Software Requirements Specification

For

[Team 4 - ACH Case Tracking]

[Feb 18, 2016]

[Version 1]

Prepared by:

Dan Stucky, Drew Shoemaker, Hemin Qaradagi, Mohamed Said, Jeff Sterner

**Table of Contents**

[Revision History 3](#_Toc443577379)

[1 Introduction 4](#_Toc443577380)

[1.1 Overview 4](#_Toc443577381)

[1.2 Goals and Objectives 4](#_Toc443577382)

[1.3 Scope 4](#_Toc443577383)

[1.4 Definitions 4](#_Toc443577384)

[2 General Design Constraints 5](#_Toc443577385)

[2.1 Product Environment 5](#_Toc443577386)

[2.2 User Characteristics 5](#_Toc443577387)

[2.3 Mandated Constraints 6](#_Toc443577388)

[3 Nonfunctional Requirements 6](#_Toc443577389)

[3.1 Operational Requirements 6](#_Toc443577390)

[3.2 Performance Requirements 6](#_Toc443577391)

[3.3 Security Requirements 6](#_Toc443577392)

[3.4 Other Quality Attributes 6](#_Toc443577393)

[3.5 Documentation and Training 6](#_Toc443577394)

[3.6 External Interface 6](#_Toc443577395)

[3.6.1 User Interface 6](#_Toc443577396)

[3.6.2 Software Interface 7](#_Toc443577397)

[*4* Functional Requirements 7](#_Toc443577398)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Name** | **Description** |
| 1 | 2/18/2016 | Dan Stucky | Initial Requirements Document created |
|  |  |  |  |

# Introduction

## Overview

The purpose of ACH Case Tracking Phase I will be to inhance visibility into various types of Government Reclamation requests received. This will allow the ACH Department to limit their risk of missing a request, resulting in a loss to the bank. This will also allow greater efficiency for the DNE process to be able to import the DNEs instead of logging each one individually in excel. The team would use this functionality to keep track of open requests, make appropriate comments in relation to action taken on requests, and be able to research past requests easily. This phase should also include reporting abilities, specifically month end type reporting, to allow us (at a minimum) to see how many cases were worked in a given month. These counts should be able to be broken down by sub-type.

## Goals and Objectives

There are three main goals associated with the production of the ACH Case Tracking project:

1. Standardize and automate the process of DNE case creation.
2. Allow users to track and quickly find DNE cases stored in the system.
3. Add accountability by providing a transparent auditing layer for case creation and editing.

## Scope

Four different types of positions that will have varying levels of access to modify details of an ACH case through the use of a UI.

The ACH Case Tracking application will feature a friendly UI with a main menu that allows users to navigate to a manual case creation screen, import cases from a NACHA file, or search for specific cases according to various parameters. Cases may be edited after their creation and their state can be updated as progress on the case advances.

In addition, cases may be reopened and modified anytime after their closure. In order to not overwhelm the user with excess form data, the inputs for manually creating a case will update as the user enters in information so that only relevant items are shown on screen at once. Individual case views will also feature a SLA so users can see when a deadline is approaching. All case creation and editing instances will be audited in an organized and structured view according to date.

The client would like to have an authentication and reporting system built into the app as well, but these features will likely be left out due to time constraints associated with their implementation and scale of the project.

## Definitions

**Acronyms:**

**ACH** - Automated Clearing House

**DNE** - Death Notification Entry

**SLA -** Service Level Agreement

**EBS -** Electronic Banking Service

**IE** - Internet Explorer

**Definitions:**

**Product** – The software system specified in this document. A cloud-based ACH case tracking system for various types of government reclamation requests.

**Client** – the person or organization for which this product is being built. In this case, the client is Commerce Bank.

