**Kasper Testing Plan**

**Table of Contents:**

1. Responsibilities of the test team members
2. General testing strategy
3. Training requirements
4. Documentation strategy
5. Schedule

**Responsibilities of the test team members:**

Strong communication of what has been tested, what is planned to be tested, etc. Don’t want to be working on the same test suite.

Creating test cases and manually testing them

Automating the test case created

Updating the test matrix and defect report

**General testing strategy:**

**Critical items to test:**

User registration, Browse/My Listings/My favourites filters, Settings, Add Listing screens. Will mostly be tested by entering information. Most of this will be automated.

**Other items to test:**

Manually check if all screens are behaving as intended; are images showing up properly? Are there any graphical issues? Are there any unintuitive methods of use? Is navigation working properly? Are there any performance issues?

**Specifics:**

* User input fields (specifically in places such as user phone number, prices, and filter pages):

Allowable keystrokes (no characters allowed when entering phone numbers, prices, filter ranges)

* Disallow negative input when specifying prices, and disallow the min/max price in filter to go below 0.
* Ensure province filters works as intended
* When signing out after successful login, ensure user info is saved for next use.
* Ensure user setting data changes are saved, and information for personal listings have been changed as a result
* Ensure user password change is successful

**Check for invalid inputs in the forms:**

* Disallow symbols to be entered in certain fields (bedroom/bathroom filters/specifier).
* Allowable symbols: “-“ for phone number, “@”, “\_”, “-“, and “.” for email address. Other symbols are allowable in email addresses, but many email providers don’t allow the use of these symbols during email creation.
* If no character limit is set for the description, ensure that it goes through stress testing. Example: Try to break the system by entering large amount of characters.
* As a QOL change, should look into making separate description input box (rather than the one we have now)

End-to-end testing:

Using protractor test runner tool with Jasmine framework. Browser testing using Chrome (Firefox has issues with Protractor right now (see <https://github.com/angular/protractor/issues/3823>). (This might no longer be the case with a recent update of Protractor, will have to update and see)

Automatic unit testing:

Using Karma test runner tool with Jasmine framework.

**Training Requirements:**

All testers must learn how to use the testing tools, specifically Karma, Istanbul and Protractor. Karma will be used for our unit test automation, Protractor for our end-to-end test automation, and Istanbul to check test coverage. Testers will write javascript tests for the test configuration file to use. A meeting for the test team on February 9th will go over Protractor specifically. Later meetings will cover Karma. Istanbul is a simple test coverage tool that requires no prior knowledge to use. It’s a simple command line tool that outputs the coverage.

**Documentation:**

**Test matrix:**

Note what is being tested, the data entered, the expected result, and the actual results. Use the Y axis for describing the tests. Use the X axis for features, functionality, or requirements on the specific test.

**Defect document:**

Is the bug re-creatable? To what certainty? List the bug, and the steps to re-create it (input data, prior usage of application, etc). Note any variations that may occur while attempting to recreate.

**Schedule:**

**ID1:**

Most testing will be done with the GUI. Manual tests should be done, along with starting the creation of automated tests with protractor. Store all recorded data in the test matrix. Create a defect report for all defects found.

**ID2:**

Integration of automated end-to-end tests with TravisCI. Creation of unit tests(?), update the defect report and test matrix for new tests.

**ID3 and beyond:**

Integration of automated unit tests with TravisCI (if not done for ID2)

Organize a bug party

Updating of test matrix

Updating of defect report

Compatability testing for iOS, Android

Performance testing