**CMS - Medicaid Provider Screening Portal - Administrative Features 1.0 Requirements Specification**

***This document outlines the Application Scope and Requirements for   
CMS - Medicaid Provider Screening Portal - Administrative Features 1.0.***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **PRESENTED TO:** <Client Names> | **<CLIENT LOGO>** | | **PRESENTED BY:** <TopCoder Names> | topcoder_logo | |

Revision History

|  |  |  |
| --- | --- | --- |
| **Author** | **Revision Number** | **Date** |
| <handle goes here after final fix> | 1.0 | 09/15/2012 |
|  |  |  |
|  |  |  |

[Application Requirements Specification 5](#_Toc335471325)

[1. Scope 5](#_Toc335471326)

[1.1 Overview 5](#_Toc335471327)

[1.2 Objectives 5](#_Toc335471328)

[1.3 Limitations & Assumptions 8](#_Toc335471329)

[2. Logic Requirements 9](#_Toc335471330)

[2.1 Login [External] 10](#_Toc335471331)

[2.2 Logout [External] 10](#_Toc335471332)

[2.3 View Homepage 11](#_Toc335471333)

[2.4 Get Help 17](#_Toc335471334)

[2.5 Search Enrollments 20](#_Toc335471335)

[2.6 Manage My Profile 23](#_Toc335471336)

[2.7 Query Enrollment Status 26](#_Toc335471337)

[2.8 View Enrollments 29](#_Toc335471338)

[2.9 View Enrollment Details 33](#_Toc335471339)

[2.10 Create New Enrollment 37](#_Toc335471340)

[2.11 Edit Enrollment 42](#_Toc335471341)

[2.12 Export Enrollment Data 48](#_Toc335471342)

[2.13 Print Enrollment Data 51](#_Toc335471343)

[2.14 Initiate On-Demand Screening 52](#_Toc335471344)

[2.15 Initiate Scheduled Screening 55](#_Toc335471345)

[2.16 Approve Enrollment 58](#_Toc335471346)

[2.17 Reject Enrollment 61](#_Toc335471347)

[2.18 Manage Provider Types 65](#_Toc335471348)

[2.19 Manage Screening Schedules 71](#_Toc335471349)

[2.20 Manage Help Topics 73](#_Toc335471350)

[2.21 Manage Agreements and Addendums 76](#_Toc335471351)

[2.22 Manage Service Agents 80](#_Toc335471352)

[2.23 Search Users 82](#_Toc335471353)

[2.24 Manage Users 83](#_Toc335471354)

[2.25 Integrate With External Systems 87](#_Toc335471355)

[2.26 Send Email Notifications 88](#_Toc335471356)

[2.27 Perform Auditing 89](#_Toc335471357)

[2.28 Perform Logging 91](#_Toc335471358)

[3. General Requirements 93](#_Toc335471359)

[3.1 Graphical User Interface Requirements 93](#_Toc335471360)

[3.2 Performance Constraints 93](#_Toc335471361)

[3.3 Security 94](#_Toc335471362)

[4. Required Documentation 95](#_Toc335471363)

[4.1 Specification Documentation 95](#_Toc335471364)

[5. Help / User Documentation 95](#_Toc335471365)

[6. Notes 95](#_Toc335471366)

[7. Future Enhancements 95](#_Toc335471367)

[8. Glossary 96](#_Toc335471368)

[8.1 Definitions 96](#_Toc335471369)

[8.2 Acronyms 96](#_Toc335471370)

# Application Requirements Specification

# Scope

## Overview

The client for this project compensates US medical service providers through a program called Medicaid.  Medicaid is a healthcare program that is funded by taxes paid by U.S. citizens and it is operated by individual states. Medicaid program provides health benefits to more than 60 million Americans which include children, parents, senior citizens, pregnant women and disabled people.

The current business process requires medical service providers to register themselves in their individual state for compensation under Medicaid program by filling out many forms.

The process of screening claims from service providers is manual. The screener receives an application from a provider. The screener first checks the provider has given all required information and license information. The screener will then verify provider information at several websites. This verification includes provider’s information, like medical license and its validity and background check to see if provider has been involved in any violations or suspensions previously. Also the screener checks if the provider has any criminal history.

If any information in forms is incomplete or incorrect, the screener will interact with provider to correct these. After correcting errors, the provider has to resubmit the application for final screening.

However, the service provider is not comfortable in filling paper forms as they are very inconvenient and error prone. Service providers have to do lot of revisions before going for final submission.

Due to the issues mentioned above, along with several other shortcomings of the current process (explained below), Thus the client has decided to replace the existing manual process with a single automated system.

The main goal of the project is to deliver an efficient solution for implementing automated system to ease the enrollment and screening process. Also the portal will be able to automatically screen potential providers for fraud risk before they are approved for payments.

Thus by creating this new application, the client will be able to save the amount of time that takes to do enrollment and screening manually, also able to reduce maintenance costs associated with the process.

This specification will produce the detailed requirements on the administrative features part of the application.

## Objectives

The objectives of the entire CMS Medicaid Provider Screening Portal application are as following (this specification covers just a small part (administrative features -related) of that main application).

Must have:

* To design an application that will automate the process of screening and enrollment for service providers.
* To design a new portal application that reduces the time needed by service providers to submit application claim forms.
* To design a new application which will improves accuracy in making enrollment decisions.
* To design a new application which will reduces the time required to make enrollment decisions.
* To design and an MITA 3.0 compliant application.
* To design an application that implements a multi-tenant provider screening solution to save administrative/infrastructure costs.
* To design and build a new application that can be integrated into an open source solution that can deliver a reliable, scalable, and cost-effective provider-screening capability US states.
* To design an application that will build a database of service providers for use by any state of USA.
* To design an application that which Improves availability of provider data.
* To design an application that effectively reduces the time need to get details of providers by queering respective data sources regarding provider.
* To design an application that will support Risk-Based Provider Screening for all new and re-enrolling providers.
* To design an application that will improve the ability to implement Affordable Care Act provisions in Medicaid/CHIP.
* To design and build a new application that will have the ability to implement sections 1902(a)(39) and 1902(a)(77) of the Social Security Act, as amended by the Patient Protection and Affordable Care Act (P.L. 111-148 and 111-152) subsections 6401(b) and (c) (Provider Screening and Other Enrollment Requirements Under Medicare, Medicaid, and CHIP), and section 6501 (Termination of Provider Participation Under Medicaid if Terminated by Medicare or Other State Plan).
* To design and build a new application that will help in driving alignment of the Medicaid Information Technology Architecture (MITA) 3.0 framework to the Information and Technology Architecture levels
* To design an application that is deployable as a hosted service offering (SaaS).
* To design an application that nicely integrates with existing systems like license databases.
* To design an application that periodically checks existing systems for updates.
* To design an application that complies with section-508 of the Americans with Disabilities Act requirements.
* To design an application that should pass DHS audit testing.
* To design an application that allows scheduled or immediate payment processing.
* To design an application that is able to do Identity Verification:
* To design an application so as to support easy addition of 'Help Features' later
* To build an application that easily recognizes what applications need to submit an application fee and should be able to collect that fee.
* To design an application that supports multiple user types (roles).
* To build an application with the ability to send email notifications when specific enrolment is submitted, approved or denied.
* To design an application that complies with data privacy standards and other corporate standards.
* To have the application support logging. Save log data of all errors, exceptions, and warnings to log records.
* To save audit records for all activity initiated by the user.
* To save application usage metrics (as part of auditing).  
  To design an application to have the capability to Conduct Identity Verification
* To design an application to have the capability to Build Provider Profiles
* To design an application to have the capability to Evaluate and Maintain the Integrity of the Results
* To design an application that will allow admin(service and system) to find specific provider information with minimal search attempts
* To design an application to have the capability to identify applicants, including individual providers and owners of institutional providers.
* To design an application to have the capability to verify identity and prior history of problems with Medicaid/CHIP or Medicare programs.
* To design an application to have the capability to identify and schedule revalidation process.
* To design an application which allows changing the language of the application
* To design an application which has the capability to create a learning system to ensure that observed negative trends factor back into screening rules so as to flag suspicious enrollments early in the screening process, ensuring the ability to detect and reduce/eliminate the incidence of false positives.

Nice to have:

* None.

Project Return on Investment (ROI Metrics) are as following:

* Reduced support and maintenance costs resulted in helping service providers to fill forms.
* Reduced time to enroll service provider.
* Reduced time to screen claims by service providers
* Reduce time to find fraud providers
* Increased money savings by finding fraud providers before they get paid.

General Capacity Metrics:

* Expecting 150,000+ unique users
* Concurrent users expected will be in thousands

General Usability Metrics:

* The new application will be user friendly and provide a simplified interface for CMS - Medicaid Provider Screening Portal - Administrative Features functionality.

## Limitations & Assumptions

The limitations of this project are listed below:

* The help features is not defined, this may trouble service providers in understanding the new application.
* The identity and license verification is done by interacting with external systems.
* The setup and management of user roles is out of scope for this project. Both of these security features will be handled by the existing security system.
* Presently, the application will only be available in English. Later it may support multi-language.
* This application will not pay providers. Payment to providers is handled externally or may be through other systems.
* Screening rules are not part of this contest definition.

Assumptions critical to the success of this project are listed below:

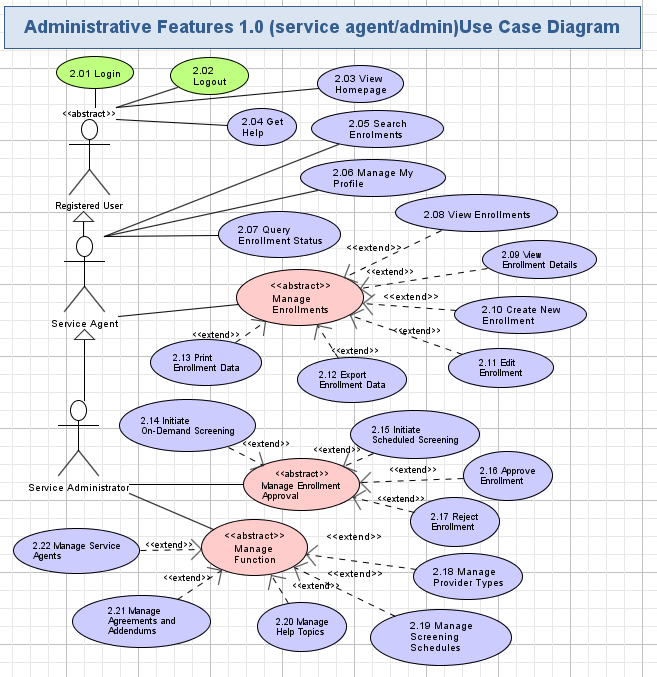
* The application will be networked and browser-based.
* The application for this project is new. Hence the architecture and implementation will be fully new.
* Users of the new application will be using a PC, web browser, a keyboard and a mouse (or equivalent).
* Users of the system will have Internet access.
* Users can access the application from multiple US time zones, but that must not affect anything in the application. The application must be time zone agnostic.
* The application does not work with any peripheral devices other than printer and scanner.
* The application will be accessed through secure socket layer (SSL).
* The state of Minnesota will continue using MN-ITS mailbox for sending email notifications and other states may have their own similar system or they can use the application’s email notification service.

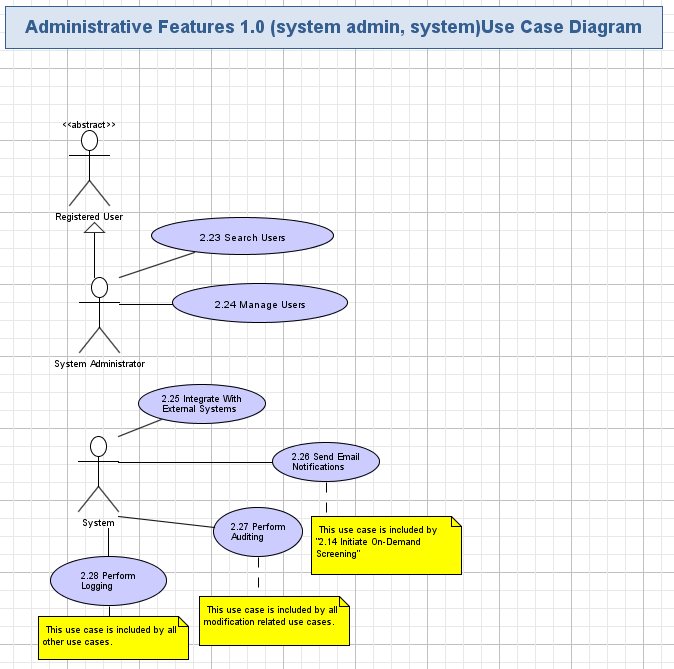
**Technical Assumptions:**

* The application should be built using web application technologies. The application can be developed using the following technology stack:
  + Java Server Faces 2.0 ,
  + WESB,
  + MQ Series 6.0,
  + WebSphere Process server (WPS) 7.0.04
  + WebSphere enterprise Service Bus (WESB) 7.0.04
  + Oracle database11g,
  + limited java services (JAXWS and JAXB),
  + WebSphere service Registry and Repository (WSRR)
  + WAS 7.0.0.21
  + J2EE 6.0
  + JAVA SDK 5.0.22
  + Hibernate 3.6.3
  + Spring 3.0.5
* The application will support all modern browsers: Firefox, Internet Explorer, Safari, and Chrome (specific versions still an open issue).

# Logic Requirements

Use case diagram is shown below:





## Login [External]

The application will allow registered users to login by entering their credentials. The application validates user credentials and performs authentication/authorization.

Conceptualization Reference: 4.4.1 Log In (external)

Wireframe Page: Login.html

* Pre-conditions: the non logged-in user opened any page of the application.
* Post-conditions: the user was successfully logged in, can access application functionality.

## Logout [External]

The user can logout at any time from the application to ensure high security. The logout can be also performed automatically due to user’s session expiration.

Conceptualization Reference: 4.4.2 Log Out (external)

Wireframe Page: All pages except “Login.html”

* Pre-conditions: the logged in user pressed “Logout” hyperlink or the user session has expired.
* Post-conditions: the user was logged out from the application and cannot work with his/her application until login again.

## View Homepage

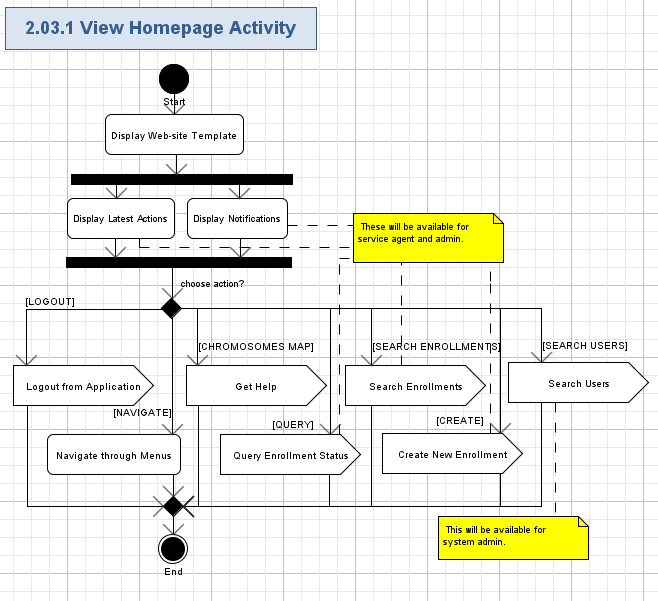
The application will show a home page for the user after he/she has just logged in. It will show the header, top navigation menu, main area and the footer. This page is like a template for displaying all the other web-pages of the application and it will allow to navigate through the most important parts of the application.

Conceptualization Reference: 4.4.3 View Homepage

Wireframes page: “Dashboard\_\_Service\_Admin\_.html”, “Dashboard\_\_Service\_Agent\_.html” and “User\_Accounts\_\_System\_Admin\_.html”.

* Pre-conditions: the user has just logged in to the application or pressed “Dashboard” option in the top navigation menu. The user has to be logged in to the application to perform this action.
* Post-conditions: the homepage (template) of the application was displayed, and the user can choose some actions on that page or navigate to some other features of the application.

### View Homepage Activity

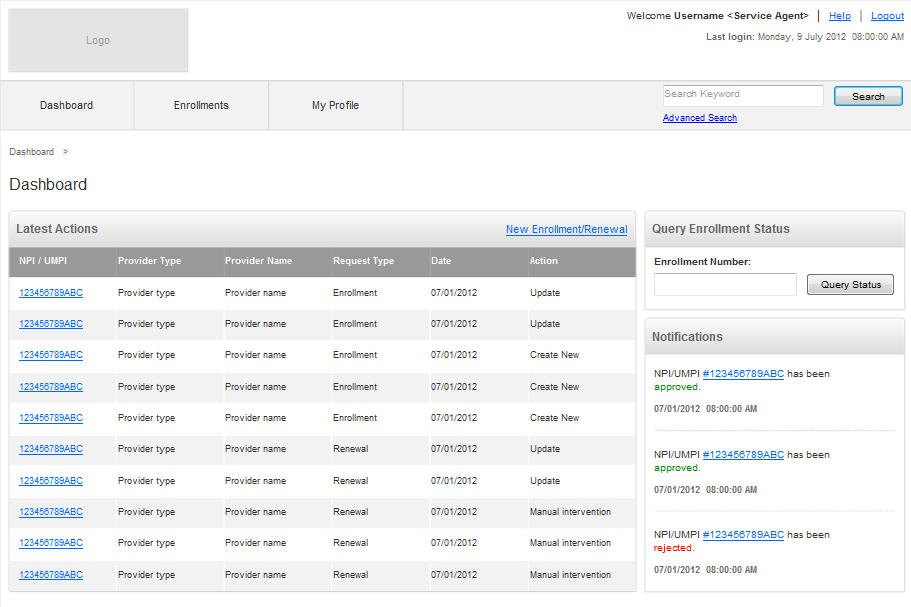


#### Display Web-site Template

* The application will display the template of the web-site.
* The template will be shown on any page of the application, so it will be described just once below.
* The template has a heading part, top menu, small navigation menu, main middle/center area, and a footer.
* The application displays a Client company logo in the left top side of the header.
* The application will display a welcome message (String, max 100 char, non empty) part, and last login (String, max 100 char, non empty) on the right of the header.
* The application will display the “Logout” hyperlink in the rightmost part of the header – please refer chapter 2.2 for details.
* The application will display the “Help” hyperlink in the rightmost part of the header – please refer chapter 2.4 for details.
* The application will display the top menu with options to quickly open the related parts of the application – please refer chapter 2.3.1.10 for more details.
* The application will show the small navigation menu with hierarchy of the current web-page in the web-site, starting from the Dashboard (homepage).
* The application will show the name of the currently shown web-page just below the small navigation menu, like:

|  |
| --- |
| *Dashboard* |

* The application will use the main area for displaying the main data, which the user works with (for example, list of users, latest actions, etc.). So, it can be recognized as a content area.
* The application will use the footer below the main area of the web-page.
* The application can display, for example, copyright information (String, max 50 chars, non empty) in the footer.
* The screen shot example of the homepage (dashboard) of the web-site is as following:

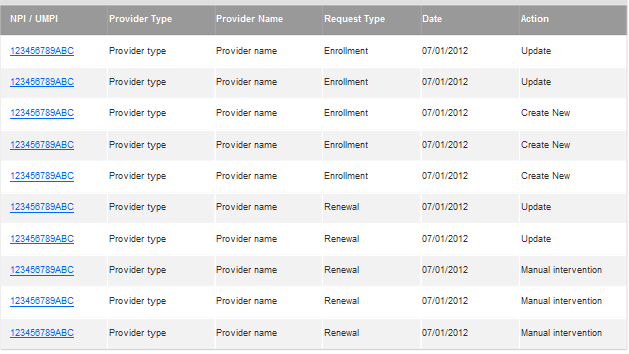


#### Display Latest Actions

* The application will show latest actions of the logged-in service admin or service agent.
* The application will display latest actions on the table.
* Each row in the table will correspond to the action and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| NPI/UMPI | The NPI/UMPI of the enrollment. | String, 20 chars, non empty.  Link to view the enrollment details. | Y |
| Provider Type | The provider type of the provider. | String, 20 chars, non empty | Y |
| Provider Name | The provider name of the provider. | String, 100 chars, non empty | Y |
| Request Type | The type of the request. | String, 20 chars, non empty | Y |
| Date | The date when the action was performed. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Action | The name of the action. | String, 20 chars, non empty | Y |

* The screen shot example of the latest actions is as following:

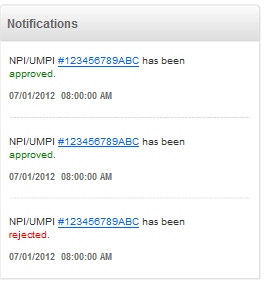


#### Display Notifications

* The application will show notifications for the logged-in service admin or service agent.
* The application will display list of notification messages on the page.
* Each message in the list will contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Message | The description of the notification. | String, 1024 chars, non empty | Y |
| Date/Time | The date/time when the notification was sent. | String, 22 chars, non empty.  Date format: MM/DD/YYYY HH:MM:SS AM/PM | Y |

* The screen shot example of the notifications is as following:



#### Logout from Application

* The user can perform logout from the application by pressing “Logout” hyperlink.
* Please refer to chapter 2.2 for more details.

#### Get Help

* The user can get help information on the application usage by pressing the “Help” hyperlink.
* Please refer to chapter 2.4 for more details.

#### Search Enrollments

* The service agent/service admin can search enrollments (advanced or basic mode) by pressing the Search/Advanced Search hyperlink on the page.
* Please refer to chapter 2.5 for more details.

#### Search Users

* The system admin can search users (advanced or basic mode) by pressing the Search/Advanced Search hyperlink on the page.
* Please refer to chapter 2.23 for more details.

#### Query Enrollment Status

* The service agent/service admin can query the enrollment status with the enrollment number.
* Please refer to chapter 2.7 for more details.

#### Create New Enrollment

* The service agent/service admin can create new enrollment from the homepage by pressing the “New Enrollment/Renewal” hyperlink.
* Please refer to chapter 2.10 for more details.

#### Navigate through Menus

* The user can navigate through the following top menu options:
  + Dashboard – an ability to return to the homepage (dashboard) from any other web-page of the application. The application will redirect to the web-page, described in the current chapter.
  + [for service agent/ admin only] Enrollments – an ability to manage the enrollments. Please refer chapter 2.8 – 2.13 for more details.
  + [for service agent/ admin only] My Profile – an ability to view the profile and update the profile if needed. Please refer chapter 2.06 for more details.
  + [for Service Admin only] Functions –an ability to manage the service management functions. Please refer chapter 2.14 – 2.22 for more details.
  + [for System Admin only] User profiles – an ability for system Admins to manage users in the application. Please refer chapter 2.24 for more details.
* There will be a special small navigation menu, shown on each web-page. The screen shot example is as following:  
  
* That navigation menu shows the position of the currently shown web-page in the web-pages hierarchy of the application starting from the Dashboard (homepage).
* The user can navigate back to the previous web-pages in the hierarchy, displayed on the navigation menu, by pressing the related hyperlink.