**User** – the person or persons who will actually interact with the product.

**Use case** – describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes.

**Scenario** – one path through a user case

**Actor** – user or other software system that receives value from a user case.

**Developer** – the person or organization developing the system, also sometimes called the supplier.

**Controls** – the individual elements of a user interface. For example, the following are user interface controls: button, check box, dropdown list.

# General Design Constraints

## Product Environment

The system will have to interact with a few components from Commerce Bank's own infrastructure including an object-relational database management system and also an authentication/authorization system. The system should make both of these external components easy to wire in and replace or modify at a later date if desired. The ACH Case Tracking system will replace the current system of recording cases inside Excel. However, the new system will still have the ability to export data into an excel spreadsheet.

## User Characteristics

There are four types of users for the end system: EBS, call center, management, and admins.

EBS represent the primary users and they may create, edit, or close a case.

Call Center and management may only create cases.

Admins have access and editing privileges for everything including setting SLAs.

However, for the scope of this project, there will likely be only one user since the variable data access and editing permissions are low priority and will only be implemented if time allows. Therefore, all users will simply be referred to as 'user' for this document.

## Mandated Constraints

The application is guided by a few constraints:

1. App built using Java MVC or .NET framework.
2. Database must be relational.
3. No hosted open source libraries may be used.

# Nonfunctional Requirements

## Operational Requirements

This section describes the general characteristics of the physical environment for the product.

***Example:***

The output of the system must be visual because it will run on machines without a sound card.

## Performance Requirements

Must function in IE9 - IE11. All database inputs should be scrubbed to prevent SQL or javascript injection. The system should be easy to install on any windows computer and available through the cloud at all times. Downtime is allowed, but should be kept minimal. No data loss is allowed.

## Security Requirements

As discussed above, there will be and authentication and authorization system with varying degrees of data and editing permissions depending on the user in the final system, but it is beyond the scope of this project's goal. An auditing trail will be kept and stored for every change made to a case. The field, user, and changes made will be included in the audit.

## Other Quality Attributes

NA

## Documentation and Training

No training is required for users to work with the product.

## External Interface

### User Interface

The user interface should require no training to become familiar with. The user base consists entirely of adults in a professional environment. The interface should reflect this by casting a professional and minimal design.

### Software Interface

The largest software interface is the MVC framework itself. Java Play 2.4.x framework will be used to implement the system and its documentation can be found here: https://www.playframework.com/documentation/2.4.x/Home.

Additional web services include:

Twitter Bootstrap: http://getbootstrap.com/

jQuery: https://api.jquery.com/

Lastly, the PostgreSQL database will be used in development. Commerce may choose to swap this with another database, but PostgreSQL documentation may be found here: http://www.postgresql.org/docs/

# Functional Requirements

**Use Cases:**

|  |  |
| --- | --- |
| **Title:** | Manual Creation of a Case |
| **Use case ID:** | 1 |
| **Value** | 9 |
| **Cost** | 9 |
| **Description:** | System interactions for a user manually creating a new DNE case. |
| **Basic Flow:**   1. This use case begins when user clicks on 'create new case' button from home screen. 2. The system displays a case creation screen that allows the user to select a case type and subtype. 3. User selects desired case type and subtype. 4. System loads screen with appropriate fields for request case. It automatically generates a new case ID for the case. 5. The user fills out the form presented on the page and selects 'submit'. 6. The system generates the auditing information for the new case and stores all the data in a database. The system informs the user the case was created successfully. 7. The system redirects the user back to the home screen. | |
| **Alternate Flows:**  6a. Unsuccessful case creation (invalid data in one or more fields).   1. The system displays a message that the case was not created successfully 2. The system redirects the user to the case creation page and informs them which fields caused a problem. 3. The user corrects the data and selects submit again when finished.   6b. User must quit case creation and resume at a later time.   1. User clicks on ‘Save as Draft’ button 2. Case is created and marked as incomplete 3. Case is saved into the database | |
| **Exceptions:**  None | |
| **Open issues:**  1. Should the system check to see if a case already exists before storing it in the database? | |

