Requirements Specification Document

Version 0.3

PAS Customization Project

**Team 7**

Stephanie Greene (Team Leader)

Matthew Boydston

Eric DeShazer

Javier Ochoa

David Rivera

SE 4485.001

Software Engineering Project

Fall 2013

# Abstract

The PAS (Patient Accounting System) Customization Project was commissioned by MedAssets to provide a greater level of control to customers using the Charge Capture Audit application. The purpose of the PAS Customization Project is to provide an easy to use interface through which MedAssets customers can customize the structure of the corrected charge export data file from the Charge Capture Audit application, as well as generate and download the export file. This software requirements specification enlists all requirements necessary for project development.

# Version Control Table

|  |  |  |  |
| --- | --- | --- | --- |
| Version Number | Purpose/Change | Author | Date |
| 0.0 | Initial draft | Matthew Boydston and David Rivera | 9/22/2013 |
| 0.1 | Refinement of initial draft | Matthew Boydston and David Rivera | 9/24/2013 |
| 0.2 | Proofreading of initial draft | Stephanie Greene | 09/24/2013 |
| 0.3 | Finishing Table of contents, list of figures and list of tables | David Rivera | 09/26/2013 |

# Table of Contents

Section 1.0 – List of Figures 5

Section 2.0 – List of Tables 5

Section 3.0 – Introduction 6

Section 4.0 – Use Case Models 7

Section 5.0 – Rationale for Use Case Models 10

Section 6.0 – Non-Functional Requirements 11

Section 7.0 – Configuration Management 13

References 14

# Section 1 – List of Figures

Use Case Diagram 7

# Section 2- List of Tables

UC-01 8

UC-02 8

UC-03 9

UC-04 9

FR1 10

FR2 11

FR3 11

FR4 11

NFR 11

# Section 3 - Introduction

This section presents the requirement specification document for The PAS (Patient Accounting System) Customization Project. This section also provides the purpose and scope of the system. An overview of the remaining sections of the document is also included in this section.

## 3.1 - Purpose

This document describes the functions and requirements that are required for the development of the PAS Customization Project. The project was commissioned by MedAssets to provide an increased level of control to their customers using their Charge Capture Audit application. Charge Capture Audit provides automated auditing of inpatient and outpatient billing to improve billing accuracy and increase consumer revenues. An important feature of the Charge Capture Audit Application is the ability to export corrected charge data from the MedAssets system into a CSV file for use by the customer’s internal record keeping system. The structure of this CSV file must be customized to each customer’s requirements. This customization is currently performed by MedAssets staff. The PAS Customization Project will provide an interface through which MedAssets customers can customize the structure of the corrected charge export file as well as generate and download the resulting CSV file.

## 3.2 - Scope

The final deliverable will be a stand-alone Windows application that will include the interface and a backend database of patient records from which the data will be exported. The programming language used to develop the interface and backend database connections will be Java. The database will be created using the MySQL relational database management system.

The delivered application will provide a model to implement a new standard feature of the Charge Capture Audit application. The delivered interface must allow each customer to specify the name of the export data file, the number of fields to export, the order of the fields, how a debit or credit is represented, whether column headers are included, and finally to download the customized export data file. Each customer must have the ability to save multiple instances of their preferences. Once saved, the customer may use their preferences to generate and export additional CSV files.

## 3.3 - Definitions, Acronyms, and Abbreviations

***CSV*** – comma-separated values

***PAS*** – Patient Accounting System

***Scrum*** - Scrum project management is a software agile development process

***ScrumMaster*** - The ScrumMaster is responsible for making sure a Scrum Team lives by the values and practices of Scrum.

## 3.4 - Overview

This document provides a high-level description of the PAS Customization Project. The involved users are identified and their roles are explained. The functional requirements are then listed, followed by their respective graphic and textual use case models. Rational for these use cases is also included. The non-functional requirements are then listed, followed by detailed descriptions for each requirement in the subsequent section. The document concludes by providing evidence that the requirements have been placed under configuration management.

