SE3352a: Software Requirements and Analysis

BRiGADE

For Self Start System

Version 1.0



Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 02/11/17 | 1.0 | Started shared group document | Everyone |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Overall Description 4

2.1 Use-Case Model Survey 4

2.1.1 Introduction 4

2.1.2 Survey Description 4

2.1.3 Use-Case Model Hierarchy 5

2.1.4 Diagrams of the Use-Case Model 5

2.2 Assumptions and Dependencies 5

3. Requirements 5

3.1 Use-Case Specifications 5

3.2 Functionality 5

3.2.1 <Functional Requirement One> 5

3.3 Usability 6

3.3.1 <Usability Requirement One> 6

3.4 Reliability 6

3.5 Performance 6

3.5.1 <Performance Requirement One> 6

3.6 Supportability 6

3.6.1 <Supportability Requirement One> 7

3.7 Design Constraints 7

3.7.1 <Design Constraint One> 7

3.8 Online User Documentation and Help System Requirements 7

3.9 Purchased Components 7

3.10 Interfaces 7

3.10.1 User Interfaces 7

3.10.2 Hardware Interfaces 7

3.10.3 Software Interfaces 7

3.10.4 Communications Interfaces 7

3.11 Licensing Requirements 7

3.12 Legal, Copyright and Other Notices 7

3.13 Applicable Standards 7

# Introduction

## Purpose

The purpose of this Modern SRS is to help in the creation and design of a software system for Marcotte Physiotherapy. This system, titled Self Start, will bring the treatment of the physiotherapist into a patient’s home through an online application, and it will provide clinicians tools to improve the traditional approaches of the patient’s progression assessment. The software must be a web-based application that is accessible 24/7. It must have security features such as authentication to protect the privacy of patients and maintain confidentiality. In addition, the software should be designed in English, and allow new content, such as multimedia, to be uploaded through administrator portals. Finally, the interface must be user-friendly, and a secure database will be kept that contains patient information.

## Scope

This Self Start software will be an online website that is integrated with a database containing Marcotte Physiotherapy patient information. The administrator of the website will be able to modify current users and add new ones. Users can also sign themselves up for Self Start. Self Start will improve the experience of physiotherapy for the patients of Marcotte Physiotherapy. By having an online website that allows for patients to get therapy at home, Marcotte Physiotherapy will have an extra benefit for choosing to go with them for treatment.

Key functionalities in the system will be developed that allow for the database, system, administrators, and patients to interact. Using the features of Self Start, patients will be able to go through their entire physiotherapy treatment in an easy to access manner accompanied by a user-friendly interface.

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
|  |  |

## References

**Title:** Assignment 1: Developing the Software Requirements Specifications SRS

**Reference:** Ouda, A. “Assignment 1: Developing the Software Requirements Specifications SRS.” SE 3352 Software Requirements and Analysis. Western University, 1 Nov. 2017. Web. 2 Nov. 2017.

<https://owl.uwo.ca/access/content/attachment/672f3446-c1d2-46a8-9426-90707ad34952/Assignments/8ebc5171-3538-47ba-adeb-85d4791f515c/SE3352a_assignment1%20_2017_.pdf>

## Overview

Section two, known as Overall Description, conveys the required functionality that the final software must have. This section will provide the foundation for understanding the in-depth descriptions that section three, which is Requirements, will contain. Section two will contain use-case descriptions and the use-case model, while section three will provide use-case specifications along with system functions and requirements.

# Overall Description

## Use-Case Model Survey

### Introduction

As a way to describe the functionality and implementation, use case is used to simulate the functions of the system. It shows the interaction between the system and external users called actors. Use case diagram simulates basic functionality. The use cases in this project is essential to achieve the end goal of an object oriented system to serve marcotte physiotherapy. The system can be used by all users who has access to the internet. Each use case in this model is going to serve both host and client to efficiently get treatment or request booking automatically.

### Survey Description

Self start is a web based application that appends from marcotte physiotherapy website. The purpose of the system is to automate the booking process and easily access treatment online with a corresponding symptom. The system is maintained by the admin and offer currently employed physiotherapist to add a treatment plan to a patient. The system interacts with available physiotherapist’s schedule to automate booking process.