## Get Help

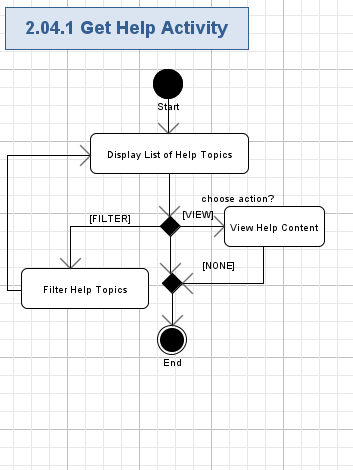
The application will provide help information to the user by request. There will be a list of the available help topics, which can be searched for, viewed by the user.

Conceptualization Reference: 4.4.4 View Help

Wireframes page: “Help\_\_Service\_Agent\_.html” and “Help\_\_Service\_Admin\_.html”.

* Pre-conditions: the user needed to get help on the application usage. The user has to be logged in to the application to perform this action.
* Post-conditions: the application shown the list of help topics and the user could view any of them.

### Get Help Activity



#### Display List of Help Topics

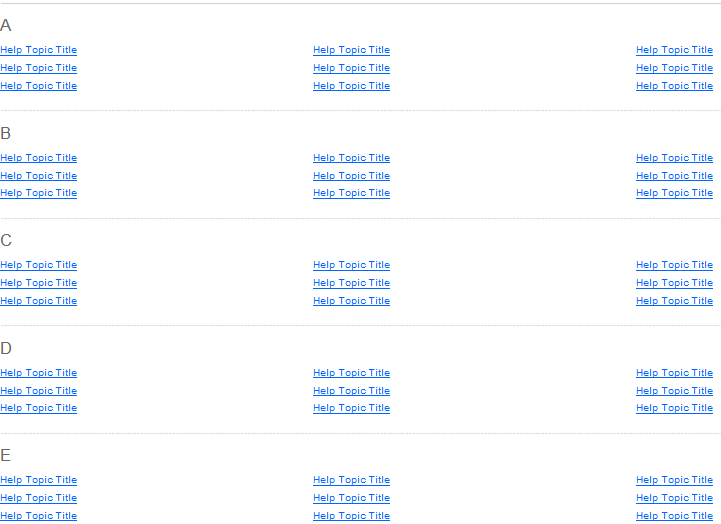
* The application will show the header of the page like following:

|  |
| --- |
| *Help Topics* |

* The application will display list of help topics.
* Each topics in the list will correspond to the related help topic and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| <Name> | The name of the help topic. | String, max 100 chars, non empty.  It will be a hyperlink to view the related help content– please refer chapter 2.4.1.3 for more details. | Y |

* The topics will be grouped by “A”, “B”, “C”, ….
* The screen shot example of the help list is as following:



#### Filter Help Topics

* The application will show alphabetical reference to all the help topics.
* The alphabetical reference will be applied to the names of the topics.
* The “All”, A, B, C, … , Z hyperlinks will be present on the alphabetical reference.
* “All” reference will be selected by defaults – it means all the help topics will be shown on the list.
* The user can press any hyperlink (i.e. English letter) in the alphabetical reference and filter the list of help documents by topic names, starting from that chosen letter (or All – for all the documents).
* The screen example of filtering for help topics is as following:



#### View Topic Content

* The user can view any of the help topics, shown on the list by pressing the topic’s name hyperlink.
* The content of the help topic will contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| <Name> | The name of the help topic. | String, max 100 chars, non empty. | Y |
| Description | The description of the help topic | String, max 10,000 chars, non empty. | Y |

## Search Enrollments

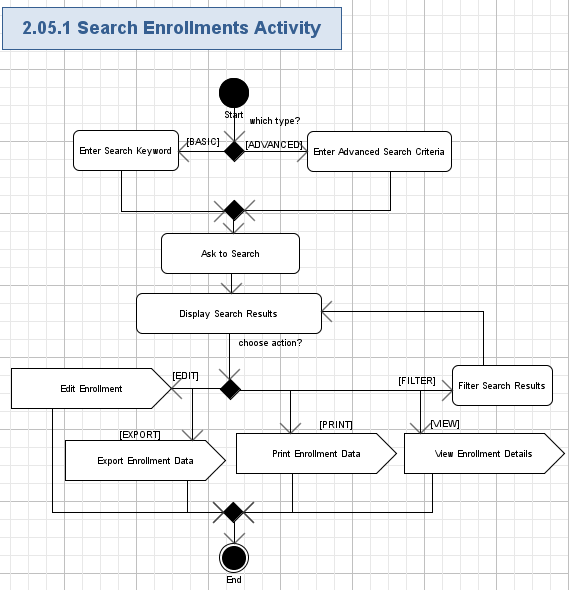
The application allows the users to search enrollments with given criteria. The user can perform basic search or advanced search on enrollments. The user can choose to view the details of the enrollment (refer chapter 2.9 for details), edit the details of the enrollment (refer chapter 2.11 for details), edit the enrollment data (refer chapter 2.12 for details) or print the enrollment data (refer chapter 2.13 for details).

Conceptualization Reference: 4.4.17 Search Enrollments

Wireframes page: “Search\_Result\_\_Service\_Admin\_.html”, “Search\_Result\_\_Service\_Agent\_.html”, “Advanced\_Search\_\_Service\_Admin\_.html” and “Advanced\_Search\_\_Service\_Agent\_.html”

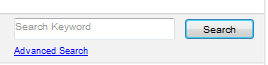
* Pre-conditions: the user needed to search the enrollments in the application. The user has to be logged in to the application to perform this action.
* Post-conditions: the application shown the found enrollments and the user can choose perform some actions on the searched results.

### Search Enrollments Activity



#### Enter Search Keyword

* The application will show the textbox on the homepage (template) for the user to enter the keyword (String, max 100, can be empty) to search the enrollments in the application.
* The screen shot example is like following:

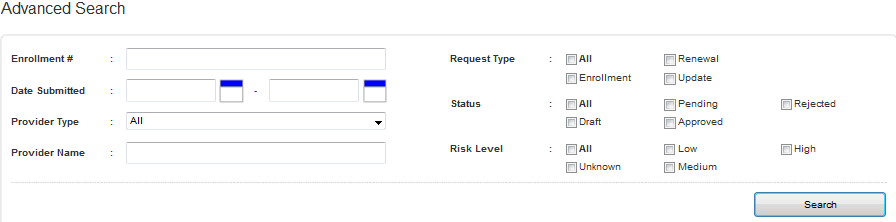


#### Enter Advanced Search Criteria

* The user can also choose to perform advanced search by pressing the “Advanced Search” on the homepage (template).
* The application will show following criteria for the user to enter for searching enrollments:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Enrollment # | The number of the enrollment. | String, max 20 chars, can be empty | N |
| Date Submitted From | The start of date range of the enrollment submitted. | String, max 10 chars, can be empty.  Date format: MM/DD/YYYY.  The user can choose the value from the calendar. | N |
| Provider Type | The type of the provider. | String, max 20 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |
| Provider Type | The name of the provider. | String, max 100 chars, can be empty. | N |
| Request Type | The type of the request. | Checkboxes with values:  All;  Renewal;  Enrollment;  Update; | N |
| Status | The status of the enrollment. | Checkboxes with values:  All;  Pending;  Rejected;  Draft;  Approved. | N |
| Risk Level | The level of the risk of the enrollment. | Checkboxes with values:  All;  Low;  High;  Unknown;  Medium | N |

* The screen shot example is like following:



#### Ask to Search

* The user can ask the application to perform search enrollments by pressing the “Search” button.

#### Display Search Results

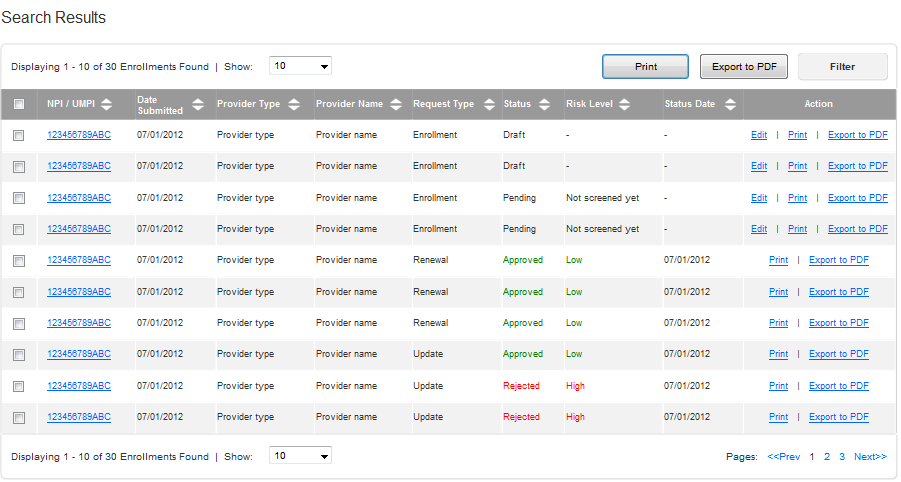
* The application will search enrollments with the given criteria/keyword.
* The list of results will be shown in a page-able table with the following columns:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| NPI/UMPI | The NPI/UMPI of the enrollment. | String, 20 chars, non empty.  It will be the hyperlink to view the details of the enrollment (refer chapter 2.9 for more details). | Y |
| Date Submitted | The date when the enrollment was submitted. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Provider Type | The provider type of the provider. | String, 100 chars, non empty | Y |
| Provider Name | The provider name of the provider. | String, 100 chars, non empty | Y |
| Request Type | The type of the request. | String, 20 chars, non empty | Y |
| Status | The status of the enrollment. | String, 20 chars, non empty | Y |
| Risk Level | The risk level of the enrollment. | String, 20 chars, can be empty | N |
| Date | The date when the action was performed. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Status Date | The date when the enrollment was approved/rejected | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| <Edit> | The hyperlink to edit the enrollment. | Hyperlink.  The rejected/approved can not be edited. | N |
| <Print> | The hyperlink to print the enrollment data. | Hyperlink. | Y |
| <Export to PDF> | The hyperlink to export the enrollment data. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z> Enrollments Found* |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of records.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The user can navigate through the table pages by using standard navigation buttons, mentioned above.
* The user can sort the table with any column (ASC/DES).
* The screen shot example of the search results is like following:



#### Filter Search Results

* The user can also choose to filter the displayed enrollments with the following criteria:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| NPI/UMPI | The NPI/UMPI of the enrollment. | String, 20 chars, can be empty. | N |
| Date Submitted From | The start of date range of the enrollment submitted. | String, max 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Date Submitted To | The end of date range of the enrollment submitted. | String, max 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Provider Type | The provider type of the provider. | String, max 100 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |
| Provider Name | The provider name of the provider. | String, 100 chars, can be empty. | N |
| Request Type | The type of the request. | String, max 20 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |
| Risk Level | The level of the risk of the enrollment. | String, max 20 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |

* The screen shot example is like following:



#### View Enrollment Details

* The user can choose to view the details of the enrollment by pressing the “NPI/UMPI” link on the chosen enrollment.
* Please refer chapter 2.9 for more details.

#### Edit Enrollment

* The user can choose to edit the details of the enrollment by pressing the “Edit” link on the chosen enrollment.
* Please refer chapter 2.11 for more details.

#### Export Enrollment Data

* The user can choose to export the data of the enrollment by pressing the “Export to PDF” link on the chosen enrollment.
* Please refer chapter 2.12 for more details.

#### Print Enrollment Data

* The user can choose to print the data of the enrollment by pressing the “Print” link on the chosen enrollment.
* Please refer chapter 2.13 for more details.

## Manage My Profile

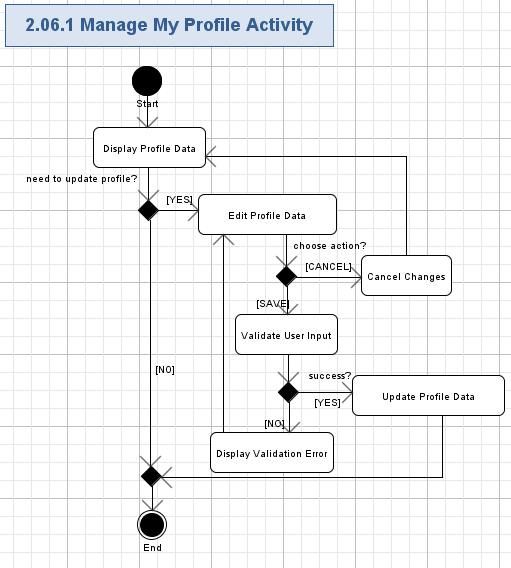
The application will provide “My Profile” page for the user. The user can freely update his/her profile on that page on the application. The profile information will be displayed as well.

Conceptualization Reference: 4.4.14 View User Profile, 4.4.15 Update User Profile

Wireframes page: “Update\_Profile\_\_Service\_Admin\_.html”, Update\_Profile\_\_Service\_Agent\_.html”

* Pre-conditions: the user pressed “My Profile” option in the top navigation panel. The user has to be logged in to the application to perform this action.
* Post-conditions: the application displayed the user profile and the user updated his/her profile.

### Manage My Profile Activity

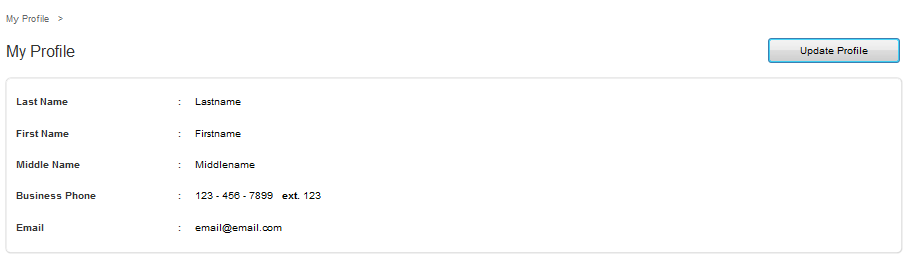


#### Display Profile Data

* The user can ask the application to manage the profile data by pressing “My Profile” option in the top navigation menu.
* The application will provide a simple single page where all fields of the user profile are shown.
* The application will display the following fields with the current data for the existing user’s profile:

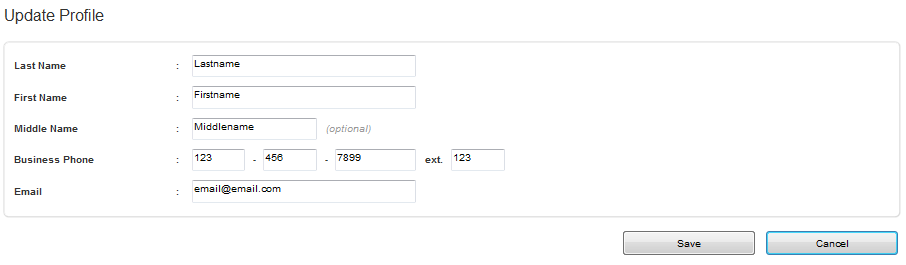
| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| Business Phone | The business phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, non empty | Y |

* The screen shot example of displaying the existing user profile is like following:



#### Edit Profile

* The user can ask the application to edit the profile by pressing “Update Profile” button.
* Please refer to the table from chapter 2.5.1.1 for more information about profile fields, which can be edited.
* The screen shot example of editing profile is like following:



#### Cancel Changes

* The user can press “Cancel” button to reject updating of the existing profile.
* All the fields on the existing profile will be reverted to their previous state (after the previous successful save of the profile).
* No updates of the existing profile will be performed.
* All the profile fields are reverted to the non-editable state and the user will view his/her profile as described in chapter 2.5.1.1

#### Validate User Input

* The user will press “Save” button to update user profile fields according to the edited fields.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.5.1.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Update Profile Data

* The application will update an existing profile with the edited data.

## Query Enrollment Status

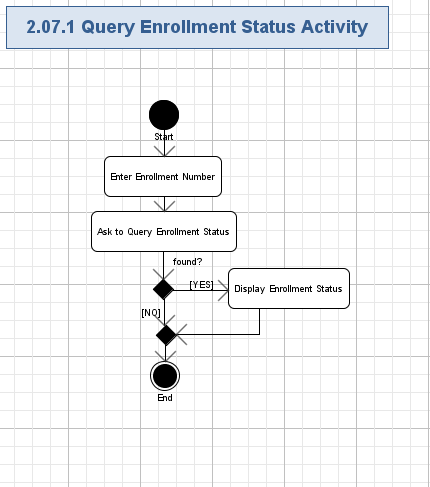
The application allows the users to query the status of the enrollment by providing the enrollment number. The application will display the status information if there is any enrollment found with the given enrollment number. The status of the enrollment can be “Draft”, “Pending”, “Approved” or “Rejected”.

Conceptualization Reference: 4.4.8 Query Enrollment Status

Wireframes page: “Status\_Query\_\_Service\_Admin\_.html”, Status\_Query\_\_Service\_Agent\_.html”

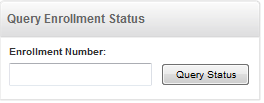
* Pre-conditions: the user needed to query the enrollment status. The user has to be logged in to the application to perform this action.
* Post-conditions: the application shown the status of found enrollment; or the application did not display anything if no enrollment found.

### Query Enrollment Status Activity



#### Enter Enrollment Number

* The application will show the textbox on the homepage (template) for the user to enter the enrollment number (String, max 100, can be empty) to query the enrollment status in the application.
* The screen shot example is like following:



#### Ask to Query Enrollment Status

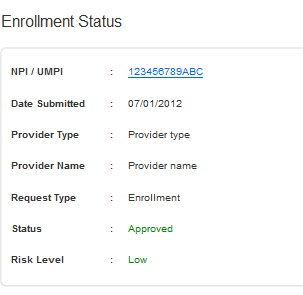
* The user can ask the application to perform querying on the enrollment status by pressing the “Query Status” button.

#### Display Enrollment Status

* The application will query the enrollment with the given enrolment number.
* The status of the enrollment will be shown with the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| NPI/UMPI | The NPI/UMPI of the enrollment. | String, 20 chars, non empty.  It will be the hyperlink to view the details of the enrollment (refer chapter 2.9 for more details). | Y |
| Date Submitted | The date when the enrollment was submitted. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Provider Type | The provider type of the provider. | String, 100 chars, non empty | Y |
| Provider Name | The provider name of the provider. | String, 100 chars, non empty | Y |
| Request Type | The type of the request. | String, 20 chars, non empty | Y |
| Status | The status of the enrollment. | String, 20 chars, non empty | Y |
| Risk Level | The risk level of the enrollment. | String, 20 chars, can be empty | N |

* The screen shot example of the enrollment status is like following:



## View Enrollments

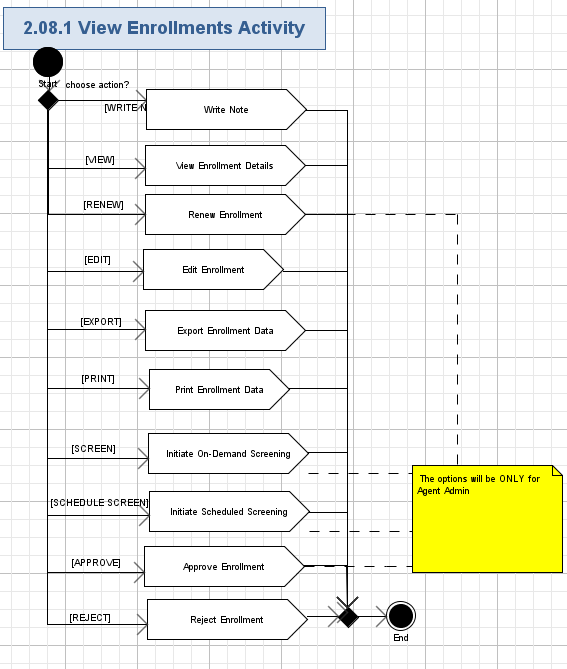
The application allows the users to view enrollments by status. The enrollment will contain provider type, provider name, status etc information. The user can also choose to filter the displayed enrollments. The user can choose to view the details of the enrollment (refer chapter 2.9 for details), edit the details of the enrollment (refer chapter 2.11 for details), edit the enrollment data (refer chapter 2.12 for details) or print the enrollment data (refer chapter 2.13 for details).

Conceptualization Reference: None.

Wireframes page: “Enrollment\_\_Service\_Admin\_.html”, “Enrollment\_\_Service\_Agent\_.html”.

* Pre-conditions: the user needed to view the enrollments in the application. The user has to be logged in to the application to perform this action.
* Post-conditions: the application shown the enrollments and the user can choose perform some actions on the displayed enrollments.

### View Enrollments Activity



#### Display Enrollments by Status

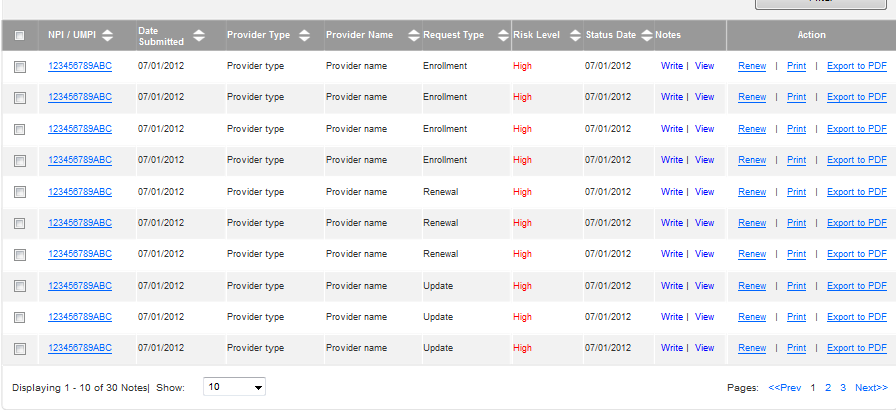
* The application will allow the user to choose the view enrollments by status (draft, pending, rejected, accepted and enrollment with notes additionally).
* The list of enrollments will be shown in a page-able table with the following columns:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| NPI/UMPI | The NPI/UMPI of the enrollment. | String, 20 chars, non empty.  It will be the hyperlink to view the details of the enrollment (refer chapter 2.9 for more details). | Y |
| Date Submitted | The date when the enrollment was submitted. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Provider Type | The provider type of the provider. | String, 100 chars, non empty | Y |
| Provider Name | The provider name of the provider. | String, 100 chars, non empty | Y |
| Request Type | The type of the request. | String, 20 chars, non empty | Y |
| Status | The status of the enrollment. | String, 20 chars, non empty | Y |
| Risk Level | The risk level of the enrollment. | String, 20 chars, can be empty | N |
| Date | The date when the action was performed. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Status Date | The date when the enrollment was approved/rejected | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| <Write> | The hyperlink to write notes. | Hyperlink.  See 2.8.2 for more details. | N |
| <Read> | The hyperlink to read notes. | Hyperlink. | N |
| <Edit> | The hyperlink to edit the enrollment. | Hyperlink.  The rejected/approved can not be edited. | N |
| <Renewal> | The hyperlink to renew the enrollment. | Hyperlink.  Only the rejected/approved can be renew. | N |
| <Print> | The hyperlink to print the enrollment data. | Hyperlink. | Y |
| <Export to PDF> | The hyperlink to export the enrollment data. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z> Enrollments* |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of records.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The user can navigate through the table pages by using standard navigation buttons, mentioned above.
* The user can sort the table with any column (ASC/DES).
* The screen shot example of the search results is like following:



#### Filter Enrollments

* The user can also choose to filter the displayed enrollments with the following criteria:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| NPI/UMPI | The NPI/UMPI of the enrollment. | String, 20 chars, can be empty. | N |
| Date Submitted From | The start of date range of the enrollment submitted. | String, max 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Date Submitted To | The end of date range of the enrollment submitted. | String, max 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Provider Type | The provider type of the provider. | String, max 100 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |
| Provider Name | The provider name of the provider. | String, 100 chars, can be empty. | N |
| Request Type | The type of the request. | String, max 20 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |
| Risk Level | The level of the risk of the enrollment. | String, max 20 chars, can be empty.  The user can choose the value from the dropdown list.  The default value is “All”. | N |

* The screen shot example is like following:



#### View Enrollment Details

* The user can choose to view the details of the enrollment by pressing the “NPI/UMPI” link on the chosen enrollment.
* Please refer chapter 2.9 for more details.