# Section 4 - Use Case Model

## 4.1 Use Case Diagram



## 4.2 Textual Use Cases

|  |
| --- |
| **Use case: Configure CSV File Settings** |
| ID: UC-01 |
| Brief description:  User will configure the export settings for a CSV Export configuration file. |
| Primary actor(s): User |
| Secondary actor(s): None |
| Entry conditions: None |
| Main flow:   1. User will start the application and Select to create a new Configuration file 2. The user will be given the option to: 3. Select fields that are included 4. Change the order of fields selected 5. Select if column headers are included 6. Select the field separator character 7. Select a representation for debit or credit 8. Select a File name for export   Alternate flow:  1A. First time user opens application they will be asked to create a Configuration file  1B. The user will select a previously saved configuration file for modification |
| Post conditions:  A valid CSV configuration file has been created. |
| Special Requirements: |

|  |
| --- |
| **Use case: Save CSV File Settings** |
| ID: UC-02 |
| Brief description:  User will save a created CSV Export configuration file. |
| Primary actor(s): User |
| Secondary actor(s): None |
| Entry conditions: A valid CSV Export configuration file has been created |
| Main flow:   1. The user will select to save a created CSV Configuration file 2. The user will select a desired name for the CSV Configuration file 3. The system will save the CSV Configuration file 4. The system will ask if the user wishes to use the saved CSV configuration file as his default configuration settings   Alternate flow: |
| Post conditions:  A valid CSV Export configuration file has been saved. |
| Special Requirements: |

|  |
| --- |
| **Use case: View Past/Saved CSV Export Configurations** |
| ID: UC-03 |
| Brief description:  User will browse past or saved CSV configuration files. |
| Primary actor(s): User |
| Secondary actor(s): None |
| Entry conditions: A valid CSV configuration file has been created or user has a CSV file export history |
| Main flow:   1. The user will select to view Past/ Saved CSV Export configuration files 2. If the user has saved CSV Export Configuration file(s) 3. A list of the saved configuration files will be displayed 4. If the user has a CSV file export history 5. The user’s CSV file export history will be displayed 6. If the user selects a configuration file from either list 7. The selected configuration file will be set as the current configuration settings   Alternate flow: |
| Post conditions:  A selected CSV Export configuration file will be set as the current configuration settings. |

|  |
| --- |
| **Use case: Export CSV File** |
| ID: UC-04 |
| Brief description:  User will export CSV File. |
| Primary actor(s): user |
| Secondary actor(s): None |
| Entry conditions: None |
| Main flow:   1. User will start the application 2. Users default configurations are set as current configurations 3. User is displayed a list of PAS files available for export 4. User selects a PAS file for export 5. User clicks Export button 6. User is prompted with a location to save the export file 7. System uses current CSV Export Configuration to format selected PAS file 8. System saves formatted CSV file in the user s desired location   Alternate flow:  2A. Includes(View Past/Saved CSV Export Configurations)  2B. Includes(Configure CSV file) |
| Post conditions:  A formatted PAS file is saved on the user’s desired location based on current export configurations. |
| Special Requirements: |

# Section 5 - Rational for Use Case Model

## 5.1 Export CSV File

The purpose of this project is to aid in the exporting of CSV files according to the user’s specifications. The Export CSV File is the central use case in this application and the most important to the user. After initial configuration of the application by the user the Export CSV File will be the primary feature used in the application.

## 5.2 Configure CSV File Settings

In order to accomplish the Export CSV File use case of this application the user must be able set their exporting specifications. The Configure CSV File Settings use case will allow the user to create a configuration that will used by the Export CSV File use case to create a CSV file according to the users required specifications. In order to stream line the CSV file export, a feature that allows the user to save their specifications for future use is needed.

## 5.3 **Save CSV File Settings**

To increase usability the CSV file export, a use case that allows the user to save their current specifications for future use is needed. The Save CSV Settings use case will allow the user to create a new configuration file with the Configure CSV File Settings and save it for future use.

## 5.4 View Past/Saved CSV Export Configurations

The history of previously exported CSV files and their configurations will be important information for the user along with the ability to view previously saved configurations which will not always be inclusive with each other. The View Past/Saved CSV Export Configurations use case will allow the user to browse through previously saved configuration files and to view past export configuration files.

# Section 6 - Functional Requirements

## 6.1 CSV Configuration File

|  |  |
| --- | --- |
| **FR1.01** | The system shall allow the user to select fields in the CSV file. |
| **FR1.02** | The system shall allow the user to select the order in which the fields will be inserted in the CSV file. |
| **FR1.03** | The system shall allow the user to select the delimiter character that will separate values in the CSV file. |
| **FR1.04** | The system shall allow the user to determine the representation of debit or credit in the CSV file. |
| **FR1.05** | The system shall allow the user to include column headers in the CSV file. |

## 6.2 Saving Configurations

|  |  |
| --- | --- |
| **FR2.01** | The system shall allow the user to save configuration settings for later use. |
| **FR2.02** | The system shall allow the user to modify saved configuration settings. |
| **FR2.03** | The system shall allow the user to select default configuration settings for future use. |

