ONBOARDING

Pega Client Lifecycle Management for Financial Services

Implementation Guide

8.4



©2020 Pegasystems Inc., Cambridge, MA. All rights reserved.

Trademarks

For Pegasystems Inc. trademarks and registered trademarks, all rights reserved. All other trademarks or service marks are property of their respective holders.

For information about the third-party software that is delivered with the product, refer to the third-party license file on your installation media that is specific to your release.

Notices

This publication describes and/or represents products and services of Pegasystems Inc. It may contain trade secrets and proprietary information that are protected by various federal, state, and international laws, and distributed under licenses restricting their use, copying, modification, distribution, or transmittal in any form without prior written authorization of Pegasystems Inc.

This publication is current as of the date of publication only. Changes to the publication may be made from time to time at the discretion of Pegasystems Inc. This publication remains the property of Pegasystems Inc. and must be returned to it upon request. This publication does not imply any commitment to offer or deliver the products or services described herein.

This publication may include references to Pegasystems Inc. product features that have not been licensed by you or your company. If you have questions about whether a particular capability is included in your installation, please consult your Pegasystems Inc. services consultant.

Although Pegasystems Inc. strives for accuracy in its publications, any publication may contain inaccuracies or typographical errors, as well as technical inaccuracies. Pegasystems Inc. shall not be liable for technical or editorial errors or omissions contained herein. Pegasystems Inc. may make improvements and/or changes to the publication at any time without notice.

Any references in this publication to non-Pegasystems websites are provided for convenience only and do not serve as an endorsement of these websites. The materials at these websites are not part of the material for Pegasystems products, and use of those websites is at your own risk.

Information concerning non-Pegasystems products was obtained from the suppliers of those products, their publications, or other publicly available sources. Address questions about non-Pegasystems products to the suppliers of those products.

This publication may contain examples used in daily business operations that include the names of people, companies, products, and other third-party publications. Such examples are fictitious and any similarity to the names or other data used by an actual business enterprise or individual is coincidental.

This document is the property of:

Pegasystems Inc.
One Rogers Street
Cambridge, MA 02142-1209, USA
Phone: 617-374-9600 Fax: 617-374-9620

www.pega.com

Document: Pega Client Lifecycle Management for Financial Services Implementation Guide

Publication date: March 04, 2020

Feedback

If you have comments for how we can improve our materials, send an email to AppDocBug@pega.com.

Contents

Overview **6**
Purpose of this guide 6

Preparing for the implementation **7**
Creating the application 7

Building features **9**
Optional: Creating operators 10
Managing your application data with the Integration Designer 11
Implementing case types and attributes 11
Updating case types 12
Adding case types 12
Overriding the application extension data transform 12
Customer journeys and sub-journeys 13
Changing the field values for customer journeys 13
Changing the field values for customer sub-journeys 13
Updating the report definition for the work class 15
Updating the the Skip synchronization requirements map rule 15
Configuring integration and security 15
Modifying the class structure 15
Implementing the security model 16
Implementing the security model and organization structure 16
Defining the taxonomy of the organization 16
Configuring your organization structure 17
Configuring work groups 17
Configuring work queues 17
Configuring work parties 17
Implementing the security model 18
Modifying the user interface 18
Customizing the user experience 18
Reviewing the content of the portals 18
Designing for screen performance 19
Modifying the user interface 19
Determining which part of the user interface to modify 19
Locating the user interface element to change 19
Adding properties to sections 20
Adding sections 20
Designing an application skin and styles 20
Updating the skin 21
Modifying the welcome pack correspondence email 21
Modifying the Welcome Pack email attachments 21
Reports 22
Implementing reporting 22
Configuring business scorecard rules 22
Configuring business map value and decision table rules 23
Configuring dynamic system setting rules 24
Third-party components 25