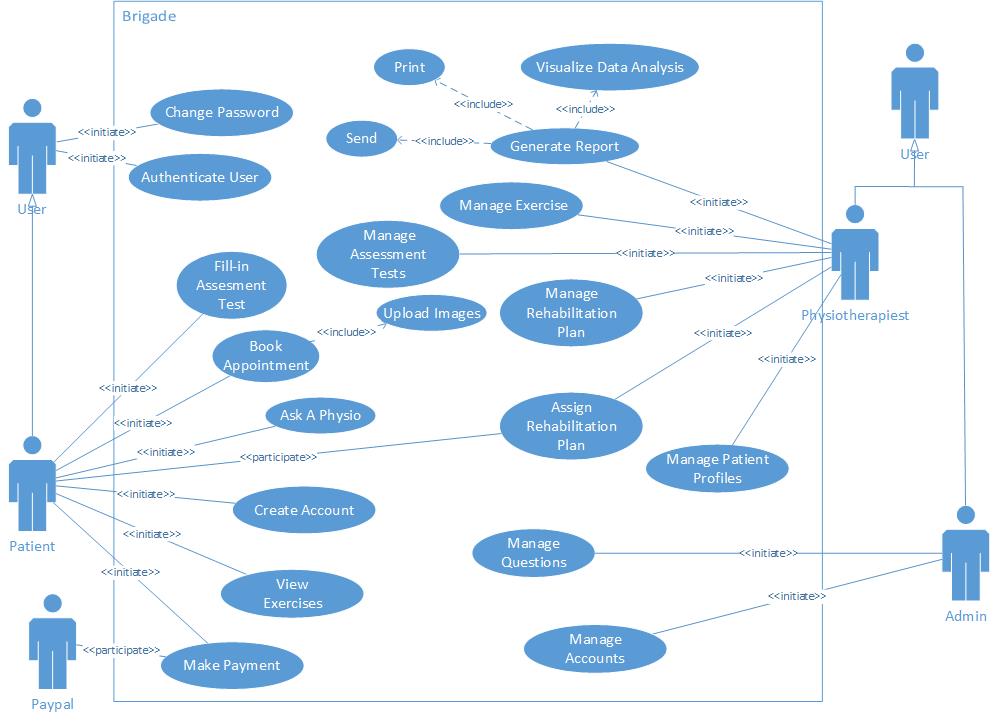
### Use-Case Model Hierarchy

**Self Start**

|  |  |
| --- | --- |
| **Actors** | **Descriptions** |
| User | The user has generic structure for other actors (admin, physiotherapist, patient) |
| Admin | The administrator of the system is responsible for maintaining data of the users |
| Physiotherapist | Currently employed physiotherapist at marcotte physiotherapy |
| Patient | New or current user accessing Self start |
| PayPal | Payment option for the patient |

|  |  |
| --- | --- |
| **Use Case** | **Descriptions** |
| changePassword | Change password of its user |
| AuthenticateUser | While logging in, ask for username and password |
| FillInAssesmentTest | Simple closed question form to the user to be assessed by the physiotherapist |
| BookAppointment | Book an appointment online based on employee's schedule. It Includes uploadImages |
| Ask A Physio | The system allows patients to fill out a form to describe their injury to be examined by physiotherapist |
| createAccount | The system allows users to create account, registering username and password to the database |
| ViewExercises | The system allows authenticated user to view list of exercise assigned by the physiotherapist |
| MakePayment | Make payment for the service via paypal |
| UploadImages | Upload the image of injury while booking an appointment to allow physiotherapist to evaluate beforehand. |
| generateReport | Generate a report of the user, summarizing symptoms and treatment. This includes VisualizeDataAnalysis, print and send use case. |
| VisualizeDataAnalysis | Uses assessment test results to visualize the effects of the treatment |
| print | The system provide an ability to print generated report |
| send | The system enables physiotherapist to send generated report by email |
| createExercise | The system allows physiotherapist to create new exercise to save it in the database. |
| manageRehabilitationPlan | The system allows physiotherapist to add, delete or update rehabilitation plan that has list of exercise and assessment test |
| AssignRehabilitationPlan | The system allows physiotherapist to assign rehabilitation plan to the patient |
| ManageQuestions | The system allows admin to add delete or update the question made by the patent |
| ManageAccounts | The system allows admin to add delete or update the account of all users in the database |

### Diagrams of the Use-Case Model



## Assumptions and Dependencies

[This section describes any key technical feasibility, subsystem or component availability, or other project related assumptions on which the viability of the software described by this Modern SRS may be based.]