#### Edit Enrollment

* The user can choose to edit the details of the enrollment by pressing the “Edit” link on the chosen enrollment.
* Please refer chapter 2.11 for more details.

#### Export Enrollment Data

* The user can choose to export the data of the enrollment by pressing the “Export to PDF” link on the chosen enrollment.
* Please refer chapter 2.12 for more details.

#### Print Enrollment Data

* The user can choose to print the data of the enrollment by pressing the “Print” link on the chosen enrollment.
* Please refer chapter 2.13 for more details.

#### Initiate On-Demand Screening

* The agent admin user can choose to initiate On-Demand Screening on the enrollment by pressing the “Screen Now” button on the chosen enrollment.
* Please refer chapter 2.14 for more details.

#### Initiate Scheduled Screening

* The agent admin user can choose to initiate Scheduled Screening on the enrollment by pressing the “Schedule Screen” button on the chosen enrollment.
* Please refer chapter 2.14 for more details.

#### Approve Enrollment

* The agent admin user can choose to approve enrollment by pressing the “Approve” button on the chosen enrollment.
* Please refer chapter 2.16 for more details.

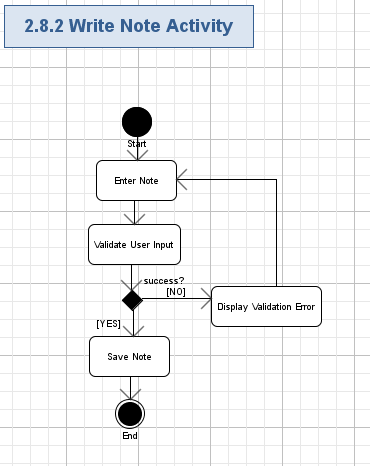
#### Reject Enrollment

* The agent admin user can choose to reject enrollment by pressing the “Reject” button on the chosen enrollment.
* Please refer chapter 2.17 for more details.

#### Renew Enrollment

* The agent admin user can choose to renew enrollment by pressing the “Renew” button on the chosen enrollment.
* Please refer chapter 2.10 for more details.

### Write Note Activity



#### Enter Note

* The user can enter the note for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Note | The note of the enrollment | String, 500 chars, non empty. | Y |

#### Validate User Input

* The user will press “Save” button to ask the application to save the note.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.8.2.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Save Note

* The application will save the note to the record.

## View Enrollment Details

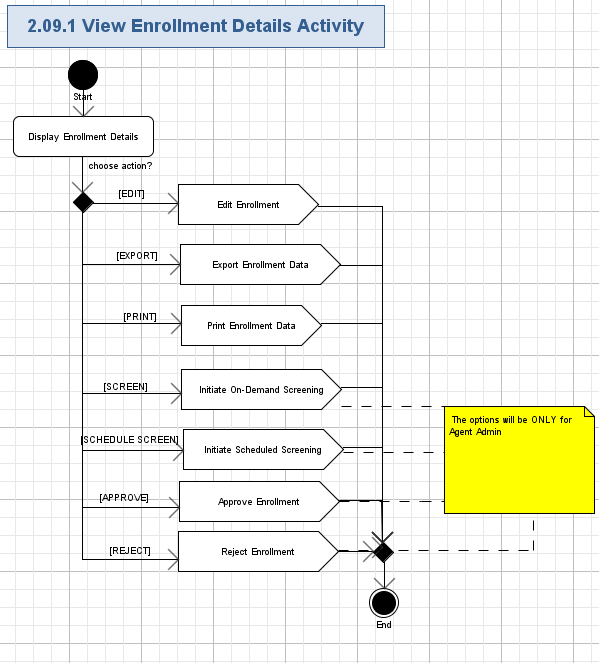
The application allows the users to view details of the enrollment. The enrollment will contain provider type, provider name, status etc information. The user can also choose to edit the details of the enrollment (refer chapter 2.11 for details), edit the enrollment data (refer chapter 2.12 for details) or print the enrollment data (refer chapter 2.13 for details).

Conceptualization Reference: 4.4.7 View Enrollment.

Wireframes page: “View\_Enrollment\_-\_Draft\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Draft\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Pending\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Pending\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Rejected\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Rejected\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Approved\_\_Service\_Admin\_.html” and “View\_Enrollment\_-\_ Approved \_\_Service\_Agent\_.html”

* Pre-conditions: the user needed to view the enrollment details in the application. The user has to be logged in to the application to perform this action.
* Post-conditions: the application shown the details of the chosen enrollment and the user can choose perform some actions on the displayed enrollment.

### View Enrollment Details Activity



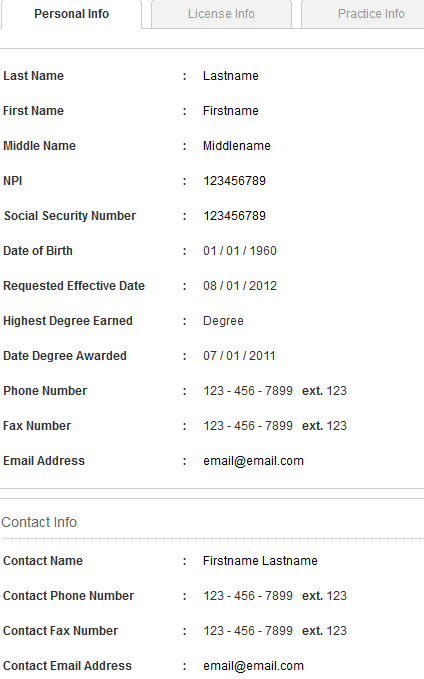
#### Display Enrollment Details

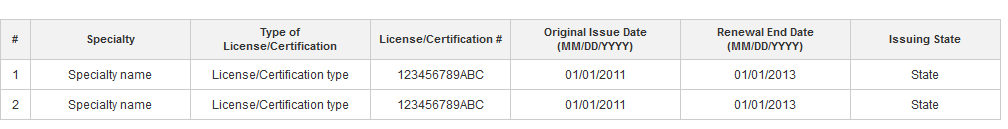
* The application will allow the user to view the details of the chosen enrollment.
* The application will display the details of the enrollment in several tabs:

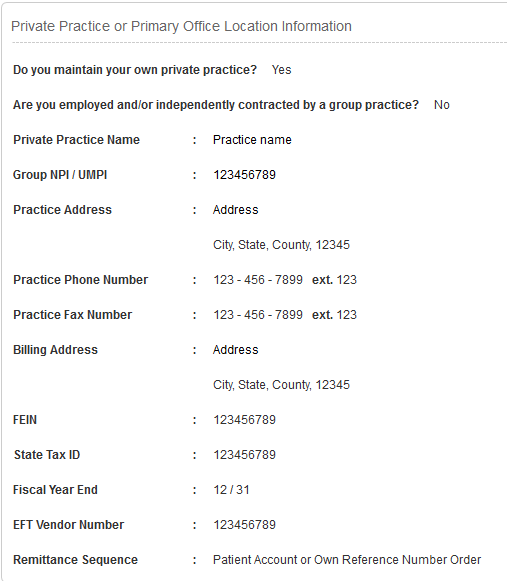
| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Request Type | The type of the request. | String, 20 chars, non empty | Y |
| Provider Type | The provider type of the provider. | String, 100 chars, non empty | Y |
| Status | The status of the enrollment. | String, 20 chars, non empty | Y |
| Created On | The date when the enrollment was created. | String, 20 chars, non empty.  Date/Time format: MM/DD/YYYY HH:MM:SS | Y |
| Status Date | The date when the enrollment was approved/rejected | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| **Personal Info** | | | |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| NPI | The NPI of the user. | String, 20 chars, can be empty | N |
| Social Security Number | The Social Security Number of the user. | String, 10 chars, non empty | Y |
| Date of Birth | The birth date of the user. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Requested Effective Date | The date when the request was effective. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Highest Degree Earned | The date when the highest degree earned. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Date Degree Awarded | The date when the degree awarded. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Phone Number | The phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Fax Number | The fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| Contact Name | The contact name of the user. | String, 100 chars, non empty | Y |
| Contact Phone Number | The contact phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Contact Fax Number | The contact fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Contact Email | The contact e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| **License Info (List of records)** | | | |
| # | The number of the license information record. | String, 100 chars, non empty | Y |
| Specialty | The Specialty name of the license information record. | String, 100 chars, non empty | Y |
| Type of  License/Certification | The type of the License. | String, 100 chars, non empty | Y |
| License/Certification # | The number of the license. | String, 100 chars, non empty | Y |
| Original Issue Date | The date when the license was original issued. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Renewal End Date | The date on which license needs to be renewed. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Issuing State | The state of the issuing. | String, 2 chars, non empty | Y |
| **Practice Info** | | | |
| Do you maintain your own private practice? | The question to ask if the user maintains her/his own private practice | Boolean, Yes/No. | Y |
| Are you employed and/or independently contracted by a group practice? | The question to ask if the user is employed. | Boolean, Yes/No. | Y |
| Private Practice Name | The private name of the practice. | String, 100 chars, non empty | Y |
| Group NPI / UMPI | The NPI/UMPI of the group of the practice. | String, 100 chars, non empty | Y |
| Practice Address | The address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| Practice Phone Number | The phone number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Practice Fax Number | The fax number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Billing Address | The billing address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| FEIN | The FEIN of the practice. | String, 20 chars, non empty | Y |
| State Tax ID | The state tax id of the practice. | String, 20 chars, non empty | Y |
| Fiscal Year End | The date of the fiscal year end. | String, 10 chars, non empty.  Date format: MM/DD | Y |
| EFT Vendor Number | The number of the EFT vendor. | String, 100 chars, non empty | Y |
| Remittance Sequence | The remittance sequence of the practice. | String, 100 chars, non empty | Y |
| **Payment Info** | | | |
| Fee | The fee that the user is required to pay to enroll. | Decimal (10, 2). | Y |
| Credit Card Type | The type of credit card. | String, 100 chars, non empty | Y |
| Name on Credit Card | The name on the credit card. | String, 100 chars, non empty | Y |
| Credit Card Number | The number of the credit card. | String, 100 chars, non empty | Y |
| Security Code | The security code of the credit card. | String, 100 chars, non empty | Y |
| Valid Until | The valid until date of the credit card. | String, 10 chars, non empty.  Date format: MM/YYYY | Y |
| **Provider Statement** | | | |
| Have you ever been convicted of a criminal offense related to involvement in any program underMedicare, Medicaid, Title XX, or Title XXI in Minnesota or any other state or jurisdiction since the inception of these programs? | Have you ever been convicted of a criminal offense related to involvement in any program underMedicare, Medicaid, Title XX, or Title XXI in Minnesota or any other state or jurisdiction since the inception of these programs? | Boolean, Yes/No. | Y |
| Have you had civil money penalties or assessments imposed under section 1128A of the Social Security Act? | Have you had civil money penalties or assessments imposed under section 1128A of the Social Security Act? | Boolean, Yes/No. | Y |
| Have you ever been excluded or terminated from participation in Medicare, Medicaid, Children’s Health Insurance Program (CHIP), or the Title XXI services program in Minnesota or any other state since the inception of these programs? | Have you ever been excluded or terminated from participation in Medicare, Medicaid, Children’s Health Insurance Program (CHIP), or the Title XXI services program in Minnesota or any other state since the inception of these programs? | Boolean, Yes/No. | Y |
| Provider Statement | The description of the statement. | String, 1024 chars, non empty | Y |
| Provider Name | The name of the provider. | String, 100 chars, non empty | Y |
| Provider Title | The title of the provider. | String, 100 chars, non empty | Y |
| Provider Signature: | The signature of the provider. | Image. | Y |
| Date | The date when the statement was made. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |

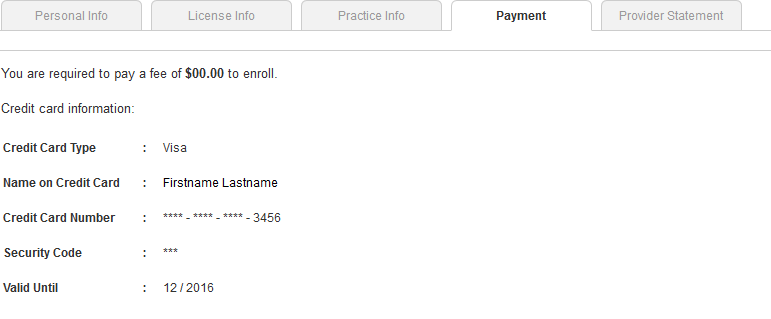
* The screen shot examples of the enrollment are like following:

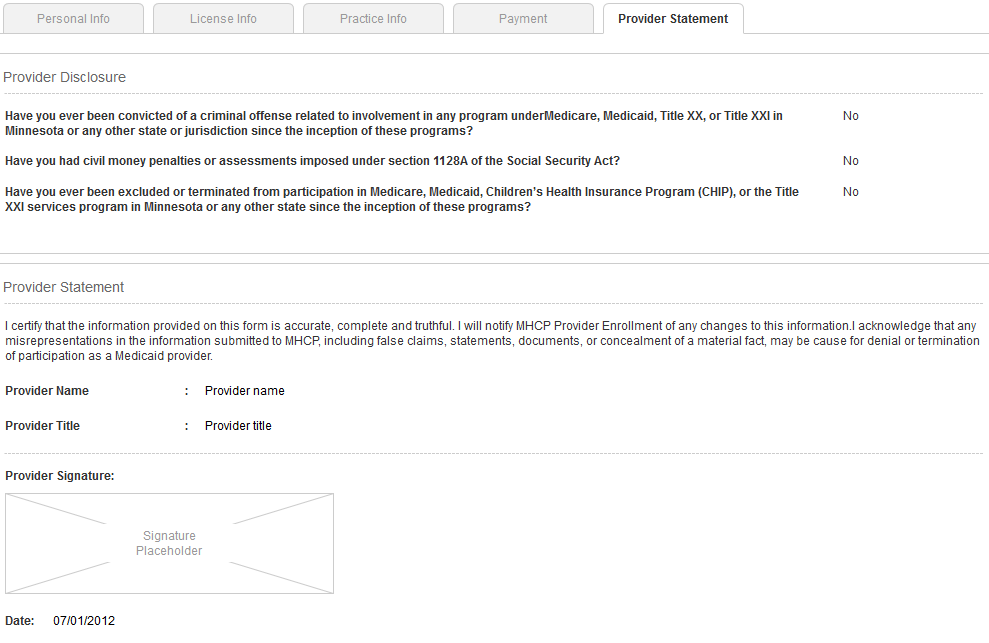












#### Edit Enrollment

* The user can choose to edit the details of the enrollment by pressing the “Edit Enrollment” button on the chosen enrollment.
* Please refer chapter 2.11 for more details.

#### Export Enrollment Data

* The user can choose to export the data of the enrollment by pressing the “Export to PDF” button on the chosen enrollment.
* Please refer chapter 2.12 for more details.

#### Print Enrollment Data

* The user can choose to print the data of the enrollment by pressing the “Print” button on the chosen enrollment.
* Please refer chapter 2.13 for more details.

#### Initiate On-Demand Screening

* The agent admin user can choose to initiate On-Demand Screening on the enrollment by pressing the “Screen Now” button on the chosen enrollment.
* Please refer chapter 2.14 for more details.

#### Initiate Scheduled Screening

* The agent admin user can choose to initiate Scheduled Screening on the enrollment by pressing the “Schedule Screen” button on the chosen enrollment.
* Please refer chapter 2.14 for more details.

#### Approve Enrollment

* The agent admin user can choose to approve enrollment by pressing the “Approve” button on the chosen enrollment.
* Please refer chapter 2.16 for more details.

#### Reject Enrollment

* The agent admin user can choose to reject enrollment by pressing the “Reject” button on the chosen enrollment.
* Please refer chapter 2.17 for more details.

## Create New Enrollment

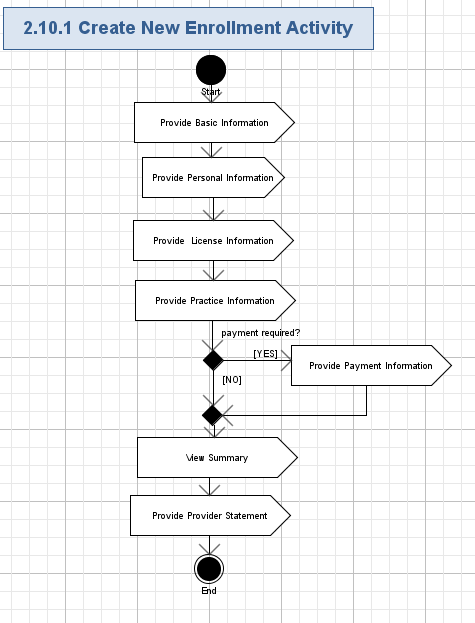
The application will allow users to create new enrollment in the application. The user will need to provide basic information, personal information, license information, practice information, payment information (optional) and provider statement. The user can view the summary the enrollment before providing the provider statement. **This use case will be used to create new enrollment or renewal enrollment.**

Conceptualization Reference: 4.4.5 Create Enrollment, 4.4.10 Create Renewal.

Wireframes page: “New\_Enrollment\_-\_No\_Payment\_\_Service\_Admin\_.html”, “New\_Enrollment\_-\_No\_Payment\_\_Service\_Agent\_.html”, “New\_Enrollment\_-\_Payment\_Required\_\_Service\_Admin\_.html”, “New\_Enrollment\_-\_Payment\_Required\_\_Service\_Agent\_.html”.

* Pre-conditions: the user needed to create new enrollment in the application. The user has to be logged in to the application to perform this action.
* Post-conditions: the enrollment was created in the application; or the user saved the enrollment as draft.

### Create New Enrollment Activity



#### Provide Basic Information

* The user will provide the basic information of the enrollment.
* Please refer chapter 2.10.2 for details.

#### Provide Personal Information

* The user will provide the personal information of the enrollment.
* Please refer chapter 2.10.3 for details.

#### Provide License Information

* The user will provide the license information of the enrollment.
* Please refer chapter 2.10.4 for details.

#### Provide Practice Information

* The user will provide the practice information of the enrollment.
* Please refer chapter 2.10.5 for details.

#### Provide Payment Information

* The user will provide the payment information of the enrollment.
* Please refer chapter 2.10.6 for details.

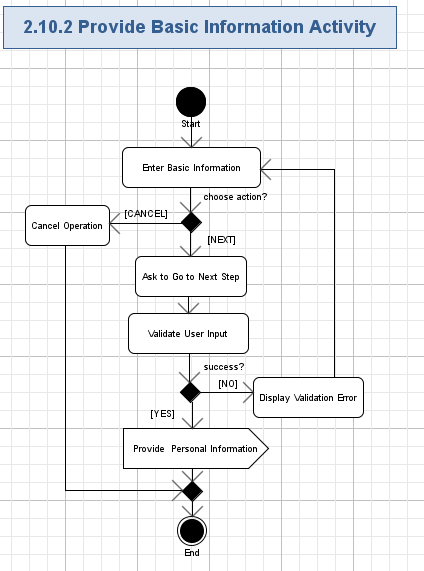
#### View Summary

* The user will be able to view summary before providing the provider statement.
* Please refer chapter 2.10.7 for details.

#### Provide Provider Statement

* The user will provide the statement to submit the enrollment.
* Please refer chapter 2.10.8 for details.

### Provide Basic Information Activity

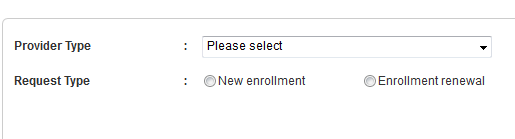


#### Enter Basic Information

* The user can enter the basic information for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Provider Type | The provider type of the provider. | String, 100 chars, non empty.  Dropdown List. | Y |
| Request Type | The type of the request. | String, 20 chars, non empty.  Radio buttons with values:  New Enrollment;  Enrollment Renewal. | Y |

* The screen shot example is like following:



#### Cancel Operation

* The user can press “Cancel” button to reject operation.
* No new enrollment will be created.

#### Validate User Input

* The user will press “Next” button to go to the next step of the enrollment creation process..
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.10.2.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

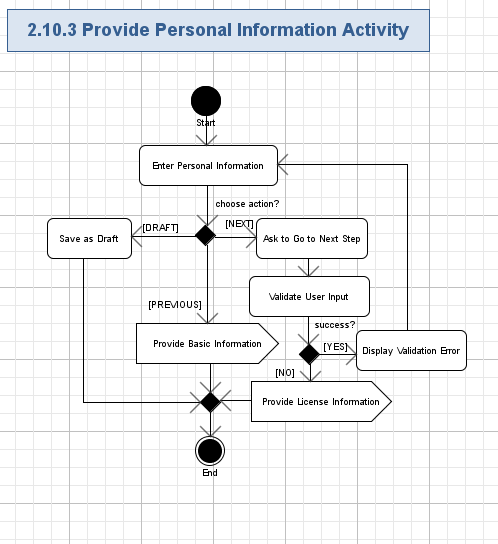
* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Provide Personal Information

* The user will got directed to the next step of the enrollment creation process.
* Please refer chapter 2.10.3 for details.