|  |  |
| --- | --- |
| **Title:** | Automatic generation of cases using NACHA file |
| **Use case ID:** | 2 |
| **Value** | 8 |
| **Cost** | 7 |
| **Description:** | System interactions for a user importing a nacha file. |
| **Basic Flow:**   1. User clicks on ‘import nacha file’. 2. System displays file browser for user to select a file from hard drive. 3. User selects a file. 4. System parses the file and generates a case with the correct information for each item in the file. 5. The system generates the auditing information for the new case and stores all the data in a database. The system informs the user the cases were created successfully. 6. The system redirects the user back to the home screen. | |
| **Alternate Flows:**  4a. Unsuccessful parsing or incorrect file type.   1. The system displays a message that the file was not successfully imported and states whether the failure is due to an incorrect file type or a parsing error. 2. No data is stored and the system goes back to the main menu screen. | |
| **Exceptions:**  None | |
| **Open issues:**  1. Should the system check to see if a case already exists before storing it in the database?  2. Should the system allow incomplete case fields? | |

|  |  |
| --- | --- |
| **Title** | Edit a Case |
| **Use case ID:** | 3 |
| **Value** | 8 |
| **Cost** | 6 |
| **Description:** | System interactions for user editing data in an existing case. |
| **Basic Flow:**   1. This use case begins once a user has selected a case to edit. 2. System displays all of the fields associated with the case. 3. User enters new values for one or more fields and selects ‘save’. 4. The system generates the auditing information for the new case data and stores all the data in a database. The system informs the user the case was updated successfully. 5. The system redirects the user back to the home screen. | |
| **Alternate Flows:**  4a. Unsuccessful case edit.   1. The system displays a message that the case was not successfully updated and states which fields caused the problem. 2. The user fixes the problem fields and submits the data gain. The system continues at step 4 above. | |
| **Exceptions:**  None | |
| **Open issues:** | |

|  |  |
| --- | --- |
| **Title** | Search Ability |
| **Use case ID:** | 4 |
| **Value** | 7 |
| **Cost** | 7 |
| **Description:** | System interactions for searching for a case based on various parameters. |
| **Basic Flow:**   1. User enters search parameter into form and selects field to search by. Possible fields include date, case state, SLA deadlines, type, subtype, and beneficiary name. 2. User selects 'Start Query'. 3. System displays a list of all cases that contain the search value in the specified field. Results are paginated if needed. 4. The system displays the number of results returned by the query. | |
| **Alternate Flows:**  3a. No search results   1. The system displays an empty list with message that no results matched the search query. | |
| **Exceptions:**  None | |
| **Open issues:**  1. Should search results be sorted in some way?  2. Can add more search filters if time allows.  3. Are there security concerns with malicious search queries? Preventative measures (eg query validation/scrubbing) | |

|  |  |
| --- | --- |
| **Title** | Home Screen Navigation |
| **Use case ID:** | 5 |
| **Value** | 6 |
| **Cost** | 4 |
| **Description:** | Main menu UI for application and its button routing. |
| **Basic Flow:**   1. This use case begins when the user starts the application. 2. The system loads the home screen where the user is able to select from different main options: Create a case, import a file, search a case, edit a case. When authentication is built in at a later point, a login screen will serve as the default application start screen. However, the team is continuing under the assumption that there will not be time to implement authentication. 3. System displays a list of all cases whose SLA's are nearing deadline or past. Results are paginated if needed. | |
| **Alternate Flows:** | |
| **Exceptions:**  None | |
| **Open issues:**  1. If authentication is implemented, it may be better to specifically display cases that the logged in user has personally worked on instead of generic cases near their deadline. | |