# Requirements

## Use-Case Specifications

|  |  |
| --- | --- |
| Use case name | Authenticate User  This use case covers: Objective 10 |
| Participating actor | Initiated by User |
| Entry condition | The user (Patient, Admin, Physiotherapist) is on the SelfStart website |
| Flow of event | 1. The user initiates “Authenticate User” from the website interface by logging in.   2. The SelfStart website responds by presenting a login screen that includes username and password fields.   1. The user fills the username and password fields. Once complete, the user submits the form by clicking on log in.   4. The SelfStart website authenticates the submitted form from the database and displays the acknowledgment |
| Exit condition | * The user receives confirmation of authentication and continues to the home page of SelfStart while logged into their account. * The user receives an error message that the username and password submitted was incorrect and to try again. |
| Quality requirement | * Not applicable |

|  |  |
| --- | --- |
| Use case name | Change Password  This use case covers: Feature 7 |
| Participating actor | Initiated by User |
| Entry condition | The user (Patient, Admin, Physiotherapist) is logged on the SelfStart website |
| Flow of event | 1. The user initiates “Change Password” from the website’s interface.  2. The SelfStart website responds by presenting an interface that includes an old password field (to confirm that it is the proper person making the changes) and a new password field.  3. The user inputs the appropriate information regarding their account in their respective fields.  4. User confirms password update.  6. The SelfStart system updates the changes made by the user in the database.  7. The SelfStart system sends an email to them telling them that a new password has been set. |
| Exit condition | * User is done changing their password. |
| Quality requirement | * Not applicable |

|  |  |
| --- | --- |
| Use case name | Ask A Physio  This use case covers: Feature 2 |
| Participating actor | Initiated by Patient |
| Entry condition | The patient is on the SelfStart website, either logged in or not. |
| Flow of event | 1. The user initiates “Ask A Physio” from the website’s interface.  2. The SelfStart website responds by presenting an interface that includes a form asking the patient to input some general information about themselves and their comment explaining their injury.  3. The user inputs the appropriate information in the form.  4. User submits form.  6. The SelfStart system adds the form into the database.  7. The SelfStart system sends an email to them telling them that they have successfully submitted an “Ask A Physio” form. |
| Exit condition | * Patient submits the “Ask A Physio” form. |
| Quality requirement | * If this is the patient’s first time submitting an “Ask A Physio”, they don’t need to make an account, but to make more in the future, they must make an account (This is an assumption, subject to change if requirement is undesired). |

|  |  |
| --- | --- |
| Use case name | Create Account  This use case covers: Feature 6 |
| Participating actor | Initiated by Patient |
| Entry condition | The patient is on the SelfStart website. |
| Flow of event | 1. The user initiates “Create Account” from the website’s interface.  2. The SelfStart website responds by presenting an interface that includes personal information fields.  3. The patient inputs the appropriate information in the appropriate fields.  4. Patient confirms that they want to create an account.  5. The SelfStart system updates the database.  6. The SelfStart system sends an email to them telling them that they have successfully created an account.  7. System logs in patient automatically after the account has been confirmed. |
| Exit condition | * User creates account. |
| Quality requirement | * Email must be unique (Assumption, subject to change if needed to) |

|  |  |
| --- | --- |
| Use case name | Book Appointment  This use case covers: Feature 2, Feature 3, Feature 4 |
| Participating actor | Initiated by Patient |
| Entry condition | The patient is logged onto their account on the SelfStart website. |
| Flow of event | 1. The user initiates “Book Appointment” from the website’s interface.  2. The SelfStart website responds by presenting the book appointment form to the patient to have them fill it out.  3. The patient inputs the appropriate information in the appropriate fields and answers the questions in the online evaluation form.  4. Patient has the option of an uploading image.  5. The SelfStart system updates the database.  6. The SelfStart system sends an email to them telling them that they have submitted a book appointment request. |
| Exit condition | * Patient submits book appointment. |
| Quality requirement | * None |

|  |  |
| --- | --- |
| Use case name | Make Payment  This use covers: Feature 30 |
| Participating actor | Initiated by Patient |
| Entry condition | The patient is logged onto their account on the SelfStart website. |
| Flow of event | 1. The user initiates “Make Payment” from the website’s interface.  2. The SelfStart website responds by presenting the payment form to the patient to have them fill it out.  3. The patient inputs the appropriate information in the appropriate fields.  4. The SelfStart system contacts the bank API to authenticate the user and make the payment.  5. The SelfStart system updates the database.  6. The SelfStart sends an email to the patient confirming that the payment has been confirmed and gone through. |
| Exit condition | * Patient confirms payment. |
| Quality requirement | * Payment method used has to be Paypal. |