### Provide Personal Information Activity



#### Enter Personal Information

* The user can enter the personal information for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| **Personal Info** | | | |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| NPI | The NPI of the user. | String, 20 chars, can be empty | N |
| Social Security Number | The Social Security Number of the user. | String, 10 chars, non empty | Y |
| Date of Birth | The birth date of the user. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Requested Effective Date | The date when the request was effective. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Highest Degree Earned | The date when the highest degree earned. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Date Degree Awarded | The date when the degree awarded. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Phone Number | The phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Fax Number | The fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| <Same as above> | The checkbox to indicate if the following fields can be same as above.  Note: the user does not need to enter the same information. | Checkbox. | Y |
| Contact Name | The contact name of the user. | String, 100 chars, non empty | Y |
| Contact Phone Number | The contact phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Contact Fax Number | The contact fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Contact Email | The contact e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |

* The screen shot example is like following:



#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

#### Provide Basic Information

* The user will press “Previous” button to go to the previous step of the enrollment creation process.
* Please refer chapter 2.10.2 for details.

#### Validate User Input

* The user will press “Next” button to go to the next step of the enrollment creation process.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.10.3.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

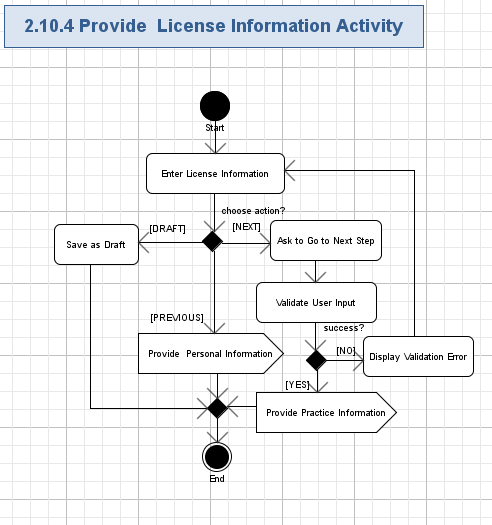
* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Provide License Information

* The user will got directed to the next step of the enrollment creation process.
* Please refer chapter 2.10.4 for details.

### Provide License Information Activity

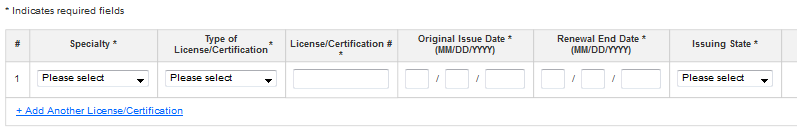


#### Enter License Information

* The user can enter the license information for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| **License Info (List of records)** | | | |
| # | The number of the license information record. | String, 100 chars, non empty.  No editable. | Y |
| Specialty | The Specialty name of the license information record. | String, 100 chars, non empty | Y |
| Type of  License/Certification | The type of the License. | String, 100 chars, non empty | Y |
| License/Certification # | The number of the license. | String, 100 chars, non empty | Y |
| Original Issue Date | The date when the license was original issued. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Renewal End Date | The date when the license was renewal. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Issuing State | The state of the issuing. | String, 100 chars, non empty | Y |
| <Add Another License/Certification> | The link to add more record. | Link. | Y |
| <Remove> | The link to remove the current. | Link. | Y |

* The screen shot example is like following:



#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

#### Provide Personal Information

* The user will press “Previous” button to go to the previous step of the enrollment creation process.
* Please refer chapter 2.10.3 for details.

#### Validate User Input

* The user will press “Next” button to go to the next step of the enrollment creation process.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.10.4.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

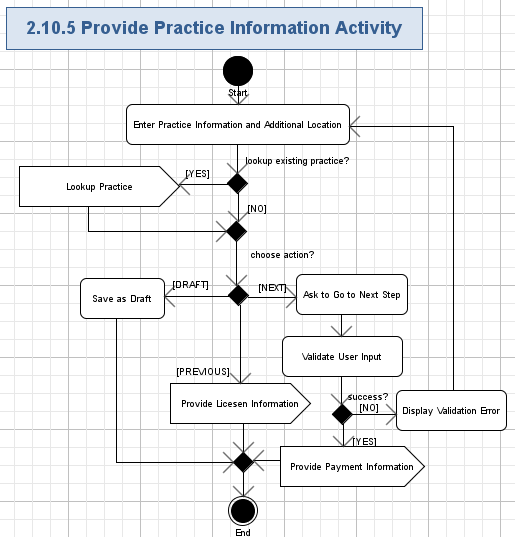
* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Provide Practice Information

* The user will got directed to the next step of the enrollment creation process.
* Please refer chapter 2.10.5 for details.

### Provide Practice Information Activity

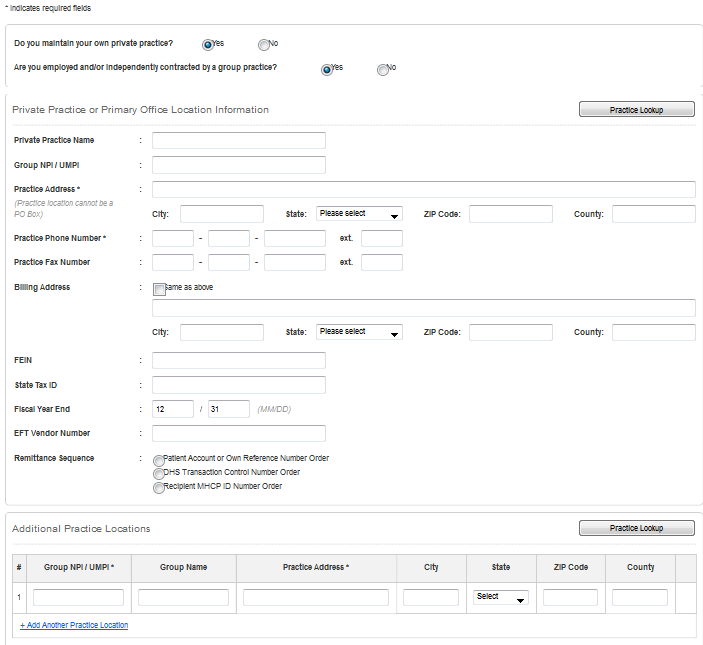


#### Enter Practice Information

* The user can enter the practice information for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| **Practice Info** | | | |
| Do you maintain your own private practice? | The question to ask if the user maintains her/his own private practice | Boolean, Yes/No. | Y |
| Are you employed and/or independently contracted by a group practice? | The question to ask if the user is employed. | Boolean, Yes/No. | Y |
| Private Practice Name | The private name of the practice. | String, 100 chars, non empty.  This field is required ONLY if the answer for the first question is “Yes” | Y |
| Primary Practice Name | The name of the primary practice. | String, 100 chars, non empty.  This field is required ONLY if the answer for the first question is “No” | Y |
| Group NPI / UMPI | The NPI/UMPI of the group of the practice. | String, 100 chars, non empty.  This field is required ONLY if the answer for the first question is “No” | Y |
| Practice Address | The address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| Practice Phone Number | The phone number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Practice Fax Number | The fax number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Billing Address | The billing address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information.  This field is required ONLY if the answer for the first question is “Yes” | Y |
| FEIN | The FEIN of the practice. | String, 100 chars, non empty  This field is required ONLY if the answer for the first question is “Yes” | Y |
| State Tax ID | The state tax id of the practice. | String, 100 chars, non empty  This field is required ONLY if the answer for the first question is “Yes” | Y |
| Fiscal Year End | The date of the fiscal year end. | String, 10 chars, non empty.  Date format: MM/DD  This field is required ONLY if the answer for the first question is “Yes” | Y |
| EFT Vendor Number | The number of the EFT vendor. | String, 100 chars, non empty  This field is required ONLY if the answer for the first question is “Yes” | Y |
| Remittance Sequence | The remittance sequence of the practice. | String, 100 chars, non empty  This field is required ONLY if the answer for the first question is “Yes” | Y |
| Reimbursement Address | The Reimbursement Address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information.  This field is required ONLY if the answer for the first question is “No” | Y |
| **Additional Locations** | | | |
| Group NPI / UMPI | The NPI/UMPI of the group of the practice. | String, 100 chars, non empty. | Y |
| Group Name | The name of the group. | String, 100 chars, non empty. | Y |
| Practice Address | The address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |

* The screen shot example is like following:



#### Lookup Practice

* The user will press “Practice Look” button to lookup the existing practices to avoid entering the same data again and again.
* Please refer chapter 2.10.9 for details.

#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

#### Provide Practice Information

* The user will press “Previous” button to go to the previous step of the enrollment creation process.
* Please refer chapter 2.10.4 for details.

#### Validate User Input

* The user will press “Next” button to go to the next step of the enrollment creation process.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.10.5.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

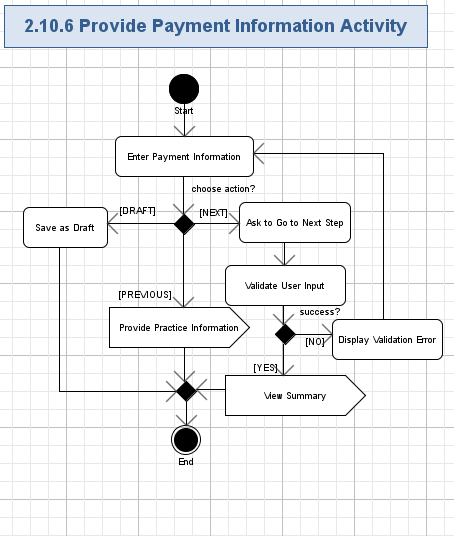
* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Provide Payment Information

* The user will got directed to the next step of the enrollment creation process.
* Please refer chapter 2.10.6 for details.

### Provide Payment Information Activity

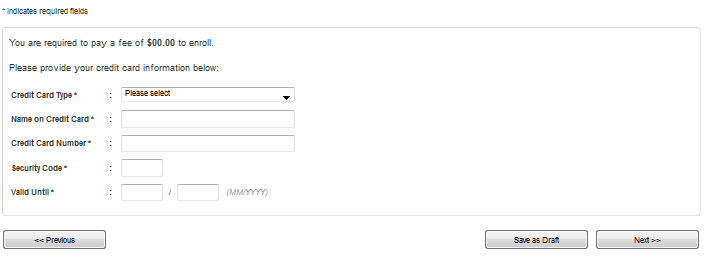


#### Enter Payment Information

* The user can enter the payment information for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| **Payment Info** | | | |
| Credit Card Type | The type of credit card. | String, 20 chars, non empty.  Dropdown list. | Y |
| Name on Credit Card | The name on the credit card. | String, 100 chars, non empty | Y |
| Credit Card Number | The number of the credit card. | String, 20 chars, non empty | Y |
| Security Code | The security code of the credit card. | String, 3 chars, non empty | Y |
| Valid Until | The valid until date of the credit card. | String, 10 chars, non empty.  Date format: MM/YYYY | Y |

* The screen shot example is like following:



#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

#### Provide Practice Information

* The user will press “Previous” button to go to the previous step of the enrollment creation process.
* Please refer chapter 2.10.5 for details.

#### Validate User Input

* The user will press “Next” button to go to the next step of the enrollment creation process.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.10.6.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

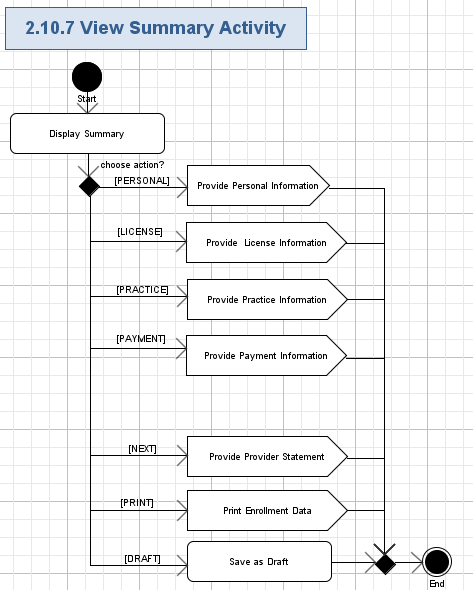
* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### View Summary

* The user will got directed to the next step of the enrollment creation process.
* Please refer chapter 2.10.7 for details.

### View Summary Activity

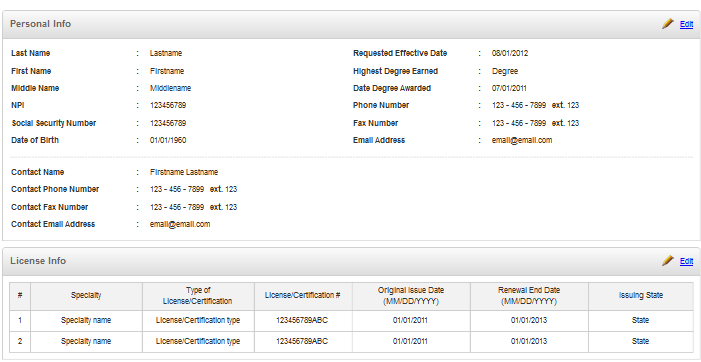


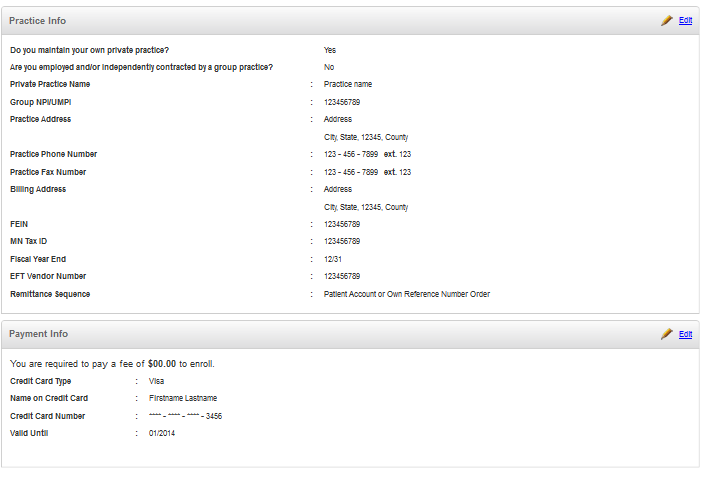
#### Display Summary

* The application will display the summary information of the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| **Personal Info** | | | |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| NPI | The NPI of the user. | String, 20 chars, can be empty | N |
| Social Security Number | The Social Security Number of the user. | String, 10 chars, non empty | Y |
| Date of Birth | The birth date of the user. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Requested Effective Date | The date when the request was effective. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Highest Degree Earned | The date when the highest degree earned. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Date Degree Awarded | The date when the degree awarded. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Phone Number | The phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Fax Number | The fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| Contact Name | The contact name of the user. | String, 100 chars, non empty | Y |
| Contact Phone Number | The contact phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Contact Fax Number | The contact fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Contact Email | The contact e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| **License Info (List of records)** | | | |
| # | The number of the license information record. | String, 100 chars, non empty | Y |
| Specialty | The Specialty name of the license information record. | String, 100 chars, non empty | Y |
| Type of  License/Certification | The type of the License. | String, 100 chars, non empty | Y |
| License/Certification # | The number of the license. | String, 100 chars, non empty | Y |
| Original Issue Date | The date when the license was original issued. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Renewal End Date | The date when the license was renewal. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Issuing State | The state of the issuing. | String, 100 chars, non empty | Y |
| **Practice Info** | | | |
| Do you maintain your own private practice? | The question to ask if the user maintains her/his own private practice | Boolean, Yes/No. | Y |
| Are you employed and/or independently contracted by a group practice? | The question to ask if the user is employed. | Boolean, Yes/No. | Y |
| Private Practice Name | The private name of the practice. | String, 100 chars, non empty | Y |
| Group NPI / UMPI | The NPI/UMPI of the group of the practice. | String, 100 chars, non empty | Y |
| Practice Address | The address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| Practice Phone Number | The phone number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Practice Fax Number | The fax number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Billing Address | The billing address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| FEIN | The FEIN of the practice. | String, 100 chars, non empty | Y |
| State Tax ID | The state tax id of the practice. | String, 100 chars, non empty | Y |
| Fiscal Year End | The date of the fiscal year end. | String, 10 chars, non empty.  Date format: MM/DD | Y |
| EFT Vendor Number | The number of the EFT vendor. | String, 100 chars, non empty | Y |
| Remittance Sequence | The remittance sequence of the practice. | String, 100 chars, non empty | Y |
| **Payment Info** | | | |
| Fee | The fee that the user is required to pay to enroll. | Decimal (10, 2). | Y |
| Credit Card Type | The type of credit card. | String, 100 chars, non empty | Y |
| Name on Credit Card | The name on the credit card. | String, 100 chars, non empty | Y |
| Credit Card Number | The number of the credit card. | String, 100 chars, non empty | Y |
| Security Code | The security code of the credit card. | String, 100 chars, non empty | Y |
| Valid Until | The valid until date of the credit card. | String, 10 chars, non empty.  Date format: MM/YYYY | Y |

* The screen shot example is like following:





#### Provide Personal Information

* The user will provide the personal information of the enrollment by pressing the “edit” link on the related panel.
* Please refer chapter 2.10.3 for details.

#### Provide License Information

* The user will provide the license information of the enrollment by pressing the “edit” link on the related panel.
* Please refer chapter 2.10.4 for details.

#### Provide Practice Information

* The user will provide the practice information of the enrollment by pressing the “edit” link on the related panel.
* Please refer chapter 2.10.5 for details.

#### Provide Payment Information

* The user will provide the payment information of the enrollment by pressing the “edit” link on the related panel.
* The user will press “Previous” button to go to the previous step of the enrollment creation process.
* Please refer chapter 2.10.6 for details.

#### Provide Provider Statement

* The user will got directed to the next step of the enrollment creation process.
* Please refer chapter 2.10.8 for details.

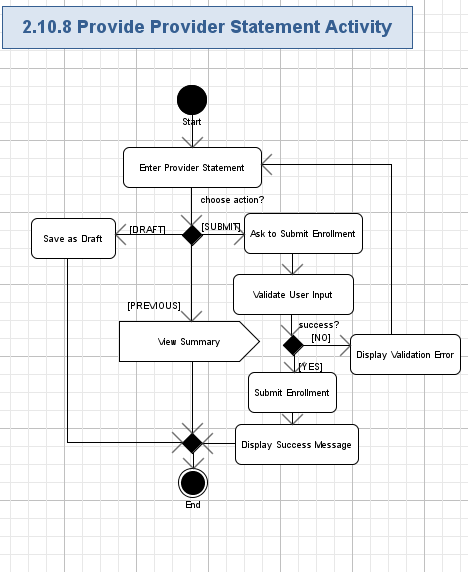
#### Print Enrollment Data

* The user can choose to print the data of the enrollment by pressing the “Print” button on the chosen enrollment.
* Please refer chapter 2.13 for more details.

#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

### Provide Provider Statement Activity

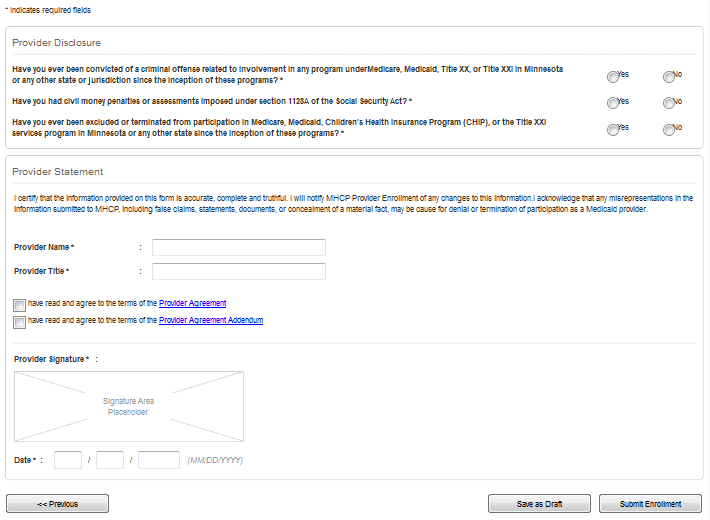


#### Enter Provider Statement

* The user can enter the statement information for the enrollment:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| **Provider Statement** | | | |
| Have you ever been convicted of a criminal offense related to involvement in any program underMedicare, Medicaid, Title XX, or Title XXI in Minnesota or any other state or jurisdiction since the inception of these programs? | Have you ever been convicted of a criminal offense related to involvement in any program underMedicare, Medicaid, Title XX, or Title XXI in Minnesota or any other state or jurisdiction since the inception of these programs? | Boolean, Yes/No.  Radio button. | Y |
| Have you had civil money penalties or assessments imposed under section 1128A of the Social Security Act? | Have you had civil money penalties or assessments imposed under section 1128A of the Social Security Act? | Boolean, Yes/No.  Radio button. | Y |
| Have you ever been excluded or terminated from participation in Medicare, Medicaid, Children’s Health Insurance Program (CHIP), or the Title XXI services program in Minnesota or any other state since the inception of these programs? | Have you ever been excluded or terminated from participation in Medicare, Medicaid, Children’s Health Insurance Program (CHIP), or the Title XXI services program in Minnesota or any other state since the inception of these programs? | Boolean, Yes/No.  Radio button. | Y |
| Provider Name | The name of the provider. | String, 100 chars, non empty | Y |
| Provider Title | The title of the provider. | String, 100 chars, non empty | Y |
| Provider Signature: | The signature of the provider. | Image. Max 2 M. | Y |
| Date | The date when the statement was made. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |

* The screen shot example is like following:



#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

#### View Summary

* The user will press “Previous” button to go to the previous step of the enrollment creation process.
* Please refer chapter 2.10.7 for details.

#### Validate User Input

* The user will press “Submit Enrollment” button to ask the application to submit the enrollment.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.10.8.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Submit Enrollment

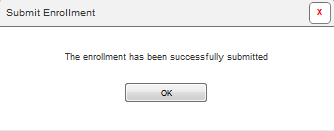
* The enrollment will be submitted in the application.

#### Display Success Message

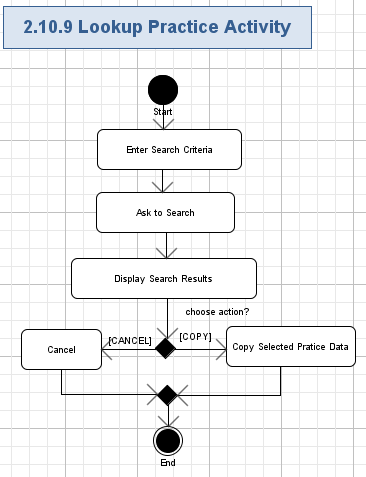
* The application will display success message within the popup window on the enrollment creation like this:

|  |
| --- |
| *Submit Enrollment*  *The enrollment has been successfully submitted.*  *<Ok>* |

* The user can close the popup window by pressing the “Ok” button.
* The screen shot example of the success message is like following:



### Lookup Practice Activity



#### Enter Search Criteria

* The application will show following criteria for the user to enter for searching practice data:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Practice Name | The name of the practice. | String, max 100 chars, can be empty | N |
| NPI/UMPI | The NPI/UMPI of the practice. | String, max 100 chars, can be empty | N |
| City | The city of the practice address. | String, max 100 chars, can be empty | N |
| State | The city of the practice address. | String, max 100 chars, can be empty | N |
| Zip | The zip code of the practice address. | String, max 100 chars, can be empty | N |

#### Ask to Search

* The user can ask the application to perform search practices by pressing the “Search” button.