Automatic detection of components....25
Configuring Markit integration....26
Configuring Equifax integration....26
Configuring AVOX integration....26
Configuring World-Check integration....26
Enhancing case searches....26
Adding the custom property to the report definition....27
Enabling the custom property to appear in searches....27
Including the custom property column in the results....27
Configuring the data indexer....27
Re-indexing the application....28
Data traceability....28
Extending tracked security changes....28
Adding properties to customer risk assessment....29
Adding the new risk factor to the risk profile class....29
Mapping the data to the data transform....29
Creating a declare expression for the new risk factor....30
Creating a scorecard rule for the new risk factor....30
Including the custom property column in the results....30
Configuring the Case summary feature....31
Customizing the color of the progress gauge....31
Configuring the progress percent value....31
Configuring the passed deadline conditions....32
Configuring data categories....32
Modifying related cases....32
Configuring pending requirements....33
Creating a new KYC due diligence case....33
Creating a subclass that contains reusable assets....33
Creating a subclass to contain specialization rules for the new regulations....34
Including the CDOT work types and creating a related data transform....34
Adding work parties for the new Miscellaneous case type....34
Creating a flow diagram for the new due diligence subcase....35
Creating an activity to reroute workbaskets....35
Creating a data transform to populate a list of subcases....35
Reviewing Regulatory due diligence case structure....36
Configuring tax cases for funds....36
Configuring controlling related parties....37
Configuring due diligence cases for funds....37
Configuring event-driven architecture....38
Overriding agents....38
Creating a new event....38
Configuring asynchronous processing....39
Enabling asynchronous processing....40
Enabling individual asynchronous processes....40
Configuring the underlying asynchronous process infrastructure....40
Extending asynchronous processes....41
Asynchronous process extension points....41
Policy profile storage....42
Master profile loading and saving mechanism....42
AML CDD profile categorization....43
AML CDD profile usage....44
Adding a new AML profile to the application....45
Adding the new AML profile to the decision table....45
Optional: Creating a new decision table....45
Adding the new AML profile to KYC types....46
Adding the new AML profile to the mapping rule....46

Configuring eScreening services....47
Enabling eScreening....47
Configuring eScreening server connection attempts....47
Configuring the amount of time that the eScreening data is valid....47
Adding data to the eScreening case....47
Related party operationalization....48
Adding an entity type....48
Optional: Installing the UPlus Web Self-Service Application....48
Case navigation....49
Configuring case navigation....50
Configuring client outreach....52
Extending client outreach case creation....52
Extending active case data management....53
Extending case withdrawal....53
Configuring notifications....54
Extending the list of recipients....54
Extending the class structure....55
Entity types....55
Relationship codes and roles....56
Enabling users to simultaneously work on cases....57
KYC Surgical policy updates....58

Testing a new application....60
Testing your application in the Build environment....60
Testing in the test or production environments....61
Testing in the UAT environment....61

Packaging a new application....62
Merging application changes....62
Packaging an application for migration....62
Importing the packaged application....62

Production maintenance and monitoring....63
Business rule maintenance in the Production environment....63
Application health monitoring....63
Identifying and reporting issues....63

Overview

Pega Client Lifecycle Management for Financial Services accelerates the process of onboarding new customers at financial institutions while managing highly complex regulation with less effort. This application enables those involved in the onboarding process, including relationship managers and sales support, to capture customer and due diligence data more easily and accurately, resulting in the reduced turn-around time while onboarding new customers.

- Purpose of this guide

Purpose of this guide

A project team implementing a Pega application uses this implementation guide to enable and extend application features to meet your business requirements. You must have your application installed before you implement it. For information on installing your application, see the product installation guides on the Pega Client Lifecycle Management for Financial Services product page.

This implementation guide presents tasks in the sequence in which they are commonly performed; however, the sequence varies based on business priorities. Follow the procedures in this guide to complete the tasks for the first minimum lovable product (MLP) release, and then use this guide during iterative releases as you configure and extend more features on top of the MLP release.

Case type backlog

An inventory of the application-provided case types that establishes the project scope by defining the changes and additions that are required for the first MLP release and later releases. For more information, see Define Journeys, Personas, Interfaces.