|  |  |
| --- | --- |
| **Title** | Auditing |
| **Use case ID:** | 6 |
| **Value** | 7 |
| **Cost** | 8 |
| **Description:** | System interactions for viewing the audit history of a case. |
| **Basic Flow:**   1. This use case begins once a user has selected an existing case from a query to view its details. 2. The system displays the case's data and an option to view auditing related to that case. 3. User selects 'View Audits'. 4. The system displays the date of the case creation and edits along with what was changed and who made the change. | |
| **Alternate Flows:** | |
| **Exceptions:**  None | |
| **Open issues:** | |

|  |  |
| --- | --- |
| **Title** | SLA Creation and Visibility |
| **Use case ID:** | 7 |
| **Value** | 5 |
| **Cost** | 7 |
| **Description:** | SLAs are set according to user input and can be viewed inside a case or through search results. |
| **Basic Flow:**   1. User selects type "Government Reclamations" with any of the following subtypes: DNE, DCN, CRF, & Gov Reclamations. 2. The system sets the SLA of the case to 3 business days after the case opened. 3. The SLA of a case can be seen by selecting any case to view its properties. Cases should also indicate how long they have been open. | |
| **Alternate Flows:**  1a. User checks the "Watch Item" checkbox for DNE, DNC, CRF, or Gov Reclamations subtype.   1. The system sets the SLA to 7 business days from the effective date.   1b. User selects one of the following subtypes: Treasury Referral or Treasury Refund.   1. The system sets the SLA to 7 business days from after the case was opened.   3a. User modifies case data. (User is working on a case).   1. The system resets the SLAs for the basic flow and 1a, but not SLA related to 1b.   3b. User searches for a case   1. The system displays a list of cases including the SLA field beside each result. | |
| **Exceptions:**  None | |
| **Open issues:**  1. Should search results be sorted in some way? | |

|  |  |
| --- | --- |
| **Title** | Case Status |
| **Use case ID:** | 8 |
| **Value** | 4 |
| **Cost** | 4 |
| **Description:** | State of cases are tracked and indicated |
| **Basic Flow:**   1. User selects a case to view. The case is closed. 2. The system displays the case's data along with an indicator that the case is closed. No 'Edit Case' option is given to user, however a 'Reopen Case' option is. 3. User selects 'reopen case'. 4. The system displays the 'edit case' button and allows the case to be modified. Audit information is stored. | |
| **Alternate Flows:**  1a. User selects a case to view. The case is open.   1. The system allows the user to edit data and an option to move the case to a closed state. 2. User selects 'close case'. 3. The system changes the case state to closed, records the audit, and updates the case view screen to now show the case is closed. | |
| **Exceptions:**  None | |
| **Open issues:** | |

|  |  |
| --- | --- |
| **Title** | Sorting of Search results |
| **Use case ID:** | 9 |
| **Value** | 3 |
| **Cost** | 5 |
| **Description:** | Search results are sorted by different user specified parameters in order to make viewing data easier. |
| **Basic Flow:**   1. User searches for a case. 2. The system displays the search results and offers options for sorting the data by fields. Cases are sorted by date by default. Other sorting options include SLA deadline, case state, and type/subtype. 3. User selects a sorting option. 4. The system reloads the page with the same data list, except sorted by the selected parameter. | |
| **Alternate Flows:** | |
| **Exceptions:**  None | |
| **Open issues:** | |

|  |  |
| --- | --- |
| **Title** | Export to Excel |
| **Use case ID:** | 10 |
| **Value** | 2 |
| **Cost** | 4 |
| **Description:** | System interactions for exporting search results to excel |
| **Basic Flow:**   1. User searches for a case. and selects 'Export to Excel'. 2. The system parses all of the data for every case in the search results into a csv file that can be consumed by excel. 3. The system displays a message that the results were successfully exported and informs the user where the csv file is located. | |
| **Alternate Flows:** | |
| **Exceptions:**  2a. System encounters a parsing error.   1. The system abandons its task for creating a csv file and informs the user that something went wrong while parsing the data. | |
| **Open issues:** | |