|  |  |
| --- | --- |
| Use case name | View Exercises  This use covers: Feature 18 |
| Participating actor | Initiated by Patient |
| Entry condition | The patient is logged onto their account on the SelfStart website. |
| Flow of event | 1. The user initiates “View Exercises” from the website’s interface.  2. The SelfStart website responds by presenting an interface that shows guides on what exercises to do and how to do them. |
| Exit condition | * Patient is done viewing exercises. |
| Quality requirement | * Exercises are presented in the form of text and/or simple animations. |

|  |  |
| --- | --- |
| Use case name | Fill-in Assessment Test  This use covers: Feature 19, Feature 20 |
| Participating actor | Initiated by Patient |
| Entry condition | The patient is logged onto their account on the SelfStart website. |
| Flow of event | 1. The user initiates “Make Payment” from the website’s interface.  2. The SelfStart website responds by presenting the payment form to the patient to have them fill it out.  3. The patient inputs the appropriate information in the appropriate fields.  4. The SelfStart system contacts the bank API to authenticate the user and make the payment.  5. The SelfStart system updates the database.  6. The SelfStart sends an email to the patient confirming that the payment has been confirmed and gone through. |
| Exit condition | * Patient confirms payment. |
| Quality requirement | * Payment method used has to be Paypal. |

|  |  |
| --- | --- |
| Use case name | Manage Accounts  This use covers: Feature 7, Feature 27, Feature 28, Feature 29 |
| Participating actor | Initiated by Admin |
| Entry condition | The Admin is logged onto SelfStart using the admin account. |
| Flow of event | 1. The admin initiates “Manage Accounts” from the admin’s account interface.  2. The SelfStart website responds by presenting an interface that contains a variety of different options to manage accounts of patients and physiotherapist.  3. The admin can reset a patient’s forgotten password.  4. The admin can create patient and physiotherapist accounts.  5. The SelfStart system saves all changes to the database. |
| Exit condition | * Admin updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Manage Questions  This use covers: Feature 5 |
| Participating actor | Initiated by Admin |
| Entry condition | The Admin is logged onto SelfStart using the admin account. |
| Flow of event | 1. The admin initiates “Manage Questions” from the admin’s account interface.  2. The SelfStart website responds by presenting an interface that contains the current introduction form.  3. The admin can add, change, or delete questions from the introduction form.  4. The admin, after updating the introduction form, clicks on the update button.  5. The SelfStart system saves all changes to the database. |
| Exit condition | * Admin updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Manage Patient Profiles  This use covers: Feature 8 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Manage Patient Profiles” from the physiotherapist’s account interface.  2. The SelfStart website responds by presenting an interface that contains a list of the physiotherapist’s patients.  3. The physiotherapist can add, change, or delete information regarding their patient’s profile.  4. The physiotherapist, after updating the patient’s account, clicks on the update button.  5. The SelfStart system saves all changes to the database. |
| Exit condition | * Physiotherapist updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Assign Rehab Plans  This use covers: Feature 16, Feature 22 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Assign Rehab Plans” from the physiotherapist’s account interface.  2. The SelfStart website responds by presenting an interface that contains the rehab plans and patients (subject to change).  3. The physiotherapist can assign one or more rehab plans to one or more patients.  4. The physiotherapist, after assigning the rehab plan, clicks on the update button.  5. The SelfStart system saves all changes to the database.  6. The SelfStart system sends emails to all the patients that received rehab plans that they have received rehab plans. |
| Exit condition | * Physiotherapist updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Assign Rehabilitation Plans  This use covers: Feature 16, Feature 22 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Assign Rehabilitation Plans” from the physiotherapist’s account interface.  2. The SelfStart website responds by presenting an interface that contains the rehab plans and patients (subject to change).  3. The physiotherapist can assign one or more rehab plans to one or more patients.  4. The physiotherapist, after assigning the rehab plan, clicks on the update button.  5. The SelfStart system saves all changes to the database.  6. The SelfStart system sends emails to all the patients that received rehab plans that they have received rehab plans. |
| Exit condition | * Physiotherapist updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Manage Rehabilitation Plan  This use covers: Feature 9, Feature 11, Feature 12, Feature 13 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Manage Rehabilitation Plans” from the physiotherapist’s account interface.  2. The SelfStart website responds by presenting an interface that includes the current rehab plans made and some other functionalities.  3. The physiotherapist can create rehab plans.  4. The physiotherapist can add exercises and self-assessment activities to the rehabilitation plans.  5. The physiotherapist, after managing the rehabilitation plans, clicks on the update button.  6. The SelfStart system saves all changes to the database. |
| Exit condition | * Physiotherapist updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Manage Assessment Tests  This use covers: Feature 15 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Manage Assessment Tests” from the physiotherapist’s account interface.  2. The SelfStart website responds by presenting an interface that includes the current assessment test made and some other functionalities.  3. The physiotherapist can create, edit, and delete assessment tests.  4. The physiotherapist, after managing the assessment tests, clicks on the update button.  6. The SelfStart system saves all changes to the database. |
| Exit condition | * Physiotherapist updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Manage Exercises  This use covers: Feature 10, Feature 14 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Manage Exercises” from the physiotherapist’s account interface.  2. The SelfStart website responds by presenting an interface that includes the current exercises made and some other functionalities.  3. The physiotherapist can create, edit, and delete exercises.  4. The physiotherapist, after managing the exercises, clicks on the update button.  6. The SelfStart system saves all changes to the database. |
| Exit condition | * Physiotherapist updates changes. |
| Quality requirement | * None. |