For more information, see Pega Delivery Methodology.

Preparing for the implementation

To prepare the implementation environment and to create your application, complete the preparation tasks.

- Creating the application

Creating the application

Run the New Application wizard to create your application.

1. To create a new operator ID for running the New Application wizard, complete the following steps:
   a. Log in to Dev Studio by using the operator ID `administrator@pega.com` and the password that you specified for that operator.
   b. Save a copy of the existing `administrator@pega.com` operator, and give it a name that identifies it as an Application Setup operator.
   c. Add the *CLMFS:AppSetup* access group to the new operator record, and then click the radio button to the left of the access group to select it as the default access group.
      When this access group is selected as the default access group for an operator, the New Application wizard opens when the operator logs in.
   d. Save the new Application Setup operator.
2. Log in as the Application Setup operator.
3. Follow the New Application wizard instructions until the **Name your application** page opens, and then follow the steps below.
   For more information about each step of the wizard, see Creating an application.
4. On the **Name your application** page, enter the name of the application, and then click **Advanced configuration**.
5. In the **Organization settings** section, enter the **Organization name**, **Division name**, and **Unit name** for this application.
   The New Application wizard creates the application class structure for you based on the organization settings that you enter. For more information, see Class layers and class hierarchy and inheritance.
   If you have not already defined the organization entities (for example, if you have not already defined the division), type the name of the new entity in the appropriate field. The application saves the new values when you create the new application.
6. Click **Save**.
7. Click **Create application**.
   The Application Wizard creates the implementation application. The application includes one system administrator operator so that you can log into the application after you complete the wizard.
8. To open the new application, click **Go to app**.

The New Application wizard creates a set of access groups for the application. Create your own operators, and then apply the appropriate access groups.

What to do next: After you create your application, enable and extend configurable functionality and features to meet your business needs.

Building features

The implementation team enables or extends features to meet your business requirements.

To configure application features, review the topics in the following sections:

- Optional: Creating operators
- Managing your application data with the Integration Designer
- Implementing case types and attributes
- Overriding the application extension data transform
- Customer journeys and sub-journeys
- Updating the report definition for the work class
- Updating the the Skip synchronization requirements map rule
- Configuring integration and security
- Modifying the class structure
- Implementing the security model
- Modifying the user interface
- Modifying the welcome pack correspondence email
- Reports
- Configuring business scorecard rules
- Configuring business map value and decision table rules
- Configuring dynamic system setting rules
- Third-party components
- Enhancing case searches
- Data traceability
- Adding properties to customer risk assessment
- Configuring the Case summary feature
- Creating a new KYC due diligence case
- Reviewing Regulatory due diligence case structure
- Configuring tax cases for funds
- Configuring controlling related parties
- Configuring due diligence cases for funds
- Configuring event-driven architecture
- Configuring asynchronous processing
- Policy profile storage
- Master profile loading and saving mechanism
- AML CDD profile categorization
- Configuring eScreening services

- Related party operationalization
- Optional: Installing the UPlus Web Self-Service Application
- Case navigation
- Configuring client outreach
- Entity types
- Relationship codes and roles
- Enabling users to simultaneously work on cases
- KYC Surgical policy updates

Optional: Creating operators

If you did not import sample data, you must manually create operators to carry out various tasks.

Note: If you are upgrading an application, you may have already added some of these operators.

1. In the header of Dev Studio, click **Configure** > **Org & Security** > **Organization** > **Operators**.
2. Click **New**.
3. Create an operator in each of the following access groups.

