Ademidun Hart & Co.

Modern Software Requirements Specification

For Self Start System>

Version <1.0>

[Note: Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

[Note: The Software Requirements Specification (SRS) captures the complete software requirements for the system, or a portion of the system.  The Modern SRS is a typical SRS outline for a project **using use-case modeling**. This artifact consists of a package containing use cases of the use-case model and applicable Supplementary Specifications and other supporting information. For a template of an SRS **not** using use-case modeling, which captures all requirements in a single document, with applicable sections inserted from the Supplementary Specifications (which would no longer be needed), see[\\program](file:///\\program) \program files\Rational\ RequisitePro\Outlines\ rup\_srs.dot.]

Many different arrangements of an SRS are possible. Refer to [IEEE93] for further elaboration of these explanations, as well as other options for SRS organization.]

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

Modern Software Requirements Specification

# Introduction

[The introduction of the Modern SRS should provide an overview of the entire Modern SRS. It should include the purpose, scope, definitions, acronyms, abbreviations, references and overview of the Modern SRS.]

## Purpose

[Specify the purpose of this Modern SRS. The Modern SRS should fully describe the external behavior of the application or subsystem identified. It also describes nonfunctional requirements, design constraints and other factors necessary to provide a complete and comprehensive description of the requirements for the software.]

## Scope

[A brief description of the software application that the Modern SRS applies to; the feature or other subsystem grouping; what Use Case model(s) it is associated with, and anything else that is affected or influenced by this document.]

## Definitions, Acronyms and Abbreviations

[This subsection should provide the definitions of all terms, acronyms, and abbreviations required to interpret properly the Modern SRS.  This information may be provided by reference to the project Glossary.]

## References

[This subsection should provide a complete list of all documents referenced elsewhere in the Modern SRS. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.]

## Overview

[This subsection should describe what the rest of the Modern SRS contains and explain how the Modern SRS is organized.]

# Overall Description

[This section of the Modern SRS should describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in section 3, and makes them easier to understand. Include such items as product perspective, product functions, user characteristics, constraints, assumptions and dependencies, and requirements subsets.]

## Use-Case Model Survey

[This section contains an overview of the use-case model or the subset of the use-case model that is applicable for this subsystem or feature.  This includes a list of names and brief descriptions of all use cases and actors, along with applicable diagrams and relationships. This section describes the use-case model comprehensively, in terms of how the model is structured into packages and what use cases and actors there are in the model. If you are using packages, the document shows the model structure hierarchically.]

### Introduction

[Introduction to the use-case model.]

### Survey Description

[Survey description of the use-case model.]

### Use-Case Model Hierarchy

[This section presents the use-case packages hierarchically, explains the dependencies among them, and shows the content of each package recursively. If the model has several levels of packages, those at the top-level are presented first. The packages within these are presented next, and so on, all the way down to the packages at the bottom of the hierarchy. For each package include:

* The Name.
* A Brief Description explaining the package's function and role in the system. The description must be understandable to any developer who wants to use the package.
* A list of the use cases owned by the package, including the name and brief description of each use case.
* A list of actors owned by the package, including the name and brief description of each actor.
* A list of relationships owned by the package, including the name and brief description of each relationship.
* A list of the packages directly owned by the package, with each package presented in the same hierarchical manner as above]

### Diagrams of the Use-Case Model

[Diagrams, primarily use-case diagrams, of the entire use-case model are included here.]

## 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

[This section of the Modern SRS should contain all the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements.   When using use-case modeling, the majority of these requirements are captured in the use cases.]

## Use-Case Specifications

|  |  |
| --- | --- |
| Use Case Name | BookAppointment |
| Participating Actors | Initiated by Patient |
| Entry Condition | The Patient has accessed the main page and has logged in to the system. |
| Flow of Events | 1. The Patient requests to book an appointment. 2. The Patient is presented a form that is used to coordinate appointment bookings. 3. The Patient fills out the form with the appropriate information requested; the Patient will have to specify the time for the appointment, but not their contact information as that will already be on file since they have created an account unlike the ***ContactClinic*** use case. 4. The Patient submits the form, which will be handled by the physiotherapist to ensure that they make note of the appointment. The patient is presented with a confirmation message that indicates the appointment has been booked. |
| Exit Condition | The patient has an appointment booked with Marcotte Physiotherapy. |
| Quality Requirement | None. |

