***TEST SPECIFICATION***

**2.3 Testing tools and environment**

A description of the test environment, including tools, simulators, specialized hardware, test files, and other resources is presented here.

**A few of the tools are listed below:**

* **Testpad:** Testpad is a simple tool that makes test case management easier. The software’s main motto says that it aims to find a bug that matters. Few features of Testpad include manual testing, reports of the test cases and software, dragging and dropping to make testing easier, inviting guest testers by email, building custom templates, and much more.
* **TestCaseLab:** TestCaseLab is easily manageable for the test cases and could swiftly integrate them with bug trackers. The features of TestCaseLab include Custom test cases, Test Runs, Integrations of test cases, Test Plans, tags and priority for test cases, search by name of test cases, description and tags, etc.
* **TestRail:** TestRail is another platform that aims to make test case management easier, it streamlines the software testing processes and along with more visibility into QA. The basic features of TestRail include management for test cases, plans, and runs, more test coverage, real-time insights into the QA progress, etc.

**3.0 Test Cases**

This section enumerates a complete list of test cases for the software. A template for test cases is as follows.

|  |  |
| --- | --- |
| ID |  |
| Test Input |  |
| Expected Output |  |
| Description |  |

User Test Cases:

* Required field - If the screen requires data entry on a specific field, it is good practice to identify the required fields with a red asterisk and to give a friendly warning if the data is left blank.
* **Onscreen Instructions** - Any screen that is not self-explanatory to the casual user should contain onscreen instructions that aid the user.
* Error Logging – If the user enters wrong password, error message should pop up which indicates due to which it caused login error.

4) User home page –

* Onscreen Instructions - Any screen that is not self-explanatory to the casual user should contain onscreen instructions that aid the user.
* Check that all the elements (checkboxes, buttons, text fields, etc) on the webpage are aligned or not.
* Check the placement and position for the error message shown on the webpage according to design.
* Check [404 page](https://en.wikipedia.org/wiki/HTTP_404) is designed or not.

5) View User Details Page:

* Invalid Choices - Do not include instructions for choices not available at the time. For example, if the user requires to fetch more details then button (view details) should direct to the page should have valid request/response.
* Invalid Menu Items - Do not show menu items that are not available for the context the users are currently in.

For view bills button on this page:

* Same Document Opened Multiple Times - If your application opens the same document multiple times, it should append a unique number (unique bill number) to the open document to keep one document from overwriting another. For example, if your application opens a document named doc.pdf if it opens the same document for the same user again, consider having it append the time to the document or sequentially number it (doc2.pdf).

6) Edit User Details Page:

* Save Confirmations- If user wants to edit his/her details, the screen allows changing of data without saving, it should prompt you to save if you move to another record or screen.
* Delete Confirmations - If a person deletes an item, it is a good idea to confirm the delete data