Access Group	Type	Description
CLMFS_API	Runtime	Required for the use of the Client Outreach API. The operator passed with each of the calls to the REST API needs to be configured with this Access Group.
CLMFS_WSS	Runtime	Required for the use of the Client Outreach WSS front end. The Pega Mashup component embedded in the web self-service application will need to be configured to authenticate with a user with this Access Group.
CLMFSSysAdmin	Administration	Administrator user for the base Pega Client Lifecycle Management for Financial Services application. Required to access rDev Studio and Admin Studio.
CLMFSCIBSysAdmin	Administration	Administrator user for the CIB Pega Client Lifecycle Management for Financial Services application. Required

Access Group	Type	Description
		to access Required to access rDev Studio and Admin Studio.
CLMFSRetSysAdmin	Administration	Administrator user for the Retail Pega Client Lifecycle Management for Financial Services application. Required to access rDev Studio and Admin Studio.
CLMFSUnifiedSysAdmin	Administration	Administrator user for the Unified (CIB and Retail) Pega Client Lifecycle Management for Financial Services application. Required to access rDev Studio and Admin Studio.

4. Save your changes.

Managing your application data with the Integration Designer

Use the Integration Designer to manage data at the application level. In the Integration Designer, you can access a comprehensive view of all business objects in your application and their associated data, such as references, data sources, and data views. You can easily create and modify data objects and connect to data sources.

1. In the navigation pane of App Studio, click **Data**.
2. Locate each data type that is marked **SIMULATED**, and click the **FIX NOW** link to configure that data type to connect to real data.
   For more information, see Managing data and integrations with the Integration Designer.

Implementing case types and attributes

Your application includes a set of predefined case types. Each case type contains one or more processes that you can extend to meet your business requirements. If you do not see an existing case type, you can create one during your planning process.

For a list of the case types, see the *Product Overview* on the Pega Client Lifecycle Management for Financial Services product page.

- Updating case types
- Adding case types

Updating case types

If an existing case type is close to meeting your business requirements, you can modify it and use it.

1. In the navigation pane of Dev Studio, click **Case types**.
2. Click the case type that you want to modify.
3. Update the settings for the case type.
4. Update the processes that are associated with the case type.
5. **Optional:** Add processes to the case type.
6. Repeat these steps for any other case types, as necessary.

For more information, see Designing a case.

Adding case types

If you require a case type that is not similar to an existing case type, create a new one that meets your business requirements. If you want to add additional request types to your change request, create them as new case types.

For a list of the case types, see the *Product Overview* on the Pega Client Lifecycle Management for Financial Services product page.

1. In the navigation pane of Dev Studio, click **Case types**.
2. Click **Add a case type**.
3. Enter the **Name** for the new case type.
4. Click **Advanced Settings**.
5. Select the **Derives from (Directed)** class.
6. Select the **Derives from (Pattern)** class.
7. Select the appropriate **Ruleset** and **Ruleset version**.
8. Click **Submit**.
9. Define the case life cycle.
10. Click **Finish**.

Overriding the application extension data transform

In order for the application to work properly, you must update the source classes with the work class you define when the new application was created using the New Application wizard.

1. In Dev Studio, click **App** and search for `PegaFS-Data-AppExtension`.
2. Click **Data Model** > **Data Transform** > **AppExtension_Ext**.
3. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
4. In the **Source** field, change the **PegaCLMFS** or **PegaCLMFS-Work** portion of the source to the name you chose during the DCO process for each case type. For example, `PegaCLMFS-Work-GlobalKYC` becomes `ABCBank-Work-GlobalKYC`.

Customer journeys and sub-journeys

Pega Client Lifecycle Management for Financial Services offers a wide range of customer journeys that enable you to onboard, maintain, review, and offboard existing customers.

You can configure business rules that control which journeys are available for customers. For instance, onboarding journeys should be the only option for first-time customers. Offboarding operations should only be available for fully onboarded and active customers. As the number of options grows, the way to configure a business configuration should not grow in complexity.

Because multiple journeys can be initiated from several different portals, the underlying business logic that controls the various options must be defined consistently throughout the application. This business logic is likely to be customized by using multiple factors, like customer type, business line, or type of business relationship.

For more information about how current customer journeys and sub-journeys are configured, see Customer journeys and sub-journeys.

- Changing the field values for customer journeys
- Changing the field values for customer sub-journeys

Changing the field values for customer journeys

In order for the application to work properly, you must update the field values with the work class you defined during the DCO session.