|  |  |
| --- | --- |
| Use case name | Generate Reports  This use covers: Feature 17, Feature 21, Feature 23 |
| Participating actor | Initiated by Physiotherapist |
| Entry condition | The Physiotherapist is logged onto SelfStart using the physiotherapist account. |
| Flow of event | 1. The physiotherapist initiates “Generate Reports” from the physiotherapist’s account interface.  2. The SelfStart website responds by generating a report and downloading it to the physiotherapist’s computer.  3. The physiotherapist can print, email the report to the patient, or generate data analysis. |
| Exit condition | * Physiotherapist either prints, sends. |
| Quality requirement | * None. |

## Functionality

1. The system should provide a way for users(patient, physiotherapist, admin) to log in. Upon login, the server system authenticates encrypted username and password. The system should dynamically provide user different level of power depending on their status.
2. The system will offer a single admin account. This admin account have capacity to create new accounts of different power (patient, physiotherapist). Additionally, administrators has the power to manage accounts in the database
3. Every user has the ability to change its own password
4. The system should provide the physiotherapist be able to create a new exercise and manage plans accordingly.
5. The system should provide a form when modifying userdata, providing all the attributes of its corresponding user
6. The system should provide a user to fill out form for a user that enables to include an image or book an appointment and send it to the available physiotherapist to evaluate.
7. The system should provide the physiotherapist to continue or terminate the treatment(s) of the user
8. The system should allow user to fill out the response after the treatment have concluded.
9. The system should support Node.js framework.
10. The system administrator shall have the ability to add users with different level of privilege.

## Usability

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.

## Reliability

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.

## Performance

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.

1. 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.
2. 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.

## Supportability

[This section indicates any requirements that will enhance the supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries, maintenance access, maintenance utilities.]

### <Supportability Requirement One>

[The requirement description.]

## Design Constraints

[This section should indicate any design constraints on the system being built. Design constraints represent design decisions that have been mandated and must be adhered to. Examples include software languages, software process requirements, prescribed use of developmental tools, architectural and design constraints, purchased components, class libraries, etc.]

### <Design Constraint One>

[The requirement description.]

## Online User Documentation and Help System Requirements

[Describes the requirements, if any, for on-line user documentation, help systems, help about notices, etc.]