#### Display Search Results

* The list of results will be shown in a page-able table with the following columns:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Practice Name | The name of the practice. | String, max 100 chars, non empty | Y |
| NPI/UMPI | The NPI/UMPI of the practice. | String, max 100 chars, non empty | Y |
| City | The city of the practice address. | String, max 100 chars, non empty | Y |
| State | The city of the practice address. | String, max 100 chars, non empty | Y |
| Zip | The zip code of the practice address. | String, max 100 chars, non empty | Y |

#### Cancel

* The user can cancel the results by pressing the “Cancel” button.

#### Copy Selected Practice Data

* The user can also choose the copy the selected practice data (chosen by radio button).

## Edit Enrollment

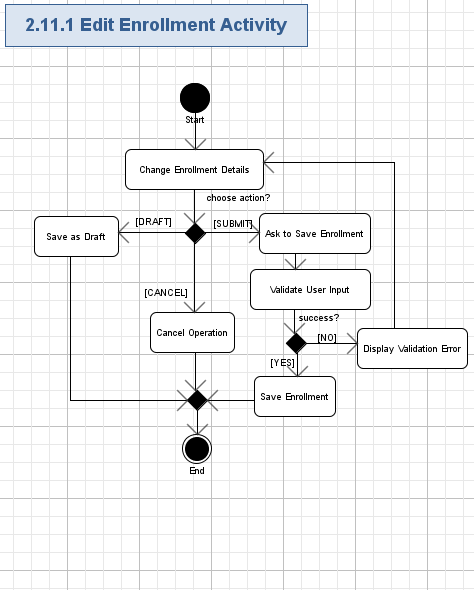
The application will allow users to edit the details of the enrollment in the application. The user can provide basic information, personal information, license information, practice information, payment information or provider statement on updating the existing enrollment.

Conceptualization Reference: 4.4.6 Edit Enrollment.

Wireframes page: “Edit\_Enrollment\_\_Service\_Admin\_.html”, “Edit\_Enrollment\_\_Service\_Agent\_.html”.

* Pre-conditions: the user needed to update the existing enrollment in the application. The user has to be logged in to the application to perform this action.
* Post-conditions: the enrollment was updated in the application; or the user saved the enrollment as draft.

### Edit Enrollment Activity

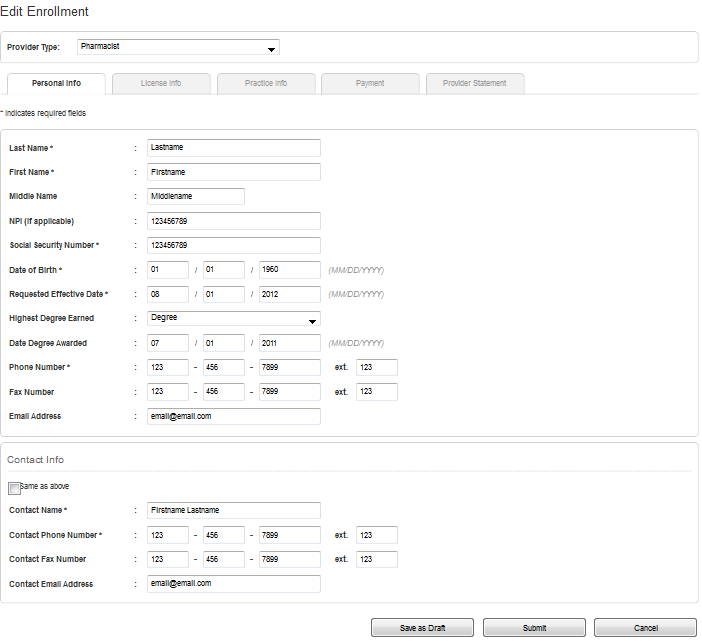


#### Change Enrollment Details

* The user can change the details of the existing enrollment for the following the fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Provider Type | The provider type of the provider. | String, 100 chars, non empty | Y |
| **Personal Info** | | | |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| NPI | The NPI of the user. | String, 20 chars, can be empty | N |
| Social Security Number | The Social Security Number of the user. | String, 10 chars, non empty | Y |
| Date of Birth | The birth date of the user. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Requested Effective Date | The date when the request was effective. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Highest Degree Earned | The date when the highest degree earned. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Date Degree Awarded | The date when the degree awarded. | String, 10 chars, can be empty.  Date format: MM/DD/YYYY | N |
| Phone Number | The phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Fax Number | The fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| Contact Name | The contact name of the user. | String, 100 chars, non empty | Y |
| Contact Phone Number | The contact phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Contact Fax Number | The contact fax number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Contact Email | The contact e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| **License Info (List of records)** | | | |
| # | The number of the license information record. | String, 100 chars, non empty | Y |
| Specialty | The Specialty name of the license information record. | String, 100 chars, non empty | Y |
| Type of  License/Certification | The type of the License. | String, 100 chars, non empty | Y |
| License/Certification # | The number of the license. | String, 100 chars, non empty | Y |
| Original Issue Date | The date when the license was original issued. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Renewal End Date | The date when the license was renewal. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |
| Issuing State | The state of the issuing. | String, 100 chars, non empty | Y |
| **Practice Info** | | | |
| Do you maintain your own private practice? | The question to ask if the user maintains her/his own private practice | Boolean, Yes/No. | Y |
| Are you employed and/or independently contracted by a group practice? | The question to ask if the user is employed. | Boolean, Yes/No. | Y |
| Private Practice Name | The private name of the practice. | String, 100 chars, non empty | Y |
| Group NPI / UMPI | The NPI/UMPI of the group of the practice. | String, 100 chars, non empty | Y |
| Practice Address | The address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| Practice Phone Number | The phone number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Practice Fax Number | The fax number of the practice. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Billing Address | The billing address of the practice. | String, 400 chars, non empty.  This field includes address, state, city, country and zip code information. | Y |
| FEIN | The FEIN of the practice. | String, 100 chars, non empty | Y |
| State Tax ID | The state tax id of the practice. | String, 100 chars, non empty | Y |
| Fiscal Year End | The date of the fiscal year end. | String, 10 chars, non empty.  Date format: MM/DD | Y |
| EFT Vendor Number | The number of the EFT vendor. | String, 100 chars, non empty | Y |
| Remittance Sequence | The remittance sequence of the practice. | String, 100 chars, non empty | Y |
| **Payment Info** | | | |
| Fee | The fee that the user is required to pay to enroll. | Decimal (10, 2). | Y |
| Credit Card Type | The type of credit card. | String, 100 chars, non empty | Y |
| Name on Credit Card | The name on the credit card. | String, 100 chars, non empty | Y |
| Credit Card Number | The number of the credit card. | String, 100 chars, non empty | Y |
| Security Code | The security code of the credit card. | String, 100 chars, non empty | Y |
| Valid Until | The valid until date of the credit card. | String, 10 chars, non empty.  Date format: MM/YYYY | Y |
| **Provider Statement** | | | |
| Have you ever been convicted of a criminal offense related to involvement in any program underMedicare, Medicaid, Title XX, or Title XXI in Minnesota or any other state or jurisdiction since the inception of these programs? | Have you ever been convicted of a criminal offense related to involvement in any program underMedicare, Medicaid, Title XX, or Title XXI in Minnesota or any other state or jurisdiction since the inception of these programs? | Boolean, Yes/No. | Y |
| Have you had civil money penalties or assessments imposed under section 1128A of the Social Security Act? | Have you had civil money penalties or assessments imposed under section 1128A of the Social Security Act? | Boolean, Yes/No. | Y |
| Have you ever been excluded or terminated from participation in Medicare, Medicaid, Children’s Health Insurance Program (CHIP), or the Title XXI services program in Minnesota or any other state since the inception of these programs? | Have you ever been excluded or terminated from participation in Medicare, Medicaid, Children’s Health Insurance Program (CHIP), or the Title XXI services program in Minnesota or any other state since the inception of these programs? | Boolean, Yes/No. | Y |
| Provider Statement | The description of the statement. | String, 1024 chars, non empty | Y |
| Provider Name | The name of the provider. | String, 100 chars, non empty | Y |
| Provider Title | The title of the provider. | String, 100 chars, non empty | Y |
| Provider Signature: | The signature of the provider. | Image. | Y |
| Date | The date when the statement was made. | String, 10 chars, non empty.  Date format: MM/DD/YYYY | Y |

* The screen shot example is like following:



#### Save as Draft

* The user can press “Save as Draft” button to save the enrollment as draft status.

#### Cancel Operation

* The user will press “Cancel” button to cancel the changes to the existing enrollment.
* The enrollment will not be changed.

#### Ask to Save Enrollment

* The user will press “Submit” button to ask the application to submit the changes to the enrollment.

#### Validate User Input

* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.11.1.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Save Enrollment

* The enrollment will be submitted in the application.

## Export Enrollment Data

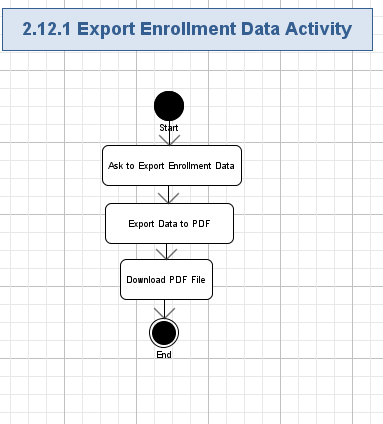
The user will be able to export the enrollment data being viewed to PDF format. The PDF file formatting has to be professional and have the layout similar to the displayed on the web-page.

Conceptualization Reference: 4.4.9 Export Enrollment Data

Wireframes page: “View\_Enrollment\_-\_Draft\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Draft\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Pending\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Pending\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Rejected\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Rejected\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Approved\_\_Service\_Admin\_.html” and “View\_Enrollment\_-\_ Approved \_\_Service\_Agent\_.html”

* Pre-conditions: the user was viewing the displayed enrollment data. The user has to be logged in to perform this action.
* Post-conditions: the enrollment was exported to PDF format.

### Export Enrollment Data Activity



#### Ask to Export Enrollment Data

* The user can press “Export to PDF” button to export currently displayed data to the PDF file.

#### Export Data to PDF

* All the displayed data will be exported to PDF file in the tabular format.
* PDF file formatting has to be professional and have the layout similar to the displayed on the web-page.

#### Download PDF File

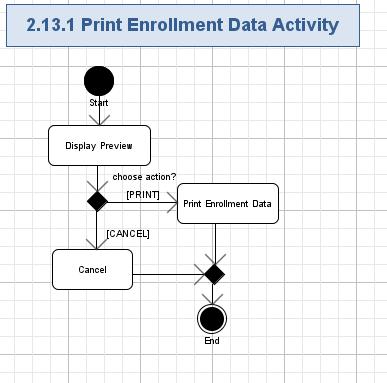
* The user can download exported PDF file by his/her web-browser to the local file system and open it.

## Print Enrollment Data

The application will allow the user to print the enrollment data being viewed. The displayed enrollment data will be printed as printer-friendly version.

* Conceptualization Reference: 4.4.16 Print Form
* Wireframes page: “View\_Enrollment\_-\_Draft\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Draft\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Pending\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Pending\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Rejected\_\_Service\_Admin\_.html”, “View\_Enrollment\_-\_Rejected\_\_Service\_Agent\_.html”, “View\_Enrollment\_-\_Approved\_\_Service\_Admin\_.html” and “View\_Enrollment\_-\_ Approved \_\_Service\_Agent\_.html”
* Pre-conditions: the user was viewing the displayed enrollment data. The user has to be logged in to the application to perform this action.
* Post-conditions: the displayed enrollment data printed using the print feature of the web browser; or the user cancel the printing.

### Print Enrollment Data Activity



#### Display Preview

* The application will display the preview version of the enrollment data.

#### Cancel

* The user can press the “Cancel” button to reject the printing.

#### Print Enrollment Data

* The user can print the enrollment data by clicking the “Print” button.
* The enrollment data will be printed as printer-friendly version.
* The System will use the print capabilities provided by the web browser.

## Initiate On-Demand Screening

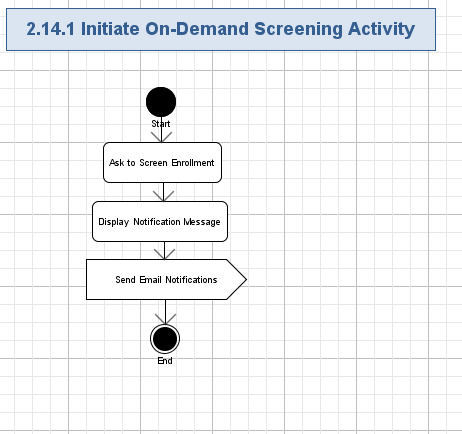
The application will allow user to initiate on-demand screening from the page when viewing the non-screened enrollments. The screening will be the auto-validation that will be performed on the enrollment data. The agent admin can approve/reject the screened enrollments. The system will send notification on the screening results to the service admin.

Conceptualization Reference: 4.4.20 Initiate On-Demand Screening

Wireframes page: “View\_Enrollment\_-\_Not\_Screened\_\_Service\_Admin\_.html”

* Pre-conditions: the service admin pressed the “Screen Now” button when viewing the non-screen enrollment. The user has to be logged in to the application to perform this action.
* Post-conditions: the non-screened enrollment was screened.

### Initiate On-Demand Screening Activity



#### Ask to Screen Enrollment

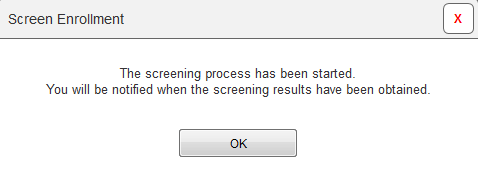
* The user can press “Screen Now” button to ask the application to start the screening process on the enrollment.

#### Display Notification Message

* The application will display notification message within the popup window on the enrollment screening like this:

|  |
| --- |
| *Screen Enrollment*  *The screening process has been started.*  *You will be notified when the screening results have been obtained.*  *<Ok>* |

* The user can close the popup window by pressing the “Ok” button.
* The screen shot example of the notification message is like following:



#### Send Email Notifications

* The system will send email notification to the service agent on the screening results.
* Please refer chapter 2.26 for more details.

## Initiate Scheduled Screening

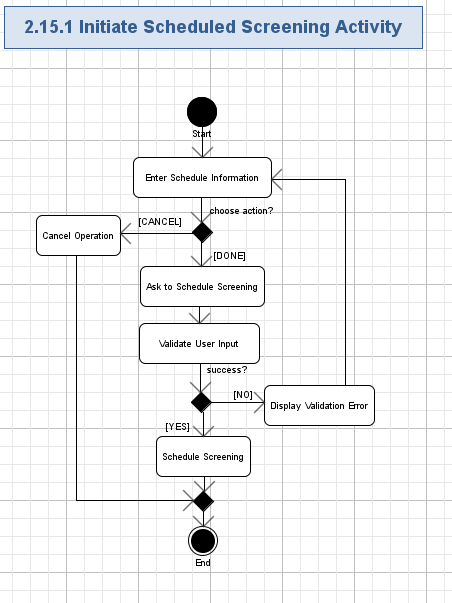
The application will allow user to initiate scheduled screening from the page when viewing the non-screened enrollments. The screening will be the auto-validation that will be performed on the enrollment data. The agent admin can approve/reject the screened enrollments. The system will start the screening process on the scheduled date.

Conceptualization Reference: 4.4.21 Initiate Scheduled Screening

Wireframes page: “View\_Enrollment\_-\_Not\_Screened\_\_Service\_Admin\_.html”

* Pre-conditions: the service admin pressed the “Schedule Screening” button when viewing the non-screen enrollment. The user has to be logged in to the application to perform this action.
* Post-conditions: the non-screened enrollment was scheduled to screen, or the user cancelled the operation.

### Initiate Scheduled Screening Activity

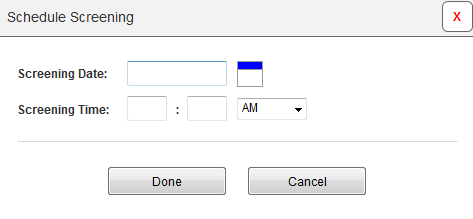


#### Enter Schedule Information

* The user can enter the schedule information for the screening:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Screening Date | The date when the screening will be started. | String, 10 chars, non empty.  Date format: MM/DD/YYYY.  The user can choose the value from the calendar control. | Y |
| Screening Time | The time when the screening will be started. | String, 10 chars, non empty.  Date format: HH:MM AM/PM | Y |

* The screen shot example is like following:



#### Cancel Operation

* The user can press “Cancel” button to reject operation.
* No new enrollment will be scheduled to screen.

#### Ask to Schedule Screening

* The user will press “Done” button to ask the application to schedule the screening.

#### Validate User Input

* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.15.1.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Schedule Screening

* The enrollment will be scheduled to screen on the scheduled date/time.

## Approve Enrollment

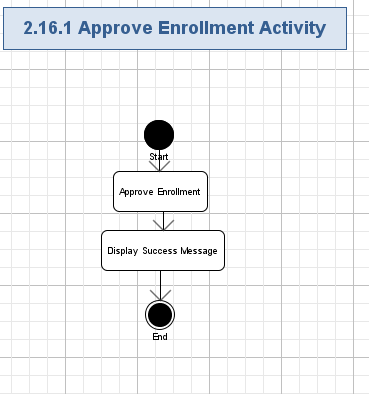
The application will allow user to approve the screened enrollment from the page when viewing the screened enrollments. The status of the enrollment will be changed to be “Approved”. The enrollment that was approved will be stored in the database.

Conceptualization Reference: 4.4.18 Set Enrollment Status

Wireframes page: “View\_Enrollment\_-\_Screened\_\_Service\_Admin\_.html”

* Pre-conditions: the service admin pressed the “Approve” button when viewing the screen enrollment. The user has to be logged in to the application to perform this action.
* Post-conditions: the screened enrollment was approved.

### Approve Enrollment Activity



#### Approve Enrollment

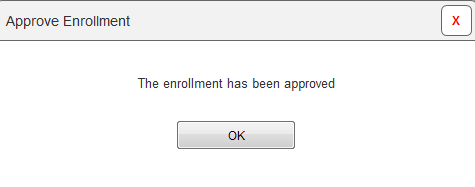
* The application will change the status of the enrollment from “pending” to “approved”.

#### Display Success Message

* The application will display success message within the popup window on the enrollment approval like this:

|  |
| --- |
| *Approve Enrollment*  *The enrollment has been approved.*  *<Ok>* |

* The user can close the popup window by pressing the “Ok” button.
* The screen shot example of the notification message is like following:



## Reject Enrollment

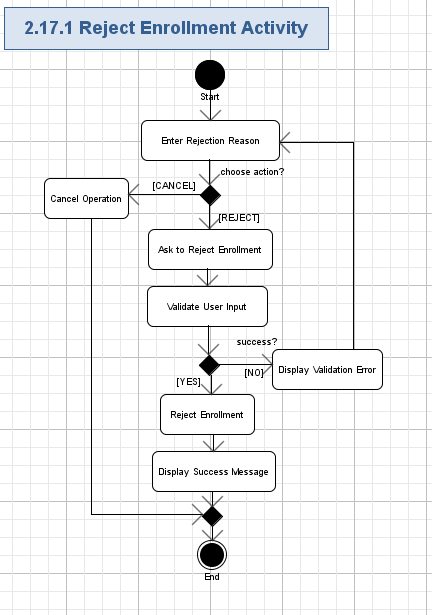
The application will allow user to reject the screened enrollment from the page when viewing the screened enrollments. The status of the enrollment will be changed to be “Rejected”. The enrollment that was rejected will be stored in the database.

Conceptualization Reference: 4.4.18 Set Enrollment Status

Wireframes page: “View\_Enrollment\_-\_Screened\_\_Service\_Admin\_.html”

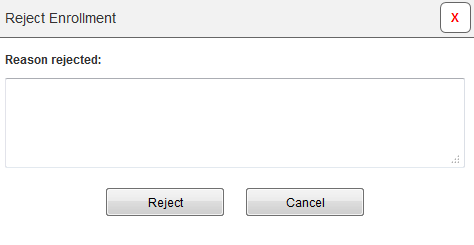
* Pre-conditions: the service admin pressed the “Rejected” button when viewing the screen enrollment. The user has to be logged in to the application to perform this action.
* Post-conditions: the screened enrollment was rejected.

### Reject Enrollment Activity



#### Enter Rejection Reason

* The user will need to enter the reason for the rejection (String, max 1024 chars, non empty).
* The screen shot example is like following:



#### Cancel Operation

* The user can cancel the action by pressing “Cancel” button.
* No enrollment will be rejected in this case.

#### Ask to Reject Enrollment

* The user can press “Reject” button to reject the enrollment.

#### Validate User Input

* The application will validate user input according to the rule in 2.17.1.1.

#### Display Validation Error

* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> cannot exceed 1024 chars.* |

#### Reject Enrollment

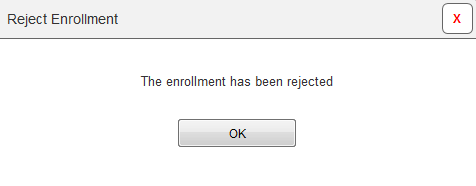
* The application will change the status of the enrollment from “pending” to “rejected”.

#### Display Success Message

* The application will display success message within the popup window on the enrollment rejection like this:

|  |
| --- |
| *Reject Enrollment*  *The enrollment has been rejected.*  *<Ok>* |

* The user can close the popup window by pressing the “Ok” button.
* The screen shot example of the notification message is like following:



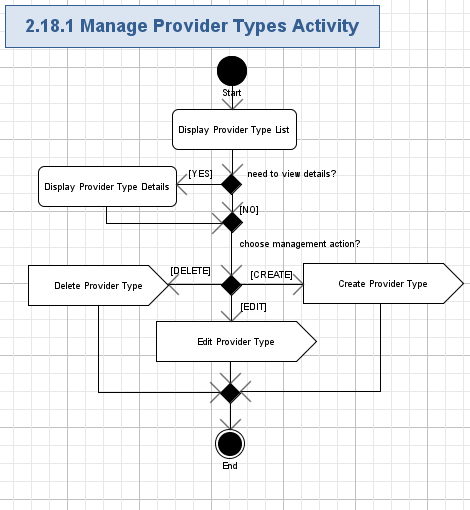
## Manage Provider Types

The user can manage the provider types on the GUI page. It will be possible to view existing provider types, create new provider types, update provider types, or remove them. There will be many types of the providers. The provider type means what services the provider can provide.  
Conceptualization Reference: None

Wireframes Page: “Functions\_\_Service\_Admin\_.html”

* Pre-conditions: the user pressed “Function” menu in the top navigation menu and then selected “Provider Types”. The user has to be logged in to the application to perform this action.
* Post-conditions: the user successfully managed the provider types.