## 6.3 View Existing Configurations

|  |  |
| --- | --- |
| **FR3.01** | The system shall allow the user to view previously saved configuration settings. |
| **FR3.02** | The system shall maintain a history of the user’s previously exported CVS files and their particular settings. |
| **FR3.03** | The system shall allow the user to select any of the existing configurations. |
| **FR3.04** | The system shall allow the user to set a selected configuration as the current configuration |

## 6.4 Export CSV File

|  |  |
| --- | --- |
| **FR4.01** | The system shall allow the user to export a CSV file. |
| **FR4.02** | The system shall allow the user to select the location where the CSV file will be saved. |
| **FR4.03** | The system shall allow the user to select the name of the CSV file. |

# Section 7 – Non-Functional Requirements

## 7.1 Non-Functional Requirements

|  |  |
| --- | --- |
| **NFR1** | The interface must respond to users actions within .1 seconds. |
| **NFR2** | The system must begin generation of the CSV file within 1 second of the users action. |
| **NFR3** | The system must finish generation of the CSV within 10 seconds from the start of the action. |
| **NFR4** | The generated CSV file must be available for download within 5 seconds after generation is complete |
| **NFR5** | The system must have less than 1 hour downtime per 3 months. |
| **NFR6** | The system must validate the users log in information within 1 second of the user attempting to log in. |
| **NFR7** | The system must load all of the users saved CSV preferences upon the launch of the application. |
| **NFR8** | The system must save a user’s preferences within 1 second. |

# Section 8 – Non-Functional Requirements Detail

## 8.1 NFR1 Detail

The interface must respond to user’s actions within .1 seconds.

### 8.1.1 Description

The user will be required to make use of various buttons to navigate through and use the application. Any time one of these buttons is pressed, the intended outcome must ensue within .1 seconds. For example, if a user clicks a radio button in order to select it, the radio button should appear to be selected in no longer than .1 seconds, starting from the time of the user’s click. Another example is that when a user clicks a button to display a drop-down menu, the menu should appear correctly within .1 seconds of the user’s click. This time limit applies to all client-side features.

## 8.2 NFR2 Detail

The system must begin generation of the CSV file within 1 second of the user clicking the “generate report” button.

### 8.2.1 Description

Starting from the time the user clicks the “generate report” button, the system must begin generation of the CSV file in less than 1 second.

## 8.3 NFR3 Detail

The system must finish generation of the CSV within 10 seconds.

### 8.3.1 Description

Starting from the time the system begins the generation of the CSV file, the system must be finished generating the file in no longer than 10 seconds.

## 8.4 NFR4 Detail

The generated CSV file must be available for download within 5 seconds after generation is complete

### 8.4.1 Description

Once the system completes generation of the CSV file, the system makes the report available for the user’s download in no longer than 5 seconds.

## 8.5 NFR5 Detail

The download of the CSV file must begin within 1 second of the user’s clicking the “Download” button.

### 8.5.1 Description

Starting from the time the user clicks the “Download” button, the system should initiate the user’s download in no longer than 1 second.

## 8.6 NFR6 Detail

The CSV file will be available for export to the user’s desired location within .5 seconds following the completion of the download.

### 8.6.1 Description

Once the user has finished downloading the file to their local machine, they will have the option to export it to their desired location. Once the user has chosen the desired location and name of the file, they will click the “Export” button. Starting from the time they click this button, the system must save the file to the given location with the name provided within .5 seconds.

## 8.7 NFR7 Detail

The system must load all of the users saved CSV preferences upon the launch of the application.

### 8.7.1 Description

Upon launch of the application, the system must retrieve and make available all of the user’s previous CSV configurations.

## 8.8 NFR8 Detail

The system must save a user’s preferences within 1 second.

### 8.8.1 Description

In the event the user wishes to save their current configuration, the system must save the current settings within 1 second of the user clicking the “Save Configuration” button.

## 8.9 NFR8 Detail

The system must validate the users log in information within 1 second of the user attempting to log in.

### 8.9.1 Description

Starting from the time the user clicks the “Login” button, the system must validate the user’s credentials within 1 second.

## 8.10. NFR8 Detail

The system must have less than 1 hour downtime per 3 months.

### 8.10.1 Description

The time in which the system is not totally available for use by the user must be less than 1 hour per 3 months.

# Section 9 – Configuration Management

Configuration management responsibilities for this project are shared by the entire project team. The ScrumMaster oversees the quality management process for the project as a whole, and the individual deliverables for each phase will be overseen and managed by the appropriate project team members with the assistance of the project manager and ScrumMaster. If the projects progress at any milestone strays from the scope of the project, the requirements will be reevaluated and the completed modules will be reconsidered.

# References