|  |  |
| --- | --- |
| Use Case Name | CreateAccount |
| Participating Actors | Initiated by Patient or by the System Administrator |
| Entry Condition | Patient has accessed the main page and has not registered an account with Marcotte Physiotherapy yet. |
| Flow of Events | 1. The patient requests to create an account on the main screen. 2. The patient is served with a form requesting account related information including a password. 3. The patient fills out the form and confirms their information to create their account. |
| Exit Condition | The user has created an account. |
| Quality Requirement | The information submitted should use TLS/SSL to create a secure connection to the server that will protect the Patient’s personal information. Additionally, the password for the user should be hashed using an algorithm, such as SHA-256 or and equivalent, before being stored in the database. Lastly, the system should require that the user input a robust password of a specified minimum length and mix of characters. |

|  |  |
| --- | --- |
| Use Case Name | ViewTreatmentExercises |
| Participating Actors | Initiated by Patient |
| Entry Condition | The Patient has logged into the system. |
| Flow of Events | 1. The patient chooses to view their treatment exercises. If more than one rehabilitation plan is assigned, they will first have to select the plan from which the exercises will come from. 2. The patient is served with a page detailing the exercises prescribed under their current treatment plan. If the patient is not under a treatment plan currently, this page will be blank. 3. The patient chooses a specific exercise if applicable and is served detailed information regarding the exercise. This can be in the form of a text description or a video depiction of the exercise. 4. The patient may navigate away to another portion of the site at any time. |
| Exit Condition | The patient has viewed an exercise and has now navigated away from the page. |
| Quality Requirement | The exercise list and accompanying descriptions should be laid out in an intuitive manner that is easy for new clients to adapt to. |

|  |  |
| --- | --- |
| Use Case Name | SubmitSelfAssessment |
| Participating Actors | Initiated by Patient |
| Entry Condition | The patient has logged into the system and has been requested by a physician to fill out an assessment test. |
| Flow of Events | 1. The patient chooses to start a self-assessment test. 2. The system server the patient with an online form to complete the self-assessment. 3. The patient fills out the required information and submits the form. |
| Exit Condition | The patient has completed the form and has submitted their self-assessment test. |
| Quality Requirement | The completion of the form should be simple and easy for patients to reduce their barriers to receiving treatment. |

|  |  |
| --- | --- |
| Use Case Name | ProcessPayment |
| Participating Actors | Initiated by Patient |
| Entry Condition | The patient has logged into the system and has a balance outstanding on their account. |
| Flow of Events | 1. The patient elects to pay the outstanding balance on their account. 2. The system serves the patient with a payment form requesting information regarding insurance coverage and the payment amount. 3. The patient fills out the required information and is transferred to a secure payments processing platform to input their payment method and complete payment. 4. The patient returns to the system and is notified that the payment has been processed. |
| Exit Condition | The patient has made a payment and has been notified. |
| Quality Requirement | The payment system should submit payment information using TLS/SSL to create a secure connection to the server that will protect the Patient’s payment information. |

|  |  |
| --- | --- |
| Use Case Name | ContactClinic |
| Participating Actors | Initiated by Patient |
| Entry Condition | Patient has accessed the main page and has not been registered yet. |
| Flow of Events | 1. The Patient chooses whether to request an appointment or fill out and submit an online injury form. 2. The Patient is presented the appropriate form depending on their prior choice. 3. The Patient fills out the form with the appropriate information requested. In the case of requesting an appointment, the Patient will have to specify the time for the appointment. The injury form will require information about the Patient’s ailment. Both forms will require the Patient report their contact information so that they can be contacted for further steps. 4. The Patient submits the form, which will be handled by the physiotherapist, and is presented with a confirmation message that indicates the form has been received. |
| Exit Condition | The Patient has submitted their form and will wait for Marcotte Physiotherapy to follow up with next steps. |
| Quality Requirement | The information submitted should use TLS/SSL to create a secure connection to the server that will protect the Patient’s personal information. |

## Functionality

[This section describes the functional requirements of the system for those requirements that are expressed in the natural language style. For many applications, this may constitute the bulk of the Modern SRS Package and thought should be given to the organization of this section. This section is typically organized by feature, but alternative organization methods, for example organization by user, or organization by subsystem may also be appropriate. Functional requirements may include: **feature sets, capabilities and security**.

Where application development tools (requirements tools, modeling tools, etc) are employed to capture the functionality, this section document will refer to the availability of that data and indicate the location and name of the tool which is used to capture the data.]

### <Functional Requirement One>

[The requirement description.]

