Software Requirements Specification

for

CS 441

Version 1.1

CSUSM

Spring, 2014

Modified by RL Fletcher

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
RL Fletcher	1/20/2014	Initial Release	1.0
Corey Paxton	02/18/14	Release for our project	1.1

1. Introduction

1.1 Purpose

This Software Requirements Document (referred to as SRS) defines the scope of the various phases of the CS 441 Software Project for Spring, 2014. (Hereafter referred to as '*Sheriff Hodor*')

1.2 Document Conventions

Requirements designated as **REQ**-[**type**]-**#** are to be tested

1.3 Intended Audience and Reading Suggestions

The SRS is to be used by Testers, Programmers and Analysts. Only Analysts will modify the document. The SRS is organized to give an overview of '*Sheriff Hodor*', followed by specifics and then covers other related requirements. It is recommended that all readers read the entire document. Testers and Programmers should pay special attention to the System Features section. Testers will use the Functional Requirements section as a starting point for their test cases.

1.4 Project Scope

'*Sheriff Hodor*' will be an educational math drill game based on a stopping a train robbery. The project will be divided into three phases. For the first phase '*Sheriff Hodor*' will allow end-users to answer arithmetic questions presented. '*Sheriff Hodor*' will allow for an Admin who will have options unavailable to other users. During the second and third phase, '*Sheriff Hodor*' will be enriched with added functionalities to create a more robust software application

1.5 References

The document, **Project.pdf**, defines the overall goals of the Software project and can be found at http://cc.csusm.edu.

2. Overall Description

2.1 Product Perspective

'*Sheriff Hodor*' is a new, self-contained product. It is not part of any larger system.

2.2 Product Features

'*Sheriff Hodor*' is a stand-alone system for drilling and learning math facts. The details of the product are provided in Section 3.

2.3 User Classes and Characteristics

The Software will be used by an Administrator (Teacher) and an End-user(Student).

2.4 Operating Environment

The Software should run on the Windows XP, Windows 7 and Windows 8 platforms.

2.5 Design and Implementation Constraints

The Software will be delivered as an executable, written in C# using Microsoft Visual Studio.

2.6 User Documentation

Teams will provide a user manual which includes screen shots and explanations of the software use. The Analyst will create the manual.

3. System Features

In this section, give a detailed explanation of all requirements. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Include how the product should respond to anticipated error conditions or invalid inputs in the Use Case documentation

3.1 Math Drill

3.1.1 End-users shall be able to answer presented math facts

High Priority Item
'Sheriff Hodor' shall provide a means for the End-user to enter the answer to a math fact question.

3.1.2 Stimulus/Response Sequences

Once the End-user is identified, The App presents the correct problem set which was predetermined by an Admin

3.1.3 Functional Requirements

REQ-gameplay-1: The app shall present one problem at a time with the positive numbers in green and the negative numbers in red ontop of each other lined aligned right. During subtraction of only positive numbers the higher value must be on top.

REQ-gameplay-2: The End-user shall be able to enter their answer under the presented question.

REQ-gameplay-3: A timer for each question will appear that counts up and displays the time taken on the question. The max time per question is 99.9 seconds. This value will be used in creating the score at the end of the test.

REQ-gameplay-4: If the answer is correct an animation will play in which they shoot the train robber and move onto the next train car. If the answer is wrong they will take 'damage' and it will reduce their superficial score at the end of the test.

REQ-gameplay-5: The results of the test shall be recorded in an XML document attribute that keeps track of the problem set ID, the index of each problem, whether or not they got the problem correct and the date of the test.

REQ-gameplay-6: A superficial score representing how quickly and accurately the End-user completed the test shall be shown after they complete all of the problems in the problem set. A value from 0-1000 will be displayed with the formula specified in section 4 of this document.

3.2 Admin Options

3.2.1 An admin shall be able to designate different session parameters for each End-user

High Priority Item
'*Sheriff Hodor*' shall provide a means for the Admin to setup a problem set for each End-user.

3.2.2 Stimulus/Response Sequences

Once the Admin is identified, setup options for all End-users are presented in which the Admin must select from a check box list of each End-user they wish to modify.

3.2.3 Functional Requirements

REQ-admin-1: The Admin shall be able to designate the arithmetic operation for the problem set for each End-user (addition or subtraction).

REQ-admin-2: The Admin shall be able to designate the number of problems in a set with the default being set to 10 and the max being the max numbers of problems in the problem set.

REQ-admin-3: The Admin shall be able to designate the number range represented in tiers that designate the max number allowed in each problem set (the tiers are 10, 100, 1000, 100000, and max integer).

