System Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | Description | Pass | Fail | Explanation |
| Usability | Tests how easy it is for a user to learn to operate, prepare inputs for, and interpret outputs of a system or component.  **Example** – Can you find a specific tool/feature of the app? | x |  | Incredibly basic app |
| Accessibility | Determines usability of a product to disabled people (blind, deaf, people with motor control difficulties).  This tests tools like Speech Recognition, screen reader software, magnification software or special keyboards.  **Example** – Does Windows Magnifier work in the app? On screen keyboard? | xx |  | Magnifier works, on screen keyboard works |
| User Interface Testing | Tests how user-friendly an application is.  **Example** – completing a specific task in a program. | x |  | Two text fields and a calculate button. Doesn’t get much easier |
| Stress Test | Evaluates a system or component at or beyond the limits of its specified requirements  **Example** – Fill both fields with integers | x |  | Because it is so basic there isn’t any real stress caused on the system |
| Load Testing | Puts a load on a system or device and measures its response. Unlike Stress Tests, measure the effect on a system with gradual load increases.  **Example** – Measure processing speed when the system is using nearly no resources and compare it to medium and high CPU usage loads. | x |  | Nearly instant, even with a very high load on CPU |
| Boundary Value Testing | Designed to include representatives of boundary values. In other words, test at the limits of the system and then try to go just outside of the values.  **Example --** If ordering product, test quantities of 0, 1 then just under the max allowed, the max allowed and over the max allowed. | x |  | As long as values stay within “int” range, calculate returns a value |
| Ad-hoc Testing | Testing without planning and documentation. In other words, trying to break a system by randomly using system’s functions, or by using previous experience with apps to break a new one.  **Example –** Enable add-ons that conflict with each other |  | x | Multiplied a negative number by 7 and got a correct answer, then multiplied 7 by 1111111111 and it still gave a negative answer |
| Sanity Testing | Tests to make sure new functionality and that bugs are fixed  **Example** – When the addition was changed to multiplication, were there any unintended consequences? | x |  | At first the addition sign was still present in the view. But it was changed to an “x” to show the program is multiplying rather than adding. |
| Comparison Testing | Testing technique that compares new version of software with older versions or with similar products  **Example** – Run in Windows, then run in a VM using another OS. | x |  | Ran on Fedora 20 in a virtual machine |
| Smoke Testing | Tests against major system flaws **Example** – Does the app start successfully | x |  | Runs 100% of the time |
| Internationalization Testing | Testing to ensure the product still works and all messages still work when used in different languages and locales.  **Example** – Change language of keyboard, do all buttons work? | x |  | Changed on screen keyboard to Spanish, numbers still worked |