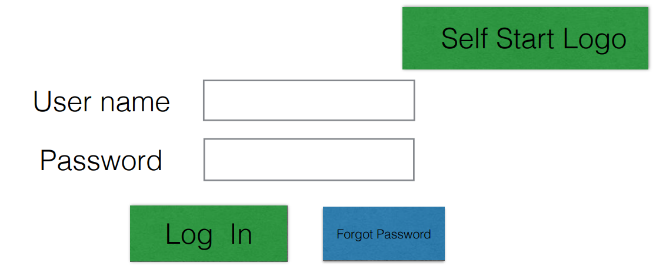
## Purchased Components

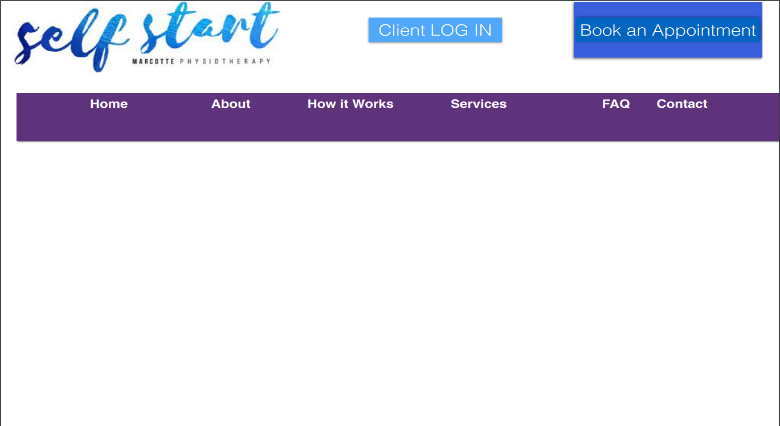
[This section describes any purchased components to be used with the system, any applicable licensing or usage restrictions, and any associated compatibility/interoperability or interface standards.]

## Interfaces

### User Interfaces

All users of the web-based application should see the welcome page when he/she opens the website, see Figure 1. At the top of the welcome page the user will be allowed to log-in or book an appointment, both directing the user to the log-in page, see figure 2.





Welcome

Figure 1 – Welcome page Figure 2 – Login page

In Figure 3, the video of the exercise is shown. When the physiotherapist creates the client’s clinic-defined standard rehabilitation plans (for the common injuries), it will include exercises and self-assessment activities.

The physiotherapist and administrators interact with the system through the easy to follow browser based screens. A physiotherapist should be able to register on the webpage to log-in and manage the client exercises, rehabilitation plans and assessment tests. An administrator should also be able to log-in to the webpage where he/she can administer the system by for instance editing user accounts and settings.

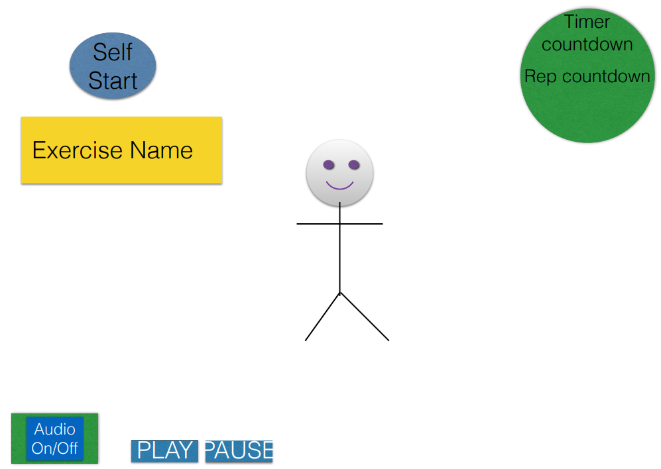


Figure 3 – Exercise video

### Hardware Interfaces

Since the application must run over the internet, all the hardware shall require connecting to the internet will be the hardware interface for the system. For example, a server that is accessed via HTTP requests and a Secure-Socket Layer (SSL) to ensure security.

### Software Interfaces

1. The Self Start system shall communicate with the PayPal system to identify available payment methods, validate the payments and process payments.
2. The Self Start system shall communicate with a video conferencing tool to provide remote communication between the client and physiotherapist
3. The Self Start system shall communicate with Let’s Encrypt to publish the Self Start application over a secure web using TLS technology

The user interface for the software runs on cross web browsers by which the user can have access to the system.

### Communications Interfaces

As briefly discussed in the hardware interface, the Self Start system shall use the HTTP requests for communication over the internet and SSL providing encryption for the communication.

## Licensing Requirements

All the software used on this project are open source and don’t need the purchasing of a license.

## Legal, Copyright and Other Notices

Brigade is the sole owner of its logo, trademark and has certain copywrite notices.

## Applicable Standards

The privacy of the client data and actions will be protected using the standard encryption and key exchange protocol existing in the transport layer security protocol suite (TLS). Clients, however, need to consent to the exercises they are doing on their own, as the clinic will not be reliable for any injuries. The Self Start system will generate its security certificates from the Let’s Encrypt services. Let’s Encrypt is a free, automated, and open certificate authority brought to you by the non-profit Internet Security Research Group (ISRG).