3.3 User Report

3.3.1 Admin users shall be able to view end-user activity

High Priority Item
'*Sheriff Hodor*' shall provide a means for the Admin to view on-screen results for each user

3.3.2 Stimulus/Response Sequences

Once the user is selected from a table, '*Sheriff Hodor*' provides a means for an Admin user to view a history of end-user activity

3.3.3 Functional Requirements

REQ-userReport-1: An Admin can view a summary of each End-user selected from the list displaying the name and overall % correct for that user

REQ-userReport-2 The table in which the summary is displayed for each End-user selected from the list shall have an option in which the Admin can select an End-user to display their name and each of their tests and the % correct of that test and what problems were missed in that problem set. If an End-user is selected from the table it must update the check box list so only that user is selected

3.4 User Designation

3.4.1 Users shall be able to identify themselves to the system

High Priority Item
'*Sheriff Hodor*' shall have a login screen where an Admin or User can select themselves from a drop down list.

3.4.2 Stimulus/Response Sequences

Once identified and logged in, '*Sheriff Hodor*' will display the correct UI based on access rights

3.4.3 Functional Requirements

REQ-login-1: The End-user can select his/her name from a drop down list of all possible End-users sorted alphabetically with their unique login ID # after in the format of:
[First Name] [Last Name] ([unique ID#])

REQ-login-2: *'Sheriff Hodor'* will display the last “login” date for the user

3.5 Problem Set Generation

3.5.1 Problem Sets are predefined

<u>High Priority Item</u>
A predefined problem set for each tier of difficulty, addition or subtraction, and positive and negative numbers allowed must be created

3.5.2 Stimulus/Response Sequences

When a game is loaded generate random problems from the problem set pool as questions the End-User must answer.

3.5.3 Functional Requirements

REQ-problemSet-1: Problem set pool is maximum 500 questions (some problem set tiers have smaller maximum because the total combination of numbers in the problem set are less than 500)

REQ-problemSet-2: Problem set with subtraction and only positive numbers must have the larger number come first

4. System Design

Req-gamePlay-6 formula:

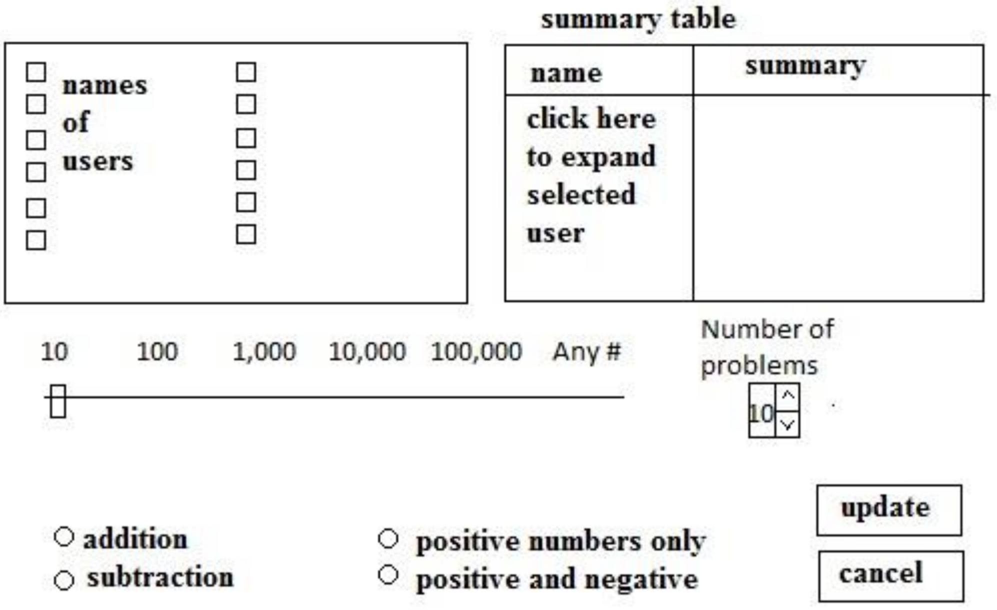
if average time per answer <= 3 seconds -> (Questions correct / total # of questions) * (3000/3)

if average time per answer > 3 seconds -> (Questions correct / total # of questions) * 3000/averageTime

