1. Brief Testing/Bug Fixing Report:

- **Login –** The first screen that the user sees, prompting login. This form has been tested with a variety of incorrect logins: blanks, correct usernames & incorrect passwords and correct passwords with incorrect usernames. All are rejected, while correct logins are accepted. We've also done mild testing against SQL injection, which has not caused an issue with the database.
- **Register –** This page is the one that handles user creation. It has been tested similarly to the login page, with a variety of inputs been rejected if they are invalid with some basic validation. An early issue that was fixing a bug in which it was possible to create duplicate users. This was resolved by adding a check for existing users.
- **TreasureHunt –** This page is the main page that users will see, tracking the waypoints and allowing for switching between different menu options. Issues that have being resolved with this page are markers failing to load onto the map, incorrect order of display and various issues with scaling on mobile devices. Other fixes done were to reduce the performance impact of the QR scanner by only opening it when necessary and ensuring that its scan registered correctly and displayed a new waypoint.
- **adminPage –** This page was not particularly error prone due to its simplicity, the only issues being bugs with getting the user's access level via php sessions and rejecting access if they were not an administrator. This was a logic error that was fixed quickly. A small issue with not writing the new access levels to the database occurred but was fixed after correcting the syntax of the SQL statements.
- **PHP backend –** For a variety of functions php was used. There were frequent problems with communication between the frontend and the backend, the error messages having to be displayed via pop-up for AJAX. Eventually the issue turned out to be with the format of the data as a JSON which was fixed by altering the code to be appropriate for JSON objects.
- **gamekeeper –** Issues with this page mostly involved access levels not being correctly assessed, which was fixed with altering the code managing the php sessions that stored that information.

2. Unit Testing:

Unit testing was performed to test the functionalities of most JavaScript functions used by the main script files, such as script.js, map_script.js, gamekeeper_map.js which provide the back-end JavaScript code for the TreasureHunt.php and gameKeeper.php pages. In this unit testing, there was a heavy focus on markers – testing whether they are correctly implemented, modified and removed and that all arrays are correct, without any errors. Of course, a lot of the other functions were also thoroughly tested. No frame-work was used for the unit testing to reduce memory usage and to save time. The entire test suite uses custom test functions which were implemented manually, such as the 'assertEquals' function, which asserts whether a value is equal to the expected output and other essential testing features, such as resetting each variable used in the tests at the end of each test, so the next test will not be affected (using the 'endTest' function).

Throughout carrying out unit tests, several bugs were found and fixed until each test case was passed successfully. For example (of many), there was an issue where a label was attempted to be changed whilst being undefined during one of the assert tests (which we did not spot before carrying out unit testing). There was also another instance where we found that the 'removeMarker' function (which is a pretty important function) would not correctly remove markers if the marker being removed did not have an Info Window that is active in some circumstances. There were a lot of

other bugs, involving animations not always changing (as there are a lot of conditions that must be checked and fulfilled, i.e. whether a marker is opened, clicked on, whether animations are enabled in the settings and other 'gap' moments where there may be an event that takes place which involves a timeout, or which triggers another even – which must also be taken into consideration). There were also similar issues with the other settings, but they have all been fixed and tested thoroughly and prove to be working without any faults. Some other tests, such as scaling marker sizes depending on the user's zoom level of the map, were carried out using some unit testing but in this case it was also necessary to test this feature by eye to see the markers, including icons and labels, change size correctly when zooming in/out, and for them to appear/disappear when zooming out too far/zooming back in (as well as take into account the extra markers and other enabled settings). Each feature, including each major one, was tested thoroughly.

The test script which runs the tests is located in the file named TestSuite.js. It uses the functions directly from the actual script files, e.g. map_script.js, that are currently being run on the website. However, it loads and saves temporary data on a separate test page: TreasureHuntTest.php.

The Test Suite can be accessed and run by developers by logging onto the website and changing the page URL from 'TreasureHunt.php' to 'TreasureHuntTest.php'. Once on the test page, open Developer Tools (by pressing F12 if you are on Google Chrome) and navigate to the console. In the console, you will see the result for each individual test case, which will either say 'Successful' or 'FAILED!'. There will be a summary after all test cases were completed, detailing how many test cases were passed and how many failed (if there any). If a test case fails, you should refer to the TestSuite.js file and locate the test case in that file (each test has an assigned ID to it, e.g. Test '1' is the first test case, Test '2' is the second test case).

The screenshot below shows the console log of the test cases being run. You can view the code for it in the TestSuite.js file.