## Usability

[This section should include all of those requirements that affect usability. Examples:

1. Specify the required training time for a normal users and power users to become productive at particular operations.
2. Specify measurable task times for typical tasks, or
3. Base usability requirements of the new system on other systems that the users know and like.
4. Specify requirements to conform to common usability standards – e.g., IBM’s CUA standards, or the GUI standards published by Microsoft for Windows 95.]

### <Usability Requirement One>

The requirement description.

## Reliability

[Requirements for reliability of the system should be specified here. Suggestions:

1. Availability – specify % of time available ( xx.xx%), hours of use, maintenance access, degraded mode operations etc.
2. Mean Time Between Failures (MTBF) – this is usually specified in hours, but it could also be specified in terms of days, months, or years.
3. Mean Time To Repair (MTTR) – how long is the system allowed to be out of operation after it has failed?
4. Accuracy – specify precision (resolution) and accuracy (by some known standard) that is required in the systems output.
5. Maximum bugs or defect rate – usually expressed in terms of bugs/KLOC (thousands of lines of code), or bugs per function-point.
6. Bugs or defect rate – categorized in terms of minor, significant, and critical bugs: the requirement(s) must define what is meant by a “critical” bug (e.g., complete loss of data, complete inability to use certain parts of the functionality of the system).]

#### *<Reliability Requirement One*>

[The requirement description.]

## Performance

[The performance characteristics of the system should be outlined in this section. Include specific response times. Where applicable, reference related Use Cases by name.

1. Response time for a transaction (average, maximum)
2. Throughput (e.g., transactions per second)
3. Capacity (e.g., the number of customers or transactions the system can accommodate)
4. Degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner)
5. Resource utilization: memory, disk, communications, etc.]

### <Performance Requirement One>

[The requirement description.]

## 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

[This section defines the interfaces that must be supported by the application. It should contain adequate specificity, protocols, ports and logical addresses, etc, so that the software can be developed and verified against the interface requirements.]

### User Interfaces

[Describe the user interfaces that are to be implemented by the software.]

### Hardware Interfaces

[This section defines any hardware interfaces that are to be supported by the software, including logical structure, physical addresses, expected behavior, etc.]

### Software Interfaces

[This section describes software interfaces to other components of the software system. These may be purchased components, components reused from another application, or components being developed for subsystems outside of the scope of this SRS, but with which this software application must interact.]

### Communications Interfaces

[Describe any communications interfaces to other systems or devices such as local area networks, remote serial devices, etc.]

## Licensing Requirements

The agreement between Ademidun Hart & Company (“Ademidun Hart & Co” or the “Company”) and Marcotte Physiotherapy Clinic (“Marcotte” or the “Client”) states that any and all works created by the Company during the course of the project will be owned exclusively by the Company. The system developed at the commencement of the project will be sold to Marcotte where it will be licensed for use in operating a tele-health physiotherapy service. Any usage outside this domain is strictly forbidden without the express written consent from the Company.

## Legal, Copyright and Other Notices

In accordance with the Company’s Copyright Policy, the Client agrees to recognize the name and logo of the Company as copyrighted symbols under the World Intellectual Property Organization Copyright Treaty of 1996. The interpretation and enforceability of this agreement and the rights and liabilities of the parties stated herein shall be governed by the laws of the Province of Ontario. To the maximum extent permitted by applicable law, the provisions of this letter shall supersede any contrary provisions of applicable common law.

## Applicable Standards

The Company will follow industry standards during the course of the project to meet compliance expectations. The standards that will be upheld will fall into the following three domains: regulatory standards, accessibility and usability, and web browser compatibility.

Regulatory – The Company will follow the recommendations of the World Wide Web Consortium (W3C) where necessary and applicable to achieve a final system design that is compliant with best practices as set forth by the W3C. This will help to ensure the system is compatible with future web technologies. In addition, the Company will observe the ISO/IEC 25010:2011 standard during the development process. While compliance with this standard is less clearly defined, the Company deems it useful to keep in mind the focus of this standard on achieving maintainability, reliability, efficiency, and security.

Accessibility and Usability – The Company will develop the platform with photosensitive users in mind. To this end, the software should avoid bright, flashing displays where possible. The United States Access Board has defined this in Section 508 as requiring that “software shall not use flashing or blinking text, objects, or other elements having a flash or blink frequency greater than 2 Hz and lower than 55 Hz.”

Web Browser Compatibility – The Company will ensure the system is able to be accessed on the most common web browsers including but not limited to Chrome, Safari, and Firefox. In addition, customers of the Client should be able to access the system from a mobile device. Performance on platforms other than personal computers may be affected due to the potential lack of support for necessary hardware such as cameras used in the interaction with the system.