1. In Dev Studio, search for and select the `CustJourneyType CustomerReview` field value rule in the `PegaCLMFS-Work-CLM` class.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Click **Save**.
4. Repeat steps 1-3 for the following field values.
   - `CustJourneyType MaintainBusinessRelationship`
   - `CustJourneyType MaintainExistingCust`
   - `CustJourneyType Offboardexistingcustomer`
   - `CustJourneyType OnboardNewBusRel`
   - `CustJourneyType OnboardNewCustomer`
   - `CustJourneyType DueDiligence`
   - `CustJourneyType  FullfillClusteredProducts`

Changing the field values for customer sub-journeys

In order for the application to work properly, you must update the field values with the work class you defined during the DCO session.

1. In Dev Studio, search for and select the `CustJourneySubtype CustomerPeriodicReview` field value rule in the `PegaCLMFS-Work-CLM-CustomerReview` class.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.

3. Click **Save**.
4. Repeat steps 1-3 for the following field values.

Field value	Class
CustJourneySubtype CustomerMaterialChange	PegaCLMFS-Work-CLM-CustomerReview
CustJourneySubtype CustomerExpiredRequirement	PegaCLMFS-Work-CLM-CustomerReview
CustJourneySubtype CustomerAmendment	PegaCLMFS-Work-CLM-CustomerReview
CustJourneySubtype NewFundmanager	PegaCLMFS-Work-CLM-OnboardNewCust
CustJourneySubtype NewIndividualEntity	PegaCLMFS-Work-CLM-OnboardNewCust
CustJourneySubtype NewParentEntity	PegaCLMFS-Work-CLM-OnboardNewCust
CustJourneySubtype NewSubsidiaryEntity	PegaCLMFS-Work-CLM-OnboardNewCust
CustJourneySubtype NewFundmanager	PegaCLMFS-Work-CLM-OnboardNewBusRel
CustJourneySubtype ExitProductsOrLocations	PegaCLMFS-Work-CLM-OffBoardExistingCust
CustJourneySubtype OffboardExistingCustomer	PegaCLMFS-Work-CLM-OffBoardExistingCust
CustJourneySubtype EventDrivenReview	PegaCLMFS-Work-CLM-MaintainExistingCust
CustJourneySubtype MaintainCustomerGeneralData	PegaCLMFS-Work-CLM-MaintainExistingCust
CustJourneySubtype AddFund	PegaCLMFS-Work-CLM-MaintainBusRel
CustJourneySubtype AddProductsOrLocations	PegaCLMFS-Work-CLM-MaintainBusRel
CustJourneySubtype FulfillProductsByLocation	PegaCLMFS-Work-CLM-FulfillClusteredProducts
CustJourneySubtype NewPrincipal	PegaCLMFS-Work-CLM-OnboardNewBusRel
CustJourneySubtype CustomerPeriodicReview	PegaCLMFS-Work-CLM-CustomerReview
CustJourneySubtype CustDueDilegence	PegaCLMFS-Work-CLM- CustDueDilegence

Updating the report definition for the work class

During the creation of the application, the system created a new table to store the work of the new implementation. The report definition `pyLoadMyCasesNestedINWP` should be updated to point to the new implementation work class.

1. In Dev Studio, search for and select the `pyLoadMyCasesNestedINWP` report definition rule in the `PegaCLMFS-Work` class.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Click **Save**.

Updating the the Skip synchronization requirements map rule

The map rule `SkipSynchronizeRequirement` must be overridden to the implemation layer to correctly function.

1. In Dev Studio, search for and select the `SkipSynchronizeRequirements` map rule in the `PegaCLMFS-Work` class.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Update the existing work classes in each row with the work classes from the new implementation layer, for example `ABCBank-Work`.
4. Click **Save**.

Configuring integration and security

Configure the integration and security of your application.