### Manage Provider Types Activity



#### Display Provider Type List

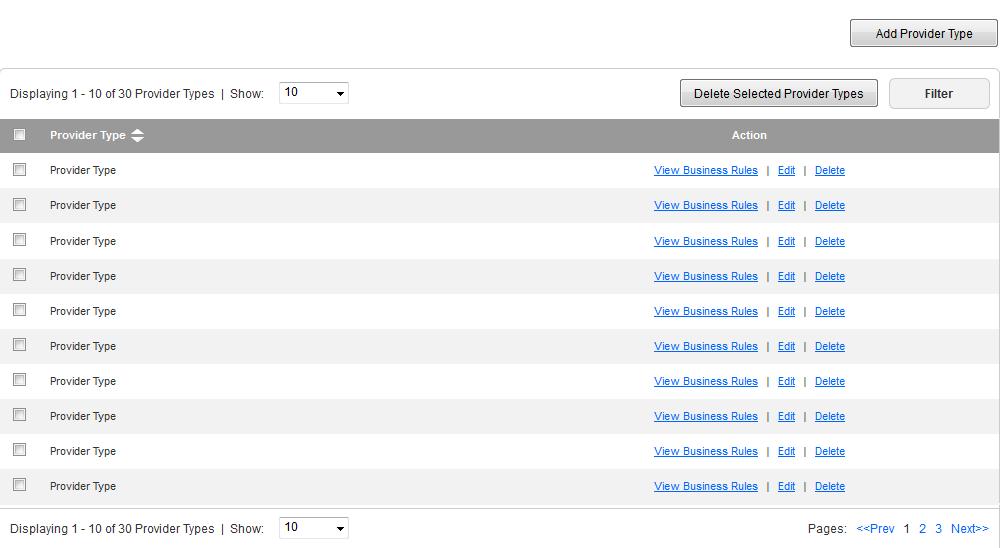
* The application will display the page able table with the list of all the existing provider types in the system.
* Each row in the table will correspond to the related provider type and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Name | The name of the Provider Type | String, max 100 chars, non empty. | Y |
| <View Business Rule> | The hyperlink to view the business rule. | Hyperlink. | Y |
| <Edit> | The hyperlink to edit the provider type. | Hyperlink. | Y |
| <Delete> | The hyperlink to delete the provider type. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 5, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z> Provider Types* |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of entities in the system.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The Application will allow the user to sort the table by “Provider Type” column (DES/ASC).
* The screen example is like following:

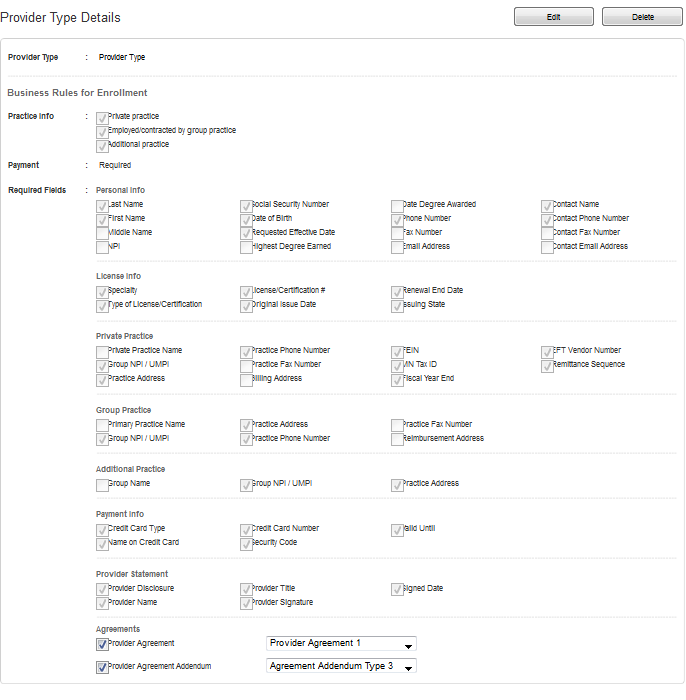


#### Display Provider Type Details

* The user will be able to view the details of the provider type by pressing the “View Business Rule” hyperlink:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Provider Type | The name of the Provider Type | String, max 100 chars, non empty. | Y |
| Practice Info | The practice information of the provider type. | List of checkboxes.  The values of the checkboxes are:  Private practice;  Employed/contracted by group practice;  Additional practice | Y |
| Payment | The field to indicate if the payment is required for this provider type. | String, max 20 chars, non empty.  Required/Not Required. | Y |
| Required Fields | The required fields of the provider type. | List of checkboxes.  The values of the checkboxes are:  **Personal Info tab:**  Last Name;  First Name;  Middle Name;  NPI;  Social Security Number;  Date of Birth;  Requested Effective Date;  Highest Degree Earned;  Date Degree Awarded;  Phone Number;  Fax Number;  Email Address;  Contact Name;  Contact Phone Number;  Contact Fax Number;  Contact Email Address;  **License Info:**  Specialty;  Type of License/Certification;  License/Certification #;  Original Issue Date;  Renewal End Date;  Issuing State;  Private Practice:  Private Practice Name;  Group NPI / UMPI;  Practice Address;  Practice Phone Number;  Practice Fax Number;  Billing Address;  FEIN;  MN Tax ID;  Fiscal Year End;  EFT Vendor Number;  Remittance Sequence;  **Group Practice:**  Primary Practice Name  Group NPI / UMPI  Practice Address  Practice Phone Number  Practice Fax Number  Reimbursement Address  **Additional Practice:**  Group Name;  Group NPI / UMPI;  Practice Address  Payment Info:  Credit Card Type;  Name on Credit Card;  Credit Card Number;  Security Code;  Valid Until  **Provider Statement:**  Provider Disclosure;  Provider Name;  Provider Title;  Provider Signature  Signed Date  **Agreement:**  Provider Agreement;  Provider Agreement Addendum |  |

* The screen example is like following:



#### Delete Provider Type

* The user will press “Delete” button for the chosen entities (chosen by checkboxes).
* Please refer chapter 2.18.2 for more details.

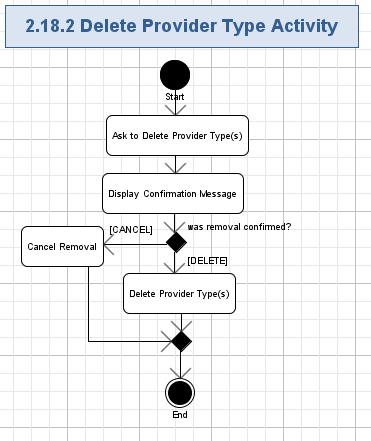
#### Create Provider Type

* The user will be able to create new provider type.
* Please refer chapter 2.18.3 for more details.

#### Edit Provider Type

* The user will be able to edit existing provider type.
* Please refer chapter 2.18.4 for more details.

### Delete Provider Type Activity



#### Ask to Delete Provider Type(s)

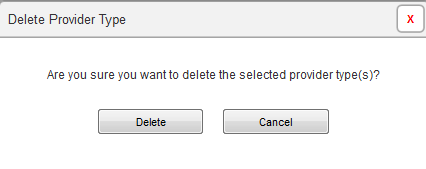
* The user will press “Delete” button for the chosen entities(chosen by checkboxes)

#### Display Confirmation Popup

* The user will press “Delete” button for the chosen entities(chosen by checkboxes)
* The application will display the confirmation popup window with the following message on the removal of the chosen entities.

|  |
| --- |
| **Delete Provider Type**  *Are you sure to delete the selected provider type(s)?*  *<Yes>, <No>* |

* The user has to confirm removal before it will be actually performed.
* The screen example is like following:



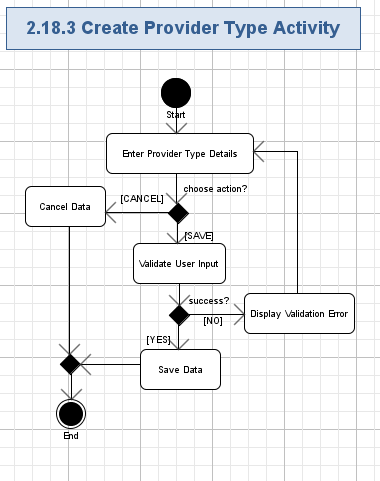
#### Cancel Removal

* The user can press “No” button to cancel removal of the chosen entities.
* No entity is removed in this case.

#### Delete Provider Type(s)

* The user can press “Delete” button to confirm removal of the chosen entities.
* The application will remove the related entities from the system – the data for that entity will be no more shown on the list of entities.

### Create Provider Type Activity



#### Enter Provider Type Details

* The application will ask the user to enter the new entity’s data according to the following table:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Provider Type | The name of the Provider Type | String, max 100 chars, non empty. | Y |
| Practice Info | The practice information of the provider type. | List of checkboxes.  The values of the checkboxes are:  Private practice;  Employed/contracted by group practice;  Additional practice | Y |
| Payment | The field to indicate if the payment is required for this provider type. | String, max 20 chars, non empty.  Required/Not Required. | Y |
| Required Fields | The required fields of the provider type. | List of checkboxes.  The values of the checkboxes are:  **Personal Info tab:**  Last Name;  First Name;  Middle Name;  NPI;  Social Security Number;  Date of Birth;  Requested Effective Date;  Highest Degree Earned;  Date Degree Awarded;  Phone Number;  Fax Number;  Email Address;  Contact Name;  Contact Phone Number;  Contact Fax Number;  Contact Email Address;  **License Info:**  Specialty;  Type of License/Certification;  License/Certification #;  Original Issue Date;  Renewal End Date;  Issuing State;  Private Practice:  Private Practice Name;  Group NPI / UMPI;  Practice Address;  Practice Phone Number;  Practice Fax Number;  Billing Address;  FEIN;  MN Tax ID;  Fiscal Year End;  EFT Vendor Number;  Remittance Sequence;  **Group Practice:**  Primary Practice Name  Group NPI / UMPI  Practice Address  Practice Phone Number  Practice Fax Number  Reimbursement Address  **Additional Practice:**  Group Name;  Group NPI / UMPI;  Practice Address  Payment Info:  Credit Card Type;  Name on Credit Card;  Credit Card Number;  Security Code;  Valid Until  **Provider Statement:**  Provider Disclosure;  Provider Name;  Provider Title;  Provider Signature  Signed Date  **Agreement:**  Provider Agreement;  Provider Agreement Addendum |  |

#### Cancel Date

* The user can cancel data of the entity by pressing “Cancel” button.
* No entity is created in this case

#### Validate User Input

* The user can press “Save” button to initiate validation (and further saving) of the entered/updated data.
* The application will automatically validate all the user input for all the required fields according to the table from chapter 2.18.3.1.
* The user can’t proceed until providing the correct data in the fields.

#### Display Validation Error

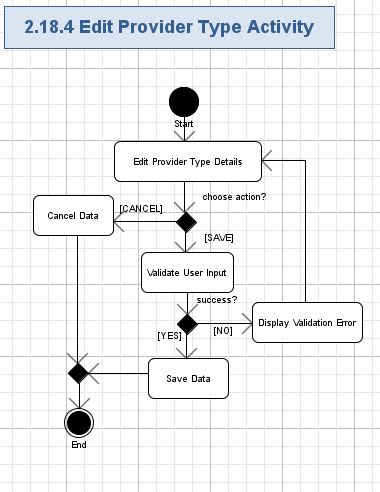
* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> is required.*  *Etc.* |

#### Save Data

* The application will create a new provider type entity.

### Edit Provider Type Activity



#### Edit Provider Type Details

* The application will ask the user to update existing entity’s data according to the table:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Provider Type | The name of the Provider Type | String, max 100 chars, non empty. | Y |
| Practice Info | The practice information of the provider type. | List of checkboxes.  The values of the checkboxes are:  Private practice;  Employed/contracted by group practice;  Additional practice | Y |
| Payment | The field to indicate if the payment is required for this provider type. | String, max 20 chars, non empty.  Required/Not Required. | Y |
| Required Fields | The required fields of the provider type. | List of checkboxes.  The values of the checkboxes are:  **Personal Info tab:**  Last Name;  First Name;  Middle Name;  NPI;  Social Security Number;  Date of Birth;  Requested Effective Date;  Highest Degree Earned;  Date Degree Awarded;  Phone Number;  Fax Number;  Email Address;  Contact Name;  Contact Phone Number;  Contact Fax Number;  Contact Email Address;  **License Info:**  Specialty;  Type of License/Certification;  License/Certification #;  Original Issue Date;  Renewal End Date;  Issuing State;  Private Practice:  Private Practice Name;  Group NPI / UMPI;  Practice Address;  Practice Phone Number;  Practice Fax Number;  Billing Address;  FEIN;  MN Tax ID;  Fiscal Year End;  EFT Vendor Number;  Remittance Sequence;  **Group Practice:**  Primary Practice Name  Group NPI / UMPI  Practice Address  Practice Phone Number  Practice Fax Number  Reimbursement Address  **Additional Practice:**  Group Name;  Group NPI / UMPI;  Practice Address  Payment Info:  Credit Card Type;  Name on Credit Card;  Credit Card Number;  Security Code;  Valid Until  **Provider Statement:**  Provider Disclosure;  Provider Name;  Provider Title;  Provider Signature  Signed Date  **Agreement:**  Provider Agreement;  Provider Agreement Addendum |  |

#### Cancel Date

* The user can cancel data of the entity by pressing “Cancel” button.
* No existing entity is updated in this case.

#### Validate User Input

* The user can press “Save” button to initiate validation (and further saving) of the entered/updated data.
* The application will automatically validate all the user input for all the required fields according to the table from chapter 2.18.4.1.
* The user can’t proceed until providing the correct data in the fields.

#### Display Validation Error

* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> is required.*  *Etc.* |

#### Save Data

* The application will update data of the existing provider type entity respectively.

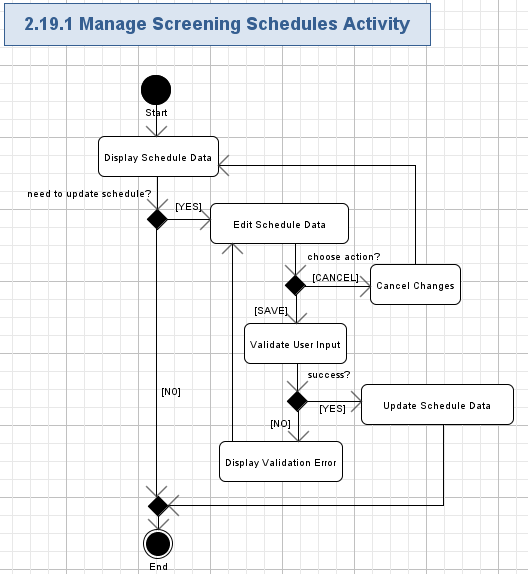
## Manage Screening Schedules

The user can manage the screening schedules on the GUI page. It will be possible to view existing schedule or update the schedule. The screening schedule includes the upcoming screening date and interval.  
Conceptualization Reference: None

Wireframes Page: “Functions\_\_Service\_Admin\_.html”

* Pre-conditions: the user pressed “Function” menu in the top navigation menu and then selected “Screen Schedules”. The user has to be logged in to the application to perform this action.
* Post-conditions: the user successfully managed the screening schedules.

### Manage Screening Schedules Activity

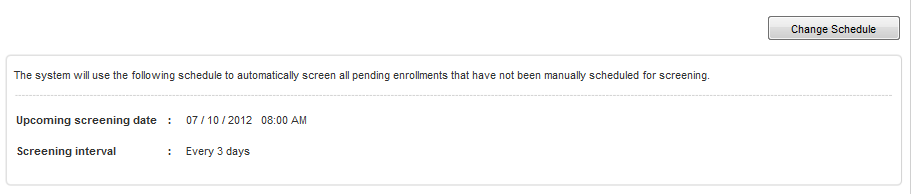


#### Display Schedule Data

* The application will provide a simple single page where all fields of the schedule are shown.
* The application will display the following fields:

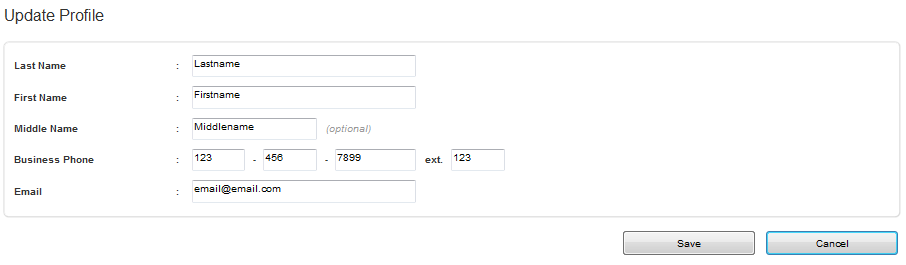
| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Upcoming screen date | The date for upcoming screen | String, max 20 chars, non empty.  Date/Time: MM/DD/YYYY hh:mm AM/PM | Y |
| Screening interval | The description of the screening interval. | String, max 50 chars, non empty.  The format is: Every <N> Days.  N is integer. | Y |

* The screen shot example of displaying the schedule is like following:



#### Edit Schedule Data

* The user can ask the application to edit the schedule by pressing “Change Schedule” button.
* Please refer to the table from chapter 2.19.1.1 for more information about schedule fields, which can be edited.
* The screen shot example of editing schedule is like following:



#### Cancel Changes

* The user can press “Cancel” button to reject updating of the existing schedule.
* No updates of the existing schedule will be performed.

#### Validate User Input

* The user will press “Save” button to update user schedule fields according to the edited fields.
* The application will automatically validate all the user input for all the required fields according to table and rules from chapter 2.19.1.1.
* The user can not proceed until providing the correct data.

#### Display Validation Error

* If the validation failed, then the validation error icon (simply red “\*” character) will be displayed nearby the wrong field and there will be red validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *This field is required.*  *// etc…* |

#### Update Schedule Data

* The application will update an existing schedule with the edited data.

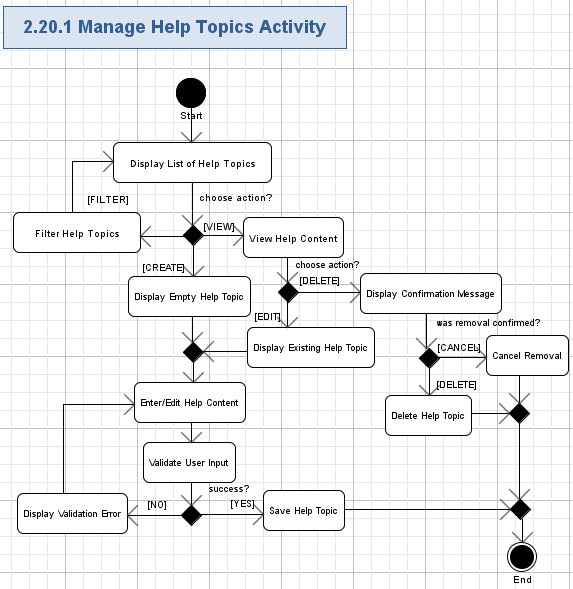
## Manage Help Topics

The user can manage the help topics on the GUI page. It will be possible to view existing help topics, create new help topics, update help topics, or remove them.   
Conceptualization Reference: None

Wireframes Page: “Functions\_\_Service\_Admin\_.html”

* Pre-conditions: the user pressed “Function” menu in the top navigation menu and then selected “Help Topic”. The user has to be logged in to the application to perform this action.
* Post-conditions: the user successfully managed the help topics.

### Manage Help Topics Activity



#### Display List of Help Topics

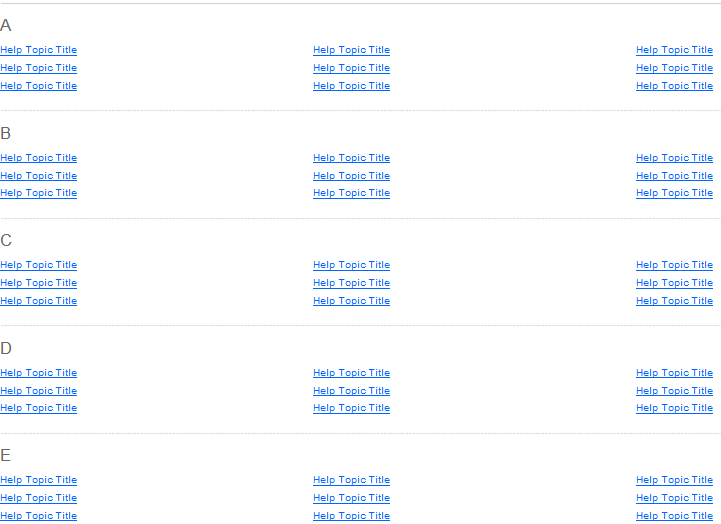
* The application will show the header of the page like following:

|  |
| --- |
| *Help Topics* |

* The application will display list of help topics.
* Each topics in the list will correspond to the related help topic and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| <Name> | The name of the help topic. | String, max 100 chars, non empty.  It will be a hyperlink to view the related help content– please refer chapter 2.20.1.3 for more details. | Y |

* The topics will be grouped by “A”, “B”, “C”, ….
* The screen shot example of the help list is as following:



#### Filter Help Topics

* The application will show alphabetical reference to all the help topics.
* The alphabetical reference will be applied to the names of the topics.
* The “All”, A, B, C, … , Z hyperlinks will be present on the alphabetical reference.
* “All” reference will be selected by defaults – it means all the help topics will be shown on the list.
* The user can press any hyperlink (i.e. English letter) in the alphabetical reference and filter the list of help documents by topic names, starting from that chosen letter (or All – for all the documents).
* The screen example of filtering for help topics is as following:



#### View Help Content

* The user can view any of the help topics, shown on the list by pressing the topic’s name hyperlink.
* The content of the help topic will contain the following fields:

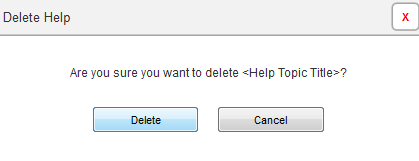
| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| <Name> | The name of the help topic. | String, max 100 chars, non empty. | Y |
| Description | The description of the help topic | String, max 10,000 chars, non empty. | Y |

#### Display Confirmation Popup

* The user will press “Delete” button for the chosen help
* The application will display the confirmation popup window with the following message on the removal of the chosen help.

|  |
| --- |
| **Delete Help**  *Are you sure you want to delete <Help Topic Title>?*  *<Delete>, <Cancel>* |

* The user has to confirm removal before it will be actually performed.
* The screen example is like following:



#### Cancel Removal

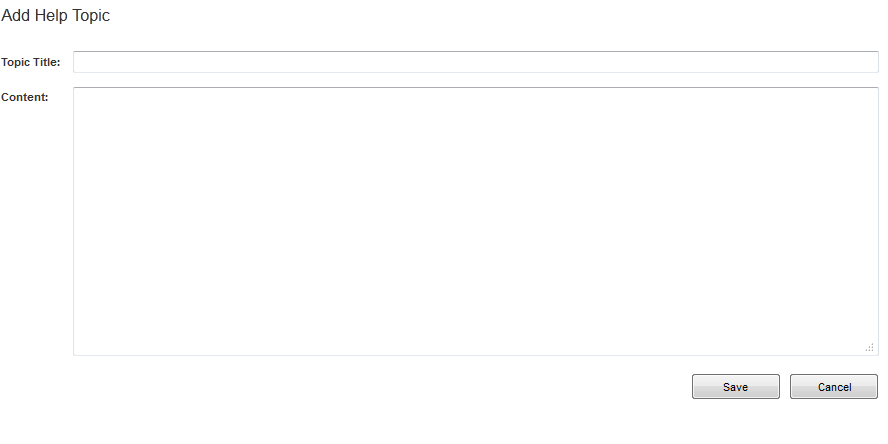
* The user can press “No” button to cancel removal of the chosen help.
* No help is removed in this case.

