Client Name: DVT
Project Name: KinderFinder

KinderFinder Testing Information

`Team Name:` `MAU Technologies`

Uteshlen Nadesan	28163304
Michael Johnston	12053300
Po-Han Chiu	11063612

`Version: 1`

20 October 2014

Contents

1 Quality Assurance

Our quality management is done and incorporated into our software development and project management process.

During the development process of this project, we have constantly updated our documentation of the requirements for this project. We immediately make the changes on our documentation whenever a change occurs or a better way of implementation of the software is found during development.

We make sure that we have achieved what we have documented and adhere to the requirements of our client, and inform our clients if a change to the requirements when necessay.

2 Testing

Testing is done at every stage of our software development. This ensures that our code are of consistent quality. Our testing processes are repeatable, this allows us to measure performance, quality and output results.

2.1 Unit Testing

We test every function that we code immediately after we have written it. This way, when we code another function that is dependant on a function we have previously written, we can be assured that any bugs that occur are from the function we are currently working with. This allows us to reduce the scope of our search of the bug, and reduces our time looking for the bug in the long term of development.

The technique we use for unit tests are generally manual testing. We manually add mock data as input and then look at the output to see if it is the same as we expected.

We have covered most of the test cases that we can think of, the test coverage can be estimated to be around 90%. We leave the last 10% for the cases that we have never thought of.

2.2 Integration Testing

After we have completed our unit testing and we have a number of units that can work together. We perform our integration testing. We followed a top-down testing approach, as well as the usage model testing approach as our integration testing. Although for the usage model testing approach, the limitation is set by the amount of hardware we have acquired.

Similar to our unit testing, we generally tested the system manually, but with more coverage of hardware testing during integration testing.

Due to the approaches we took for our integration testing, the tests are similar to the expected user environment we have thought of, although at a much smaller scale. The test coverage is estimated to be 85%. We leave the last 15% for the different in reality and what we expected the user environment to be like. It should be noted that we have tested the scalability of our system to some extent during our integration testing.

2.3 Non-functional Testing

We have performed the non-functional testing to meet the quality requirements that we have documented.

As an example of how we did the tests, we have a non-functional requirement that wants us to keep the costs as low as possible. We have searched online to compare prices of the hardware required before the purchase was made. We even changed the type of hardware technology that is used because of pricing. We initially planned to use RFID technology, but later changed to Bluetooth technology because of the price of hardware. We made sure that this change did not affect the core functionality of this project.

Most of the other non-functional testing was performed when we did our integration testing, such as performance, and scalability.

We estimate the test coverage to be 85%. Leaving the other 15% for the possible places that we have missed due to our inexperience.