1. In Dev Studio, click *Application name* **> Definition**.
2. Click the **Integration and security** tab.
3. In the **Content storage** section, select **Store in CMIS repository**.
4. In the **Connector name** field, enter `PRPC`.
5. In the **Root level folder** name field, enter `/`.
6. In the **Content storage** section, select **Store in Pega database (default)**.
7. Click **Save**.

Modifying the class structure

Pega Client Lifecycle Management for Financial Services automatically creates your class structure. When you modify the structure, follow the correct naming conventions. For more information, see Classes.

Implementing the security model

Security planning involves defining authorization and authentication strategies for your application.

- Implementing the security model and organization structure
- Implementing the security model

Implementing the security model and organization structure

Define the authorization and authentication strategies for your application.

Authentication

Proves to the application that you are who you say you are.

Authorization

Determines the functions that you can perform in the application. This corresponds to an access group and role configuration.

Security planning also involves setting up the organization structure and operator attributes. The application provides security in the form of access settings and denial rules. Many integration rules also incorporate authentication.

For more information about the additional aspects of security, enroll in the *Lead System Architect* course on Pega Academy and cover the Security lessons that correspond to the following topics:

- Defining the taxonomy of the organization
- Configuring your organization structure
- Configuring work groups
- Configuring work queues
- Configuring work parties

Defining the taxonomy of the organization

Use the Taxonomy tab to set up the blueprint for your organizational chart. You can use the UPlus Financial Services taxonomy chart as an example. The taxonomy chart defines the types of entities that appear in your organizational chart.

The following are default levels of the taxonomy:

- Financial Institution - the top-level entity. You cannot modify this
- Business Line - type of business, for example corporate investment banking and retail banking
- Country - the country in which the business line exists
- Booking Entity - balance sheet level
- A bottom level

To define your taxonomy, do the following steps.

1. Click **Configure** > **Financial Services** > **Operating Structure**, and select the **Taxonomy** tab.
2. In the **View operational structure for** field, select **UPlus Financial Services** and click **Submit**. Review the simulated taxonomy as an example of how to set up your taxonomy.
3. Click **New**.

4. In the **Name** field, enter a name for the bank. The ID field is automatically populated with a 6-character ID.
5. In the **Top level class** field, enter the class name, for example, `Data-`.
6. Modify the **Calendar** and **Currency** fields if necessary.
7. Click **Submit**.
8. In the new taxonomy, click **Specialize** and select the type of business line to create.
9. Click **Add** to assign a new operational level to the business line.
10. Enter a name for the new level and select the level type.
11. See Defining the organization structure to build out your organization.

Important: It is important to note that the you should define and complete the taxonomy before the organization chart. Once the definition of the organization chart is started, you cannot change the taxonomy.

Configuring your organization structure

Use the organization structure for routing and reporting within the application. Typically, the application organization structure does not map operators exactly to the site's organization chart but instead, it maps the work that those operators do.

Tip: For design guidance, see Organization landing page.

1. In the header of Dev Studio, click **Configure** > **Org & Security** > **Organization** > **Organizational Chart**.
2. Review the existing structure.
3. Determine the organization, division, and unit levels of the hierarchy.

Configuring work groups

A work group determines which work queues you can access.

1. In the header of Dev Studio, click **Configure** > **Org & Security** > **Tools** > **Work Groups**.
2. Review and modify the list if necessary.

   For more information, see Teams.

Configuring work queues

A work queue is a queue of open assignments in the application.

1. In the header of Dev Studio, click **Configure** > **Org & Security** > **Tools** > **Work Queues** to display a list of work queues.
2. Review and modify the list if necessary.

   For more information, see Initial Work queues.

Configuring work parties

A work party represents a person, business, or organization that is involved in a case. It receives correspondence, such as email, and can be an active or passive participant based on its role. Pega Client

Lifecycle Management for Financial Services comes with default work parties, but you might need to configure them for site-specific requirements.

1. In the **Case types** explorer, click the name of the case of which you want to modify the work parties.
2. Click the **Settings** tab and select **Parties**.
3. Click the work party name to open the settings for the work party.
4. Make your modifications and click **OK**.