#### Delete Help Topic

* The user can press “Delete” button to confirm removal of the chosen help.
* The application will remove the related help from the system – the data for that entity will be no more shown on the list of entities.

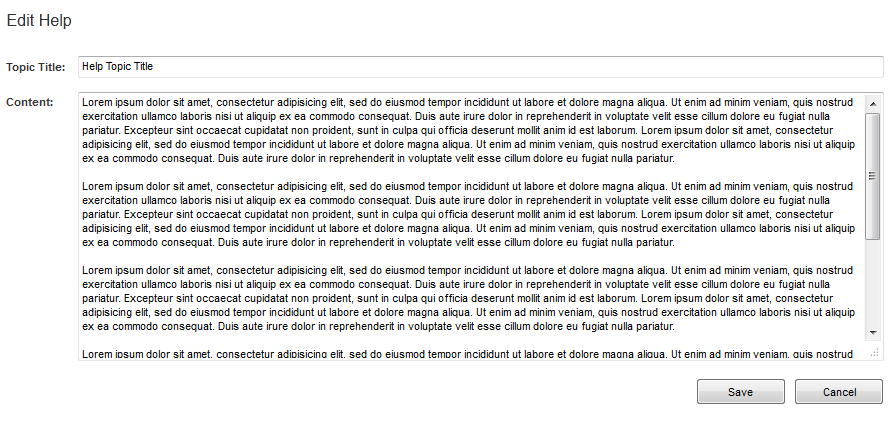
#### Display Empty Help Topic

* The application will display the new page for creating a help topic.
* The fields will be the same as described in 2.20.1.3.
* The screen example is like following:



#### Display Existing Help Topic

* The application will display the new page for updating a help topic.
* The fields will be the same as described in 2.20.1.3.
* The screen example is like following:



#### Enter/Edit Help Content

* The application will ask the user to enter the new entity’s data or update existing entity’s data according to the table from chapter 2.20.1.3.

#### Validate User Input

* The user can press “Save” button to initiate validation (and further saving) of the entered/updated data.
* The application will automatically validate all the user input for all the required fields according to the table from chapter 2.20.1.3.
* The user can’t proceed until providing the correct data in the fields.

#### Display Validation Error

* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> is required.*  *Etc.* |

#### Save Help Topic

* The application will create a new Help Topic entity or update data of the existing Help Topic entity respectively.

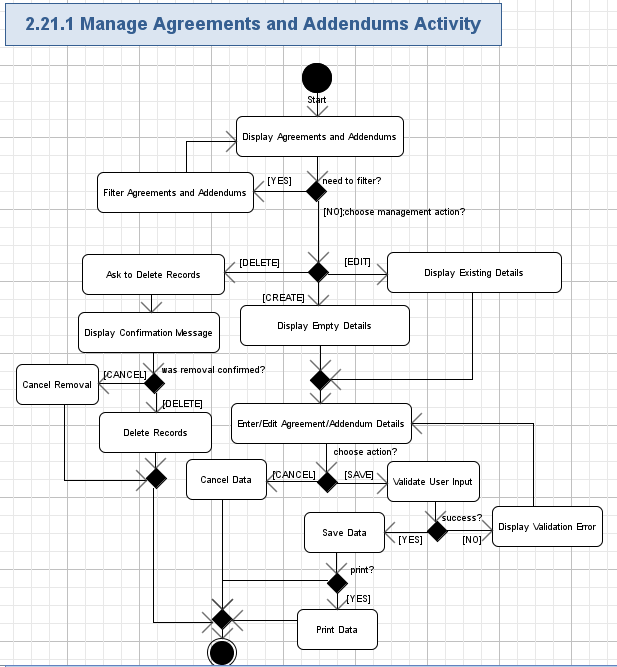
## Manage Agreements and Addendums

The user can manage the Agreements and Addendums on the GUI page. It will be possible to view existing Agreements and Addendums, create new Agreements and Addendums, update Agreements and Addendums, or remove them. The Agreements and Addendums are legal documents that can be used during the enrollment process by providers.  
Conceptualization Reference: None

Wireframes Page: “Functions\_\_Service\_Admin\_.html”

* Pre-conditions: the user pressed “Function” menu in the top navigation menu and then selected “Agreements and Addendums”. The user has to be logged in to the application to perform this action.
* Post-conditions: the user successfully managed the Agreements and Addendums.

### Manage Agreements and Addendums Activity



#### Display Agreements and Addendums

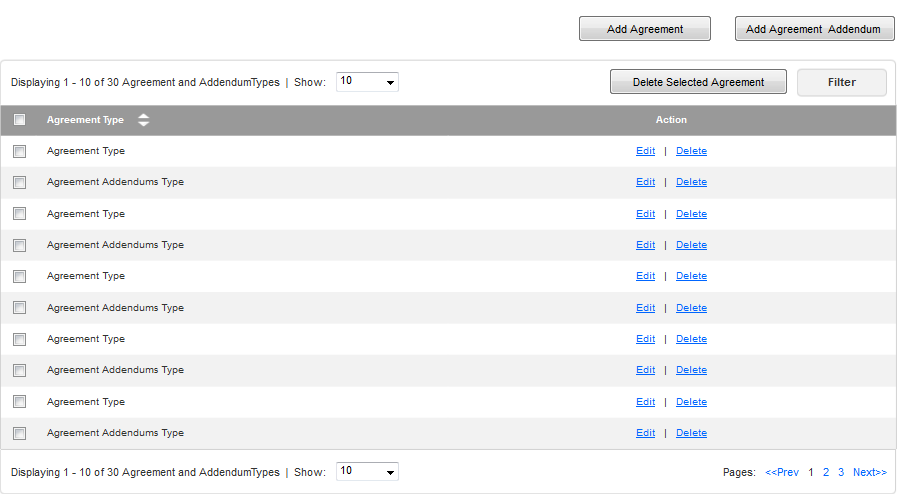
* The application will display the page able table with the list of all the existing Agreements and Addendums in the system.
* Each row in the table will correspond to the related Agreements and Addendums and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Agreement Type | The name of the agreement Type | String, max 100 chars, non empty. | Y |
| <Edit> | The hyperlink to edit the agreement/addendum. | Hyperlink. | Y |
| <Delete> | The hyperlink to delete the agreement/addendum. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 5, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z>* Agreements and Addendums |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of entities in the system.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The Application will allow the user to sort the table by “Agreement Type” column (DES/ASC).
* The screen example is like following:

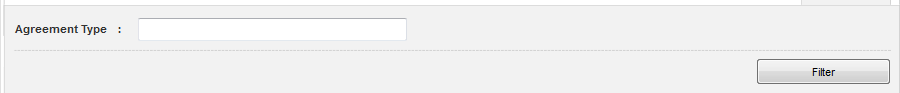


#### Filter Agreements and Addendums

* The user will be able to filter the Agreements and Addendums with the following criteria:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Agreement Type | The name of the agreement Type | String, max 100 chars, can be empty. | N |

* The screen example is like following:



#### Ask to Delete Records

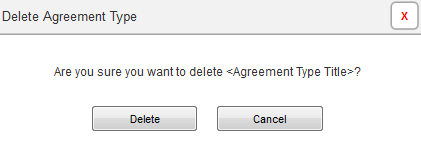
* The user will press “Delete” button for the chosen entities(chosen by checkboxes)

#### Display Confirmation Popup

* The user will press “Delete” button for the chosen entities(chosen by checkboxes)
* The application will display the confirmation popup window with the following message on the removal of the chosen entities.

|  |
| --- |
| **Delete Agreement Type**  *Are you sure you want to delete <Agreement Type Title>?*  *<Delete>, <Cancel>* |

* The user has to confirm removal before it will be actually performed.
* The screen example is like following:



#### Cancel Removal

* The user can press “No” button to cancel removal of the chosen entities.
* No entity is removed in this case.

#### Delete Records

* The user can press “Delete” button to confirm removal of the chosen entities.
* The application will remove the related entities from the system – the data for that entity will be no more shown on the list of entities.

#### Display Empty Details

* The application will display the new page for creating a new entry.
* The fields will be as following:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Agreement Title | The name of the agreement title. | String, max 100 chars, non empty. | Y |
| Content | The content of the agreement type. | String, max 10,000 chars, non empty. | Y |

#### Display Existing Details

* The application will display the new page for updating an entry.
* The fields will be as following:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Agreement Title | The name of the agreement title. | String, max 100 chars, non empty. | Y |
| Content | The content of the agreement type. | String, max 10,000 chars, non empty. | Y |

#### Enter/Edit Agreement/Addendum Details

* The application will ask the user to enter the new entity’s data according to the table:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Agreement Title | The name of the agreement title. | String, max 100 chars, non empty. | Y |
| Content | The content of the agreement type. | String, max 10,000 chars, non empty. | Y |

#### Cancel Date

* The user can cancel data of the entity by pressing “Cancel” button.
* No entity is created in this case.

#### Validate User Input

* The user can press “Save” button to initiate validation (and further saving) of the entered data.
* The application will automatically validate all the user input for all the required fields according to the table from chapter 2.21.1.9.
* The user can’t proceed until providing the correct data in the fields.

#### Display Validation Error

* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> is required.*  *Etc.* |

#### Save Data

* The application will create a new agreement type entity.

#### Print Data

* The user can choose to print the data.

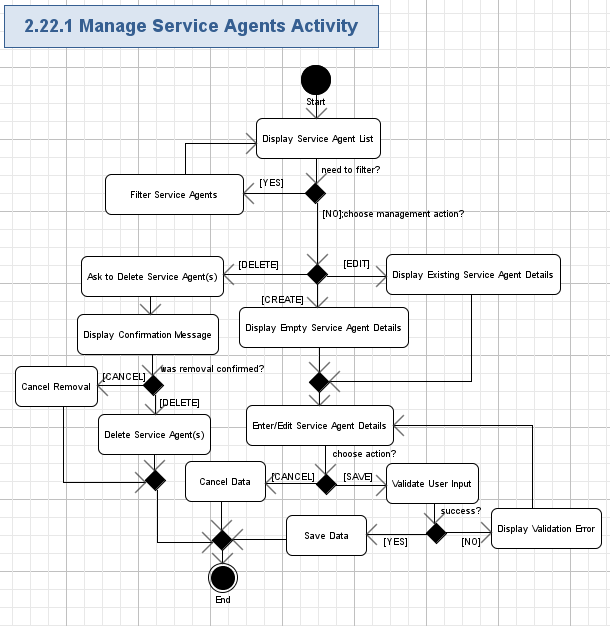
## Manage Service Agents

The user can manage the Service Agents on the GUI page. It will be possible to view existing Service Agents, create new Service Agents, update Service Agents, or remove them. The service admin can manage the agent accounts.  
Conceptualization Reference: None

Wireframes Page: “Functions\_\_Service\_Admin\_.html”

* Pre-conditions: the user pressed “Function” menu in the top navigation menu and then selected “Service Agents”. The user has to be logged in to the application to perform this action.
* Post-conditions: the user successfully managed the Service Agents.

### Manage Service Agents Activity



#### Display Service Agent List

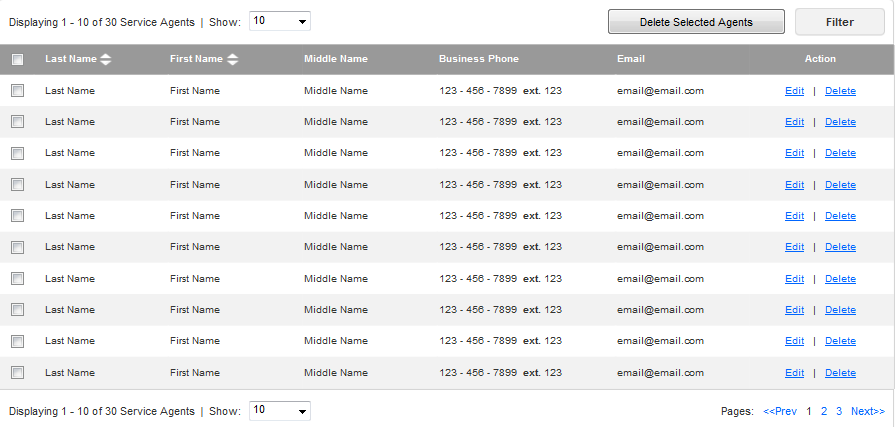
* The application will display the page able table with the list of all the existing service agents in the system.
* Each row in the table will correspond to the related service agent and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Last Name | The last name of the service agent. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the service agent. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the service agent. | String, max 50 chars, can be empty. | N |
| Business Phone | The business phone number of the service agent. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, non empty | Y |
| <Edit> | The hyperlink to edit the service agent. | Hyperlink. | Y |
| <Delete> | The hyperlink to delete the service agent. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 5, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z> Service Agents* |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of entities in the system.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The Application will allow the user to sort the table by “Provider Type” column (DES/ASC).
* The screen example is like following:

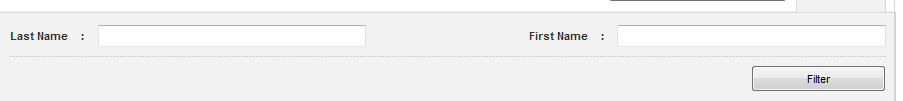


#### Filter Service Agents

* The user will be able to filter the service agents with the following criteria:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Last Name | The last name of the service agent. | String, max 50 chars, can be empty. | N |
| First Name | The first name of the service agent. | String, max 50 chars, can be empty. | N |

* The screen example is like following:



#### Ask to Delete Service Agent(s)

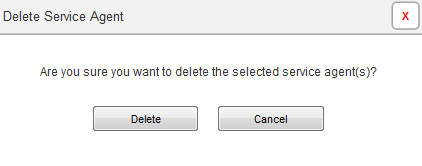
* The user will press “Delete” button for the chosen entities(chosen by checkboxes)

#### Display Confirmation Popup

* The user will press “Delete” button for the chosen entities(chosen by checkboxes)
* The application will display the confirmation popup window with the following message on the removal of the chosen entities.

|  |
| --- |
| **Delete Service Agent**  *Are you sure to delete the selected service agent (s)?*  *<Delete>, <Cancel>* |

* The user has to confirm removal before it will be actually performed.
* The screen example is like following:



#### Cancel Removal

* The user can press “No” button to cancel removal of the chosen entities.
* No entity is removed in this case.

#### Delete Service Agent(s)

* The user can press “Delete” button to confirm removal of the chosen entities.
* The application will remove the related entities from the system – the data for that entity will be no more shown on the list of entities.

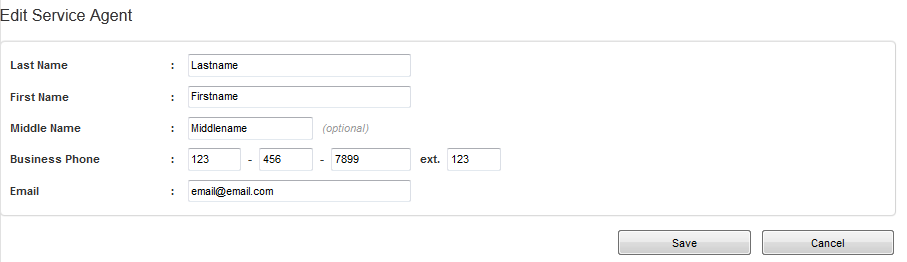
#### Display Empty Service Agent Details

* The application will display the new page for creating a service agent.
* The fields will be the same as described in 2.22.1.1.
* The screen example is like following:



#### Display Existing Service Agent Details

* The application will display the new page for updating a service agent.
* The fields will be the same as described in 2.22.1.1.
* The screen example is like following:



#### Enter/Edit Service Agent Details

* The application will ask the user to enter the new entity’s data or update existing entity’s data according to the table from chapter 2.22.1.1.

#### Cancel Date

* The user can cancel data of the entity by pressing “Cancel” button.
* No entity is created in this case and no existing entity is updated.

#### Validate User Input

* The user can press “Save” button to initiate validation (and further saving) of the entered/updated data.
* The application will automatically validate all the user input for all the required fields according to the table from chapter 2.22.1.1.
* The user can’t proceed until providing the correct data in the fields.

#### Display Validation Error

* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> is required.*  *Etc.* |

#### Save Data

* The application will create a new service agent entity or update data of the existing service agent entity respectively.

## Search Users

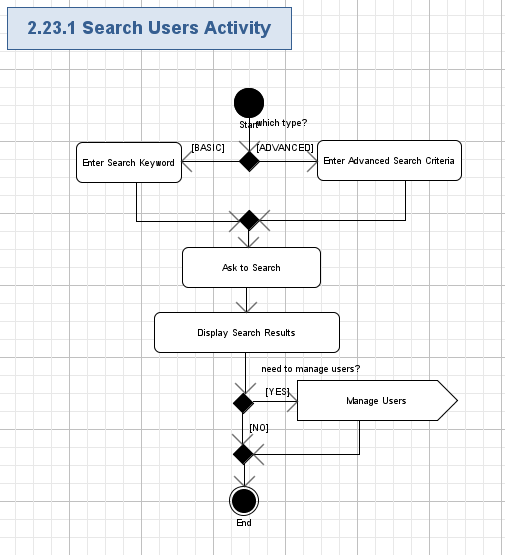
The application allows the system admin to search users with given criteria. The system admin can perform basic search or advanced search on users.

Conceptualization Reference: None.

Wireframes page: “Search\_Result\_\_System\_Admin\_.html” and “Advanced\_Search\_\_System \_Admin\_.html”.

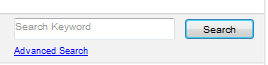
* Pre-conditions: the user needed to search the users in the application. The user has to be logged in to the application to perform this action.
* Post-conditions: the application shown the found users.

### Search Users Activity



#### Enter Search Keyword

* The application will show the textbox on the homepage (template) for the system admin to enter the keyword (String, max 100, can be empty) to search the users in the application.
* The screen shot example is like following:

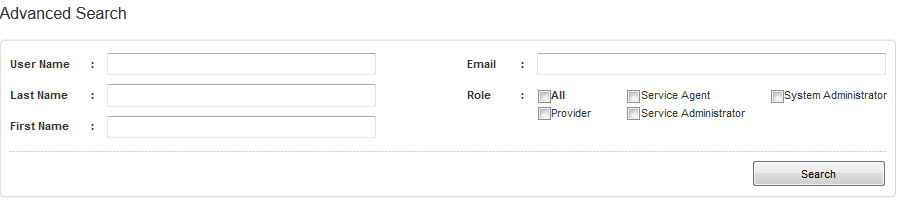


#### Enter Advanced Search Criteria

* The user can also choose to perform advanced search by pressing the “Advanced Search” on the homepage (template).
* The application will show following criteria for the system admin to enter for searching users:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Last Name | The last name of the user. | String, max 50 chars, can be empty. | N |
| First Name | The first name of the user. | String, max 50 chars, can be empty. | N |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| Phone | The phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, can be empty | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |
| Role | The role of the user. | String, max 20 chars, can be empty.  Checkboxes with values:  All;  Provider;  Service Agent;  Service Admin;  System Admin; | N |

* The screen shot example is like following:



#### Ask to Search

* The user can ask the application to perform search users by pressing the “Search” button.

#### Display Search Results

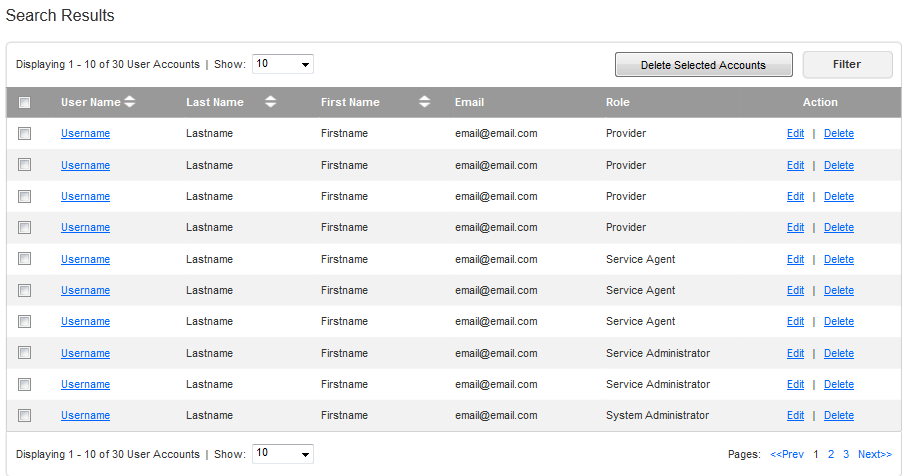
* The application will search users with the given criteria/keyword.
* The list of results will be shown in a page-able table with the following columns:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| Phone | The phone number of the user. | String, like “xxx xxx-xxxx” ext “xxx”, where “x” is a digit [0-9], max 20 chars, non empty | Y |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, non empty | Y |
| Role | The role of the user. | String, max 20 chars, non empty. | Y |
| <Edit> | The hyperlink to edit the user. | Hyperlink. | Y |
| <Delete> | The hyperlink to delete the user. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z> User Accounts* |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of records.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The user can navigate through the table pages by using standard navigation buttons, mentioned above.
* The user can sort the table with any column except “Role” column (ASC/DES).
* The screen shot example of the search results is like following:



#### Manage Users

* The admin can choose to manage the users (view, edit or delete) from the search results.
* Please refer chapter 2.24 for more details.

