Test Plan

The following testing activities have been carried out as part of this project

- Unit Testing
- Integration Testing
- Acceptance Testing
- User Evaluation Testing

Unit Testing

Automated unit testing should be completed on every module/component developed as part of the project, unless given reasons as to why not. Unit tests should be run with a tool that generates a report detailing code coverage numbers (e.g. class coverage, method coverage, line coverage) to be used as an indicator whether the unit tests are helpful or not (e.g. to help avoid having two tests doing the exact same thing). Unit tests should test ideally test one specific thing or unit (e.g. method in a class) and use mock data where appropriate.

Where possible, the unit tests should have an error message detailing the reason as to why it would fail. These messages will serve as part of unit test documentation.

Integration Testing

The purpose of integration tests within this project is to verify and validate the communications between the different components of the project (i.e. web app, NodeJS server, Java server).

Acceptance Testing

Acceptance testing should detail every scenario a user could find themselves, as per the use cases defined in the functional specification and use cases thought of during the course of the development of the project.
An ideal way to document these test cases would be to take inspiration from the 'Gherkin' syntax (https://github.com/cucumber/cucumber/wiki/Gherkin) - Gherkin is used to help describe software's behaviour in a readable way. An example test case format could have (Gherkin syntax in brackets):

- Preconditions (Given)
- Action required (When)
- Expected Results (Then)

User Evaluation

User evaluation should be completed in a way that can quantify the usability of the system.

A proposed way to carry this out would be to follow the ISO 9126-4 Approach.
This approach focuses on three main 'usability metrics' that quantify the usability of the system:

- Effectiveness
- Efficiency
- Satisfaction

Effectiveness

Firstly, a set of tasks that a user must complete must be defined. The tasks must be defined to correlate to the use cases of the systems - how a user would use the system.

Effectiveness can be measured by observing how many tasks a user completes successfully. A way to show the effectiveness could be represent as a percentage of tasks complete and could be calculated like this:
(Number of successfully completed tasks / Number of total tasks) * 100.
These numbers collected from multiple users could then be averaged to show a value for effectiveness of the system.

Efficiency

The same set of tasks could be used as was used to measure effectiveness, and can be completed concurrently.

Efficiency can be measured in terms of the time it takes to complete a task. It is important to take into account unsuccessfully completed tasks - specifically the time it took before a user gave up on the task. Efficiency per user for each task could be measured as follows:

Result of Task/Time Taken(in seconds).
where Result of Task equals 1 if task successfully complete, else 0.

These numbers can then be averaged across the corresponding tasks for all users. This will express a result of efficiency in terms of 'goals per second'. The higher the number, the better the efficiency.

Satisfaction

After users attempt to complete each task the task list, they should be given a questionnaire assessing the ease/difficulty of the task. A 'single ease question' which represents a 10 point scale(eg. 1=very easy, 10=very difficult) which will allow users to define how challenging they found the task.

There should also be a questionnaire given to each test user at the end of test session. The questionnaire can be used to measure the ease of use of the system. A suggestion would be use the 'System Usability Scale' (https://measuringu.com/sus/) to measure the overall satisfaction of the user.

These questionnaires will help pinpoint areas in need of improvement in terms of UI/UX. Users should also be able to describe any comments they have on the system, as they may bring up ideas on how to improve the overall UI/UX.