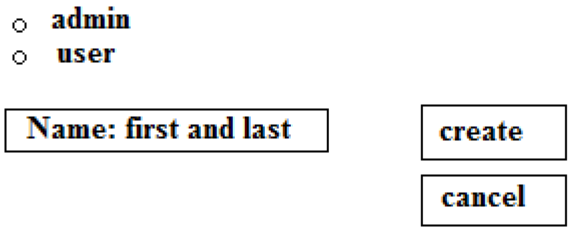
5. Interface Requirements

5.1 User Interfaces

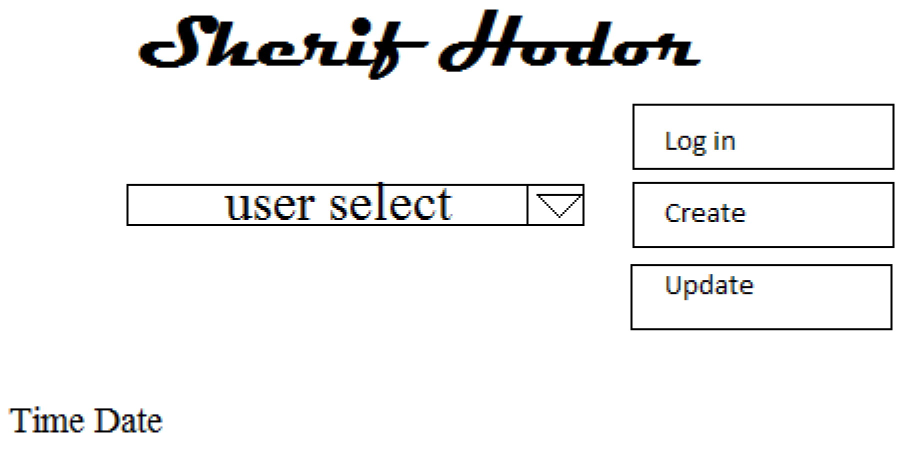
Admin Page



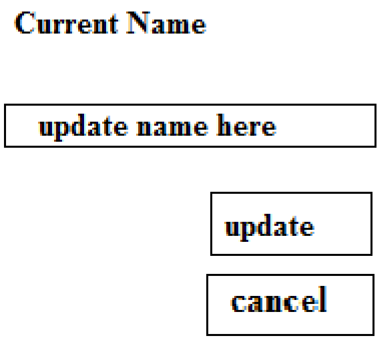
Create User



Login Screen



Update User



The initial user interface will consist of:

REQ-userInterface-1 A drop down list of all system users and their ID #s

REQ-userInterface-2 The correct date and time exact to the minute format tbd

REQ-userInterface-3 Login button that starts the game if an End-User is selected, or logs into the admin interface if an admin is selected

REQ-userInterface-4 Create button to create a new user or admin

REQ-userInterface-5 Update button to move you to the update user screen

REQ-userInterface-6 The title of the game displayed

The admin interface will consist of:

REQ-adminInterface-1 Check box list of all Users

REQ-adminInterface-2 Summary Table that displays all of the Users selected from the check box list. Refer to REQ-userReport-1 and REQ-userReport-2

REQ-adminInterface-3 A number slider with the tiers of 10, 100, 1000, 10000, 100000, 1000000, Any Number (which represents max int)

REQ-adminInterface-4 2 Radio buttons that alternate selectivity between addition and subtraction

REQ-adminInterface-5 2 Radio buttons that alternate selectivity between positive numbers and positive and negative numbers

REQ-adminInterface-6 A number scroll with default value of 10, minimum value of 1, maximum value of the # of problems in the selected problem set that allows you to select the number of questions presented in each test

REQ-adminInterface-7 Update button that updates all of the information for each End-User selected in the check box. Displays a pop up that tells you “Updated” with an OK selection then returns you to the login screen.

REQ-adminInterface-8 Cancel button that returns you to login screen

The update user interface will consist of:

REQ-updateInterface-1 Displays the current name and ID # of the selected user (unchanging value)

REQ-updateInterface-2 Text box for a user to update their name

REQ-updateInterface-3 Update button that displays “[Name] ([ID]) Updated” with an OK selection then returns you to the login screen

REQ-updateInterface-4 Cancel button that returns you to the login screen

5.2 Software Interfaces

REQ-software-1 The App must work as described on the following Operating Systems:

- Windows XP
- Windows 7
- Windows 8

Programmer Requirements:
The app will not interface with any other software components. No data will be shared with any other software.

The App data will be stored locally in the form of XML files. The App will directly read and write to the files.

The App requires no special communications setup or protocols.

6. Other Requirements

Data will be stored in a file using an XML schema to be designed by the Programmers. Programmers will “pair program”, working as a team and strive to cooperate. Programmers will take turns at the keyboard. Programmers may not talk to Testers except to hand off code for testing. All communication between Programmers and Testers will go through the Analyst(s) or bug database.

7. Approval

Before beginning work on The App, all parties must sign off on the SRS

7.1 Phase 1

ROLE	NAME	SIGNATURE	DATE
Programmer	*John Fillecia*		
Programmer	*Juliete Ly*		
Software Analyst	*Corey Paxton*	Corey Paxton	2/18/2014
Software Analyst			
Tester	*Jeff Rackauckas*		
Tester	*Mathias Lang*		

7.2 Phase 2

ROLE	NAME	SIGNATURE	DATE
Programmer			
Programmer			
Software Analyst			
Software Analyst			
Tester			
Tester			

7.3 Phase 3

ROLE	NAME	SIGNATURE	DATE
Programmer			
Programmer			
Software Analyst			
Software Analyst			
Tester			
Tester			

Appendix A: Glossary

SRS *Software Requirements Specifications*
Sherif Hodor *The programming assignment to be completed by CS 441 student teams.*
Problem Set *A group of similar arithmetic problems presented in one drill session*

Appendix B: Use Case Documentation

Add full documentation for each use case here

Appendix C: Issues List

< This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.>