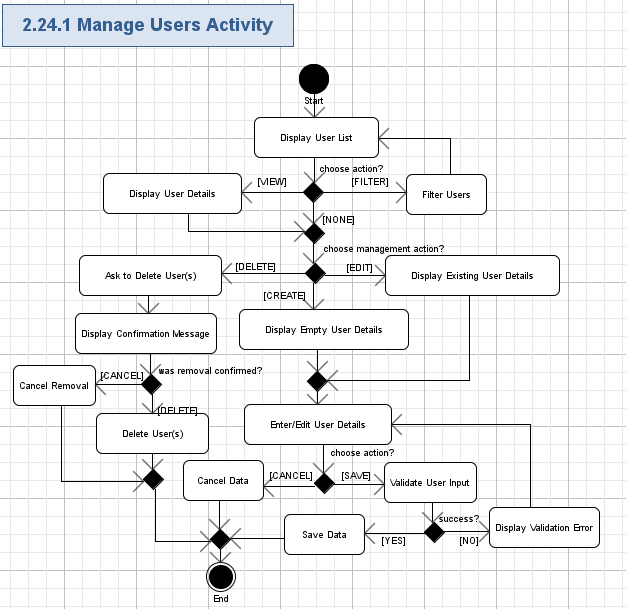
## Manage Users

The system admin can manage the users on the GUI page. It will be possible to view existing users, create new users, update users, or remove them. The system admin can manage all user accounts in the system.  
Conceptualization Reference: None

Wireframes Page: “User\_Accounts\_\_System\_Admin\_.html” and child pages.

* Pre-conditions: the system admin pressed “User Accounts” menu in the top navigation menu. The user has to be logged in to the application to perform this action.
* Post-conditions: the system successfully managed the users.

### Manage Users Activity



#### Display User List

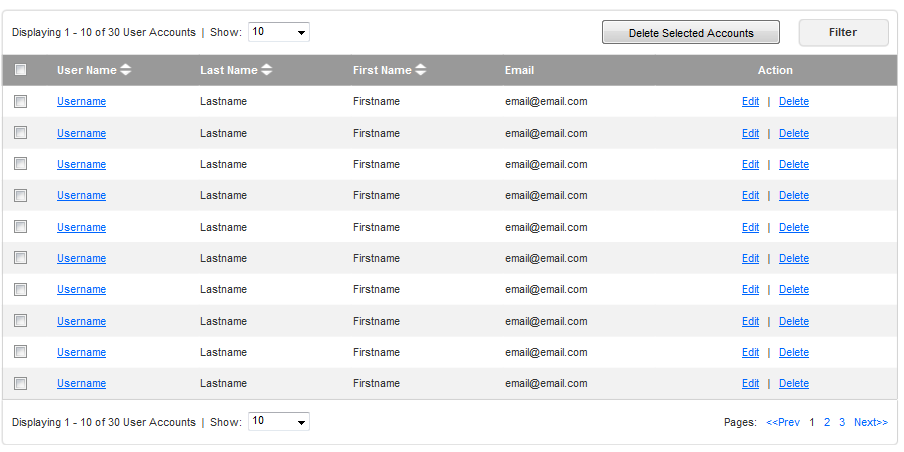
* The application will display the page able table with the list of all the existing users in the system by user role: Provider, Service Agent, Service Admin and System Admin.
* Each row in the table will correspond to the related user and contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| User Name | The user name of the user. | String, max 100 chars, non empty.  Hyperlink to view the details. | Y |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, non empty | Y |
| <Edit> | The hyperlink to edit the user. | Hyperlink. | Y |
| <Delete> | The hyperlink to delete the user. | Hyperlink. | Y |

* The table even rows have to be highlighted.
* The total number of rows on the table page will be selectable (like, 5, 10 – default, 20, 50, 100, All).
* The application will show count of rows, currently displayed on the table, like following:

|  |
| --- |
| *Displaying <X> - <Y> of <Z> User Accounts* |

* Where:
  + <X> - the order number of the first row, displayed on the list (Positive integer or 0),
  + <Y> - the order number of the last row, displayed on the list (Positive integer or 0, must not be lesser than <X>),
  + <Z> - the total count of entities in the system.
* The table will have the standard navigation buttons: “<< Prev”, 1, 2, etc. page numbers, “Next >>”.
* The Application will allow the user to sort the table by any column except “Email” column (DES/ASC).
* The screen example is like following:

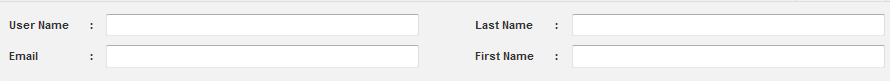


#### Filter Users

* The user will be able to filter the users with the following criteria:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Last Name | The last name of the service agent. | String, max 50 chars, can be empty. | N |
| First Name | The first name of the service agent. | String, max 50 chars, can be empty. | N |
| User Name | The user name of the user. | String, max 100 chars, can be empty.  Hyperlink to view the details. | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, can be empty | N |

* The screen example is like following:

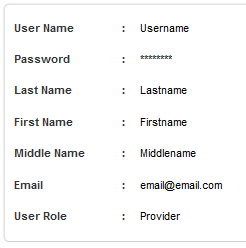


#### Display User Details

* The application will display the details of the user if the system admin presses the “User Name” hyperlink.
* The details of the user contain the following fields:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| User Name | The user name of the user. | String, max 100 chars, non empty. | Y |
| Password | The password of the user name. | String, max 50 chars, non empty.  Displayed as “\*”. | Y |
| Last Name | The last name of the user. | String, max 50 chars, non empty. | Y |
| First Name | The first name of the user. | String, max 50 chars, non empty. | Y |
| Middle Name | The middle name of the user. | String, max 50 chars, can be empty. | N |
| Email | The e-mail address of the user. | String, max 100 chars, must be a valid e-mail, non empty | Y |
| User Role | The role of the user. | String, max 20 chars, non empty. | Y |

* The screen example is like following:



#### Ask to Delete User(s)

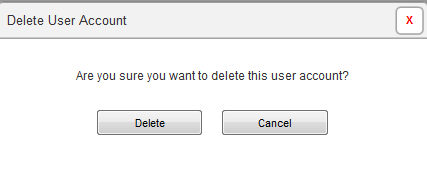
* The user will press “Delete” button for the chosen entities(chosen by checkboxes)

#### Display Confirmation Popup

* The user will press “Delete” button for the chosen entities(chosen by checkboxes)
* The application will display the confirmation popup window with the following message on the removal of the chosen entities.

|  |
| --- |
| **Delete User**  *Are you sure to delete this user account?*  *<Delete>, <Cancel>* |

* The user has to confirm removal before it will be actually performed.
* The screen example is like following:



#### Cancel Removal

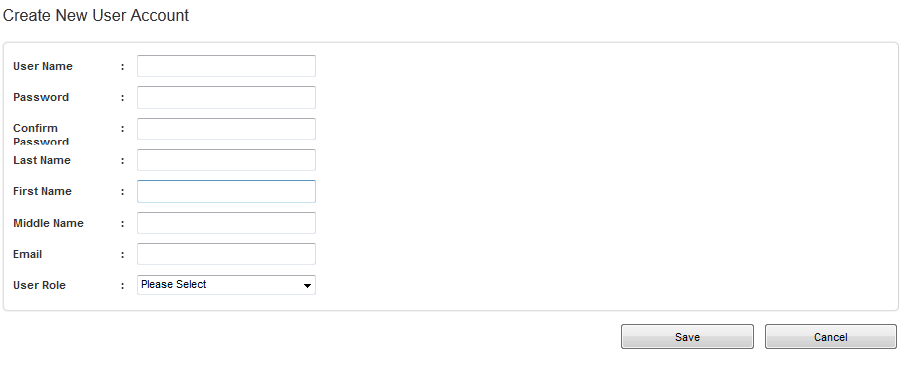
* The user can press “No” button to cancel removal of the chosen entities.
* No entity is removed in this case.

#### Delete User(s)

* The user can press “Delete” button to confirm removal of the chosen entities.
* The application will remove the related entities from the system – the data for that entity will be no more shown on the list of entities.

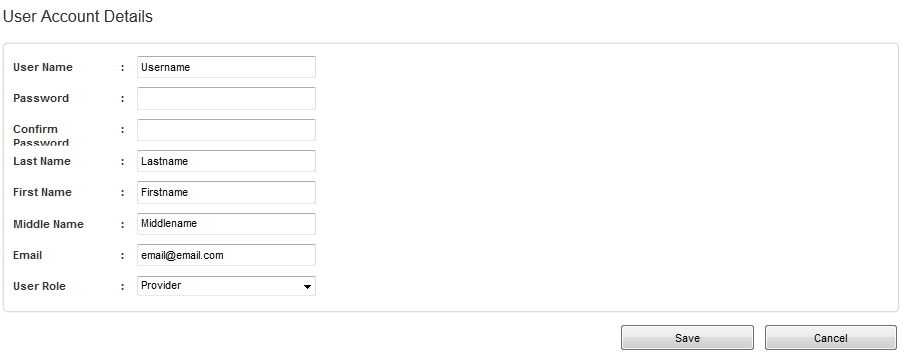
#### Display Empty User Details

* The application will display the new page for creating a user.
* The fields will be the same as described in 2.24.1.3.
* The screen example is like following:



#### Display Existing User Details

* The application will display the new page for updating a user.
* The fields will be the same as described in 2.24.1.3.
* The screen example is like following:



#### Enter/Edit User Details

* The application will ask the user to enter the new entity’s data or update existing entity’s data according to the table from chapter 2.24.1.3.

#### Cancel Date

* The user can cancel data of the entity by pressing “Cancel” button.
* No entity is created in this case and no existing entity is updated.

#### Validate User Input

* The user can press “Save” button to initiate validation (and further saving) of the entered/updated data.
* The application will automatically validate all the user input for all the required fields according to the table from chapter 2.22.1.1.
* The user can’t proceed until providing the correct data in the fields.

#### Display Validation Error

* If the validation failed, then the validation icon (or just “\*” character) will be displayed nearby the wrong field and there will be validation message like this (concrete validation error relates to the failed field):

|  |
| --- |
| *Field <FieldName> is required.*  *Etc.* |

#### Save Data

* The application will create a new user entity or update data of the existing user entity respectively.

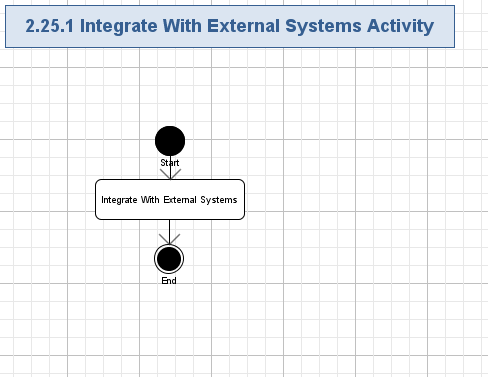
## Integrate With External Systems

The application will automatically integrate with existing systems like security system and data sources etc.

Conceptualization Reference: 4.4.35 Integrate With External Systems

* Pre-conditions: the application was started.
* Post-conditions: the application integrated with the external systems.

### Integrate With External Systems Activity



#### Integrate With External Systems

* The new system will need to integrate with existing systems.
* The new application will integrate with the existing security system.
* The new application will integrate with the existing data sources.

(For example, with the existing license database system)

* The new application will integrate with existing authoritative data sources such as:
* Terminated Providers (i.e. Terminated for cause) (CMS)
* o Professional Licensing Board: Dentists, Optometrists, Doctors, etc. (State). (only if applicable)
* o MediCare Certified Providers (CMS) (only if applicable)
* o SSDM: (Social Security Death Master) (SocSecAdmin)
* o L E I E: (List of Excluded Individuals and Entities) (OIG)
* o E P L S: (Excluded Providers List System) (HHS)
* o NNPES: (National Plan and Provider Enumeration System) (CMS) (only if applicable).
* o Previously Denied Applicants.
* o National Practitioner Data Bank.
* The new application will integrate with the PVS for data validation.
* The new application will integrate with the SIRS for integrity check.
* The new application will integrate with the NetStudy (Licensing) for background check.
* The new system will be Compliant with MITA 3.0 architecture/framework.
* The new system will be Compliant with the US Section 508 of the Americans with Disabilities Act.

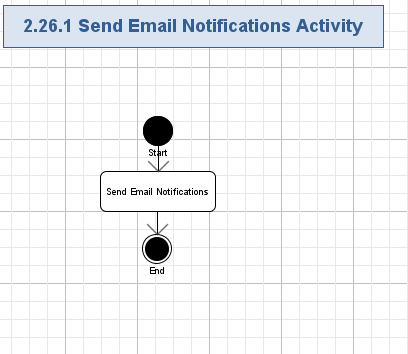
## Send Email Notifications

The system will automatically send various e-mail notifications to users when certain user actions occur (enrolment is denied).

Conceptualization Reference: 4.4.35 Send Email Notifications

* Pre-conditions: something need to send notifications have occurred in the application.
* Post-conditions: the notification e-mail with information on related actions was sent to the system administrator.

### Send Email Notifications Activity



#### Send Email Notifications

* The system will automatically send an e-mail to the user for the following cases:
  + Service admin will receive notification when user submits updated profile.
  + Provider will get notification when his/her enrolment has approved/denied.
  + Service admin will receive notification when enrolment form is submitted for review.
* The e-mail subject will have the following data:

|  |
| --- |
| *Email Notification from CMS Medicaid Provider Screen Portal* |

* The e-mail body will contain the following message:

|  |
| --- |
| *User <Username> perform the operation whose details are < Details>.* |

* Where dynamic data in the e-mail body template is like following:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| <UserName> | Unique username of the user. | String, max 100 chars, non empty | Y |
| <Error Details> | The description of the critical error. | String, max 1024 chars, non empty. | Y |

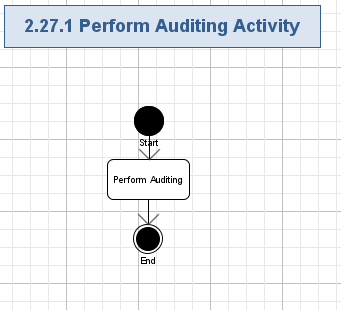
* The system will send an email notification to the user notifying them of the action that occurred and the next step required by them.
* Mailbox for Minnesota State will be handled separately. Minnesota will continue using MN-ITS mailbox for sending email notifications. The solution should support the same.

## Perform Auditing

All the entity creation/modification/removal by the user will be audited by the application. Information about the entities/actions will be audited to the database, so it can be easily retrieved and evaluated in the future by some application management/support staff. This use case is included by all the modification related use cases.  
Conceptualization Reference: 4.4.34 Perform Auditing

* Pre-conditions: some user action with some entity has occurred (i.e. entity was created, or changed, or removed).
* Post-conditions: the user action and entity data was audited.

### Perform Auditing Activity



#### Perform Auditing

* All the user actions will be audited by the application.
* All data modifications (including creating new data and removing of the existing data) will be also audited by the application.
* The next data of each user action will be audited to the database:
  + Username.
  + Date/time (Date/timestamp of the operation).
  + Previous value (if the action was a change to something or removal).
  + New value (if the action was a change to something or creation).
  + Group metadata details
* The data type of the values to be audited are like following:

| **Data Element** | **Description** | **Format** | **R?** |
| --- | --- | --- | --- |
| Username | The unique user name of the user. | String, max 255 chars, non empty.  This field must be a valid e-mail address. | Y |
| Date/time, Date/timestamp of the operation | The timestamp of the auditing moment. | String, date/time format like DD/MM/YYYY hh:mm:ss, non empty. | Y |
| IP Address | The ip address. | String, max 255 chars, can be empty. | N |
| Previous value | The previous data of the audited value (if any). | String, max 2000 chars, can be empty.  It will be a serialized data. | N |
| New value | The new data of the audited value (if any). | String, max 2000 chars, can be empty.  It will be a serialized data. | N |

* All the data will be audited to the local storage.

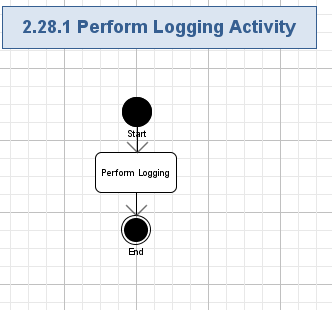
## Perform Logging

The application will automatically log errors, exceptions, warnings and debug information during its execution. Information about users, tried to login to the application will be logged as well. Logging information will be stored to some logging system (like files on the server side). This use case is included by all of other user cases.

Conceptualization Reference: 4.4.33 Perform Logging

* Pre-conditions: the application was started and it will constantly log all problems during its execution.
* Post-conditions: all errors, exceptions, warnings and debug information was logged to logging system (like log files).

### Perform Logging Activity



#### Perform Logging

* The system will log all the errors, exceptions, warning and debug information during the application execution.
* The following data will be logged:
  + Method name (String, max 100 chars, non empty),
  + Description of problem (String, max 1024 chars, non empty),
  + Input arguments (if any, String, max 4096 chars, can be empty),
  + Output results (if any, String, max 4096 chars, can be empty),
  + Date and timestamp (String, full date/time format, max 30 chars, non empty),
  + Etc. – up-to Architect.
* Logging will be performed to the logging system (like log files on the server side).
* The logged information must not contain user password.

# General Requirements

## Graphical User Interface Requirements

### Main GUI Goal

The new application will be user friendly and provide a simplified interface for CMS - Medicaid Provider Screening Portal - Administrative Features functionality.

The wireframes is provided in the wiki”

<http://apps.topcoder.com/wiki/display/docs/CMS+-+Medicaid+Provider+Screening+Portal+-+Administrative+Features>

### Resolution

Minimum supported resolution for desktops is 800x600.

### Supported Browsers

* MS Internet Explorer version 7+.
* Mozilla Firefox version 2+
* Safari 4+
* Google Chrome 4+

## Performance Constraints

User Metrics Requirements:

* Expecting 150,000+ unique users
* Concurrent users expected will be in thousands

Performance Requirements:

* The portal will have 24x7x365 uptime with reasonably high level of availability.
* All pages in the application should be loaded in 3 seconds or less seconds or less after the initial hit.

## Security

### Security Roles

#### Permissions

All security checks will occur against permissions. Each function in the system will validate a user’s permission against the required permission for the task.

#### Roles

One or more permissions will be assigned to roles. A user may have more than one role. Below is a list of roles and permissions. The System user role was added for clarity.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Registered User | Service Agent | Service Admin | System Admin | System |
| 2.1 Login [External] | X | X | X | X |  |
| 2.2 Logout [External] | X | X | X | X |  |
| 2.3 View Homepage | X | X | X | X |  |
| 2.4 Get Help | X | X | X | X |  |
| 2.5 Search Enrollments |  | X | X |  |  |
| 2.6 Manage My Profile |  | X | X |  |  |
| 2.7 Query Enrollment Status |  | X | X |  |  |
| 2.8 View Enrollments |  | X | X |  |  |
| 2.9 View Enrollment Details |  | X | X |  |  |
| 2.10 Create New Enrollment |  | X | X |  |  |
| 2.11 Edit Enrollment |  | X | X |  |  |
| 2.12 Export Enrollment Data |  | X | X |  |  |
| 2.13 Print Enrollment Data |  | X | X |  |  |
| 2.14 Initiate On-Demand Screening |  |  | X |  |  |
| 2.15 Initiate Scheduled Screening |  |  | X |  |  |
| 2.16 Approve Enrollment |  |  | X |  |  |
| 2.17 Reject Enrollment |  |  | X |  |  |
| 2.18 Manage Provider Types |  |  | X |  |  |
| 2.19 Manage Screening Schedules |  |  | X |  |  |
| 2.20 Manage Help Topics |  |  | X |  |  |
| 2.21 Manage Agreements and Addendums |  |  | X |  |  |
| 2.22 Manage Service Agents |  |  | X |  |  |
| 2.23 Search Users |  |  |  | X |  |
| 2.24 Manage Users |  |  |  | X |  |
| 2.25 Integrate With External Systems |  |  |  |  | X |
| 2.26 Send Email Notifications |  |  |  |  | X |
| 2.27 Perform Auditing |  |  |  |  | X |
| 2.28 Perform Logging |  |  |  |  | X |

##### Administrators

User with Service administrator role can manage all the service agent accounts; Users with the system administrator role can manage all the user profiles.

# Required Documentation

## Specification Documentation

* Requirements Specification (this document)
* High Level Use Case Diagrams
* Activity Diagrams
* Logical data model (as needed)
* Quality Assurance Plan (out of scope for this competition)

# Help / User Documentation

None at this time.

# Notes

None at this time.

# Future Enhancements

* In the future, the application may need to include multi-language support

# Glossary

## Definitions

| **Definition** | **Description** |
| --- | --- |
| Google | See [www.google.com](http://www.google.com) |
| Registered User | A user is anybody who is logged in to the application; this is an abstract user role. |
| Service Agent | A service agent is a user who assists Providers in completing the claim forms. The Service Agent may have additional administrative features as well. |
| Service Administrator | A service administrator is a user who has administrative policy rights on the application. This user can edit enroll and screening information. |
| System Administrator | A system administrator is a user able to manage all other users’ data and role. This user role is able to modify the authenticated users on the system as well as their individual roles. |

## Acronyms

|  |  |
| --- | --- |
| **Definition** | **Description** |
| MITA | The Medicaid Information Technology Architecture.  Reference: <http://www.medicaid.gov/Medicaid-CHIP-Program-Information/By-Topics/Data-and-Systems/Medicaid-Information-Technology-Architecture-MITA.html> |
| ESB | Enterprise Service Bus.  Reference: <http://en.wikipedia.org/wiki/Enterprise_service_bus> |
| HIPPA | Health Insurance Portability and Accountability Act of 1996 |
| NNPES | National Plan and Provider Enumeration System) (CMS) (only if applicable |
| GUI | Graphical User Interface |
| CHIP | Children’s Health Insurance Program  Reference: <http://en.wikipedia.org/wiki/State_Children's_Health_Insurance_Program> |
| ID | Identification |
| MPSE | Medicaid Provider Screening and Enrollment. |
| L E I E | List of Excluded Individuals and Entities |
| NASA | National Aeronautics and Space Administration |
| SSDM | Social Security Death Master |
| WAS | Websphere Application Server |
| CMS | Center for Medicaid Services |
| SaaS | Software as a service  Reference: <http://en.wikipedia.org/wiki/Software_as_a_service> |
| CoECI | Center of Excellence for Collaborative Innovation |
| multi-tenant implementation | A single system that can handle any type of service provider |
| JSF | Java Server Faces |
| MCP | MediCare Certified Providers |
| SSDM | Social Security Death Master |
| L E I E | List of Excluded Individuals and Entities |
| E P L S | Excluded Providers List System |
| NNPES | National Plan and Provider Enumeration System |
| EFT | Electronic Funds Transfer |
| PEP | Front-end Provider Enrollment Processor |
| PVS | Provider Validation Service |
| UMPI | Unique Minnesota Provider Identifier |
| NPI | National Provider Identifier |

# 