3.3

1. The newly developed Self Start website portal will allow users to view and interact with pertinent information about their account and treatment plans while still being easy to use and navigate. Through intuitive and logical design choices Self Start will remain robust and maintain user-friendliness. As such, the portal will require minimal time for power and normal users alike to become productive at particular operations.

2. We anticipate the most common tasks for users on the Self Start website would be to make an appointment and view a personal treatment plan. Continuing on the theme of concise and intuitive design we measure these tasks will take less than five clicks to accomplish.

3. Responsiveness is key for excellent user experience in todays world. It is frustrating for users to click on something and have seemingly nothing happen for a few moments while the page loads. As such, we will implement a loading screen or small indications to acknowledge a user’s requests while background data is being loaded, such as personal treatment plans that are being transported from a database or the authorization of login credentials.

4. Taking into the account the special consideration that many of the potential users of the Self Start portal will be patients suffering physical pains, conforming to a specific usability standard is important. For this reason, we will confirming to IBM’s CUA standards and allowing users to navigate the whole website using just the keyboard. Not only does this make it easier for users who struggle using a mouse but navigating through inputs such as those for login credentials is faster when using the Tab key versus using the mouse.

3.4

1. Ideally, the Self Start system should be available 100% of the time. Users should be able to access the site at any time of their choosing. If maintenance needs to be done two days notice should be given so users can be sufficiently aware of the temporary service outage and developers can implement important updates quickly. Maintenance should never take more than 8 hours and should be done during predetermined off-peak system usage hours. Due to the multiple functionalities of the system, degraded operations should be available in certain scenarios. For example, if there is an issue with the database pulling information on treatment plans, the system should still allow users to book appointments with the Marcotte clinic.

2. The Self Start system will be developed in such a way that we believe will never fail. This of course is not a realistic assumption; unforeseen failures (or bugs) will surely arise in the system. But a bug is a one-off problem that will be fixed with a patch. The time between the finding of bugs can be used to predict the timeline of when new bugs will be exposed in the future. Although this can give us an estimate of Mean Time Between Failures, this number is likely to change throughout the product lifecycle and very difficult to estimate prior to the release of the system. Thus, unlike the predictable failures of physical systems, the MTBF of Self Start can not be accurately measured.

3. The Mean Time To Repair should be no longer than 10 hours. This gives developers a few hours to find the bug and then a reasonable amount of time to fix the failure.

4. The system will be outputting patient treatment plans. Since these plans are administered by the physiotherapists and not the system it self, accuracy is not a major concern with our system. The system should only display these plans upon request in exactly the fore the physiotherapist has transcribed.

5. The Self Start system should have no more than 15 bugs/KLOC. This is derived form the industry average of having 85% of bugs found before deployment, leaving about .75 bugs per Function point which is roughly 50 lines. Although 15 bugs initially sounds high it is actually a very good balance between time/cost and effectiveness. Also, these 15 bugs/KLOC are the bugs that remain in the system after all the testing is complete and performance is deemed fit to release—as such any obvious points of failures or chronic issues will not be a part of this figure.

6. Minor bugs: defined as bugs which effect the general presentation of the site. Examples of these type of bugs include graphical bugs like textboxes appearing out of place or miscellaneous abnormalities like “auto-grow” properties of input fields not functioning correctly. 15bugs/KLOC allowed.

Significant bugs: defined as bugs that can hamper a user experience on the system. Examples of significant bugs include issues where users can not access their treatment plan, users can’t book an appointment with the clinic, and available physiotherapists aren’t properly displayed to the user. No Bugs of this nature will be tolerated. They should be found during the testing phase and fixed before release of the system.

Chronic bugs: defined as bugs that will unintentionally delete information, delete users from the database, allow unauthorized users to view medical treatments plans or records of other users (which is a huge violation of doctor patient confidentiality laws), shutdown the operation of the site temporarily or permanently, and/or otherwise make unauthorized changes to any of the system’s databases. Absolutely no bugs of this nature will be tolerated in the release of Self Start.

3.5

1. The longest response times should come from the “Create Account”, “View Exercises”, and “Make Payment” use cases as these use cases have to receive or send the most information to the database.

* Create Account (Average: 3 seconds, Max: 5 Seconds)
* View Exercises (Average: 3 seconds, Max: 6 Seconds)

The loading of small videos or animations depicting exercises will increase the response time.

* Make Payment (Average: 4 seconds, Max 8 Seconds)

The interaction with online banking services is out of the control of the system and may take a significant amount of response time.

Assuming the system will have a total of 300 regular users, the system should be able to do 60 transactions per second, or 20% of the user base at once.

In the event of system degradation, the system should be able to assume degraded modes of operations as specified in section 3.4. At the same time the system should notify system admin of the issue so that developers can perform maintenance as soon as possible.