suite*

*Figure 5: Selenium Test output (part 1) Figure 6: Selenium Test output (part 2)*

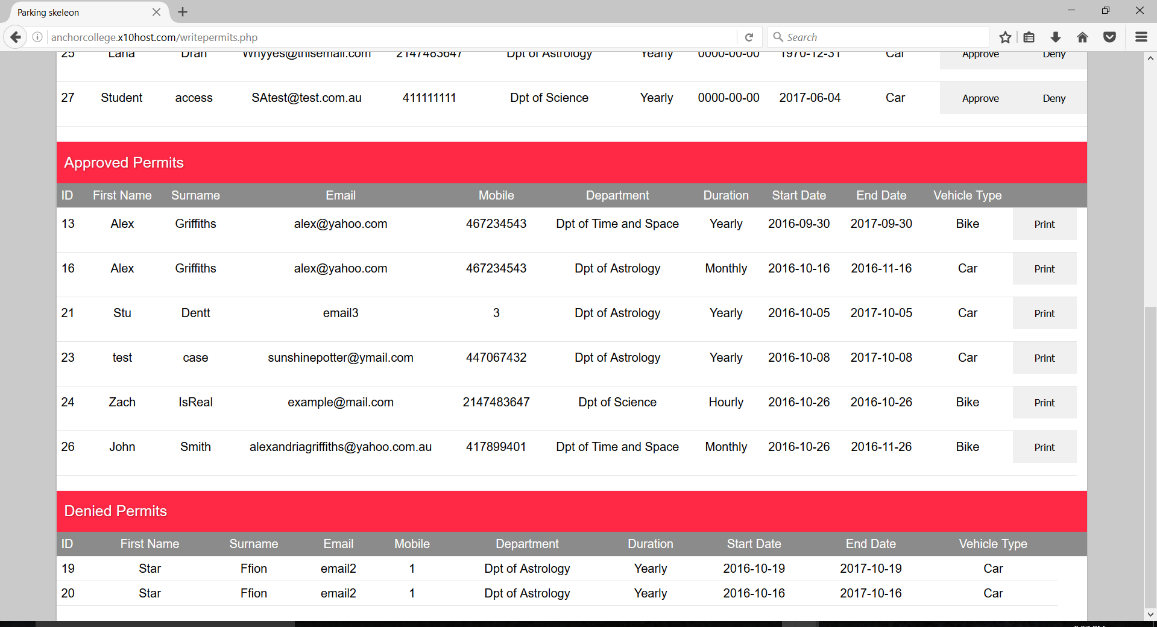
**Manual Testing: *Testing Permit Creation***

***Test 1:***

*Figure 7: Initial screen for Student Figure 8: Request of Permit*

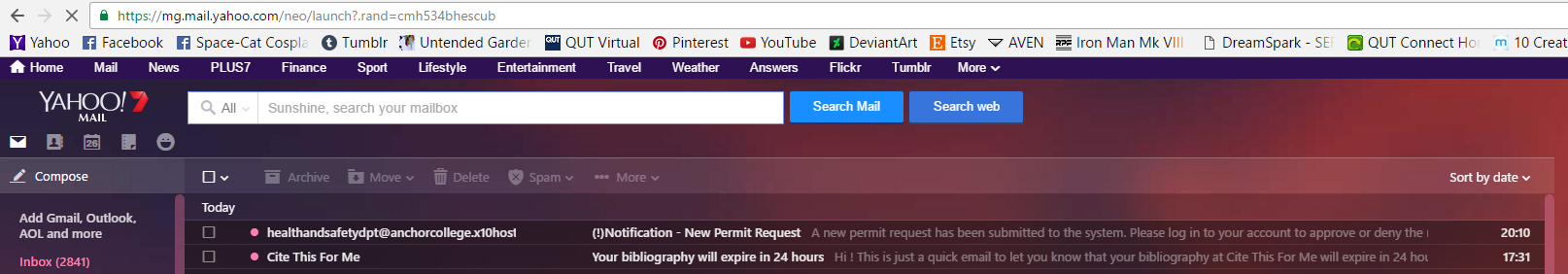
 

*Figure 9: All permits (newly created permit can be seen here, confirming expected outcome)*



***Test 2:***

*Figure 10: Test that email was sent to admin confirming permit request*

****