What to do next:

For more information, see Defining case participants.

Implementing the security model

After you review the existing groups and roles to determine additional groups and roles that you need, create them in Dev Studio when logged in as an administrator.

- For Access groups, click **Configure** > **Application** > **Structure** > **Access Groups and Users**.
- For Access roles, click **Configure** > **Org & Security** > **Tools** > **Security** > **Role Names**.

For more information, see Groups and related topics.

Modifying the user interface

While Pega Client Lifecycle Management for Financial Services is fully functional immediately after installation, you may change portions of the user interface (UI) to meet the needs of your business and users.

- Customizing the user experience
- Modifying the user interface

Customizing the user experience

While Pega Client Lifecycle Management for Financial Services is fully functional immediately after installation, you might change portions of the user experience (UX) to meet the needs of the users at your site.

- Reviewing the content of the portals
- Designing for screen performance

Reviewing the content of the portals

Review the content of your application's portals to ensure that only relevant data is presented.

Review each of the portal sections.

Designing for screen performance

You want your application to respond immediately as your end users interact with customers. As you design your user interface, complete the following steps:

1. Review transactional service level agreements.

   For example: The customer details must be available in one second or less while a representative is on the phone with the customer.
2. Review service performance, such as how long it takes back end systems to gather the data needed to display customer details.
3. Review data requirements to determine which fields contain the information that the representatives absolutely need for the task that they need to perform.
4. Determine if any network configuration could cause latency.

   For example: Representatives who are logged in through a corporate VPN or in a remote location might cause an issue.

Modifying the user interface

You can change portions of the user interface to meet the needs of your business and users.

- Determining which part of the user interface to modify
- Locating the user interface element to change
- Adding properties to sections
- Adding sections
- Designing an application skin and styles
- Updating the skin

Determining which part of the user interface to modify

You can customize the content of the dashboard so that it displays any data that you integrate into your application.

Locating the user interface element to change

You can customize the content of the dashboard so that it displays any data that you integrate into your application. To determine which user interface element to modify to implement your change, use the Live UI feature of the Pega Platform.

1. In Dev Studio, launch the portal that you want to modify.
2. Click the **Open runtime toolbar** in the bottom right corner.
3. Click the **Live UI** icon.
4. Select an element to observe and click the **Display info panel for this item** icon.
5. To update the rule, click the **Open rule in Dev Studio** icon and make your changes.

Adding properties to sections

Pega Client Lifecycle Management for Financial Services uses standard Pega Platform user interface components, so the update process is the same as any Pega Platform application.

1. In Dev Studio, launch the portal that you want to modify.
2. Click the **Open runtime toolbar** in the bottom right corner.
3. Click the **Live UI** icon.
4. Select the section that you want to modify.
5. To open the rule in Dev Studio, click the **Open rule in Dev Studio** icon.
6. In the navigation panel of Dev Studio, click **App**.
7. Locate the property that you want to add to the section and drag it into that section.
8. Click **Save as** and then save the ruleset to your application.

Adding sections

Pega Client Lifecycle Management for Financial Services uses standard Pega Platform user interface components, so whether you are updating the dashboard, portals, or the composite, you do so in the same way as any Pega Platform application.

Pega Client Lifecycle Management for Financial Services uses standard Pega Platform user interface components. Whether you are updating the dashboard, Patient 360 view, or the portals, you follow the same procedure as you do for any Pega Platform application

1. In Dev Studio, launch the portal that you want to modify.
2. Click the **Open runtime toolbar** in the bottom right corner.
3. Click the **Live UI** icon.
4. Locate the area where you want to add the section.
5. In the navigation panel of Dev Studio, click **App**.
6. Locate the section that you want to add and drag it into your specified section.
7. Click **Save as** to save the rule to the ruleset for your application.

Designing an application skin and styles

Your site very likely has user experience (UX) standards for the appearance of your applications. Your task is to determine which styles have to change to adhere to those standards.

1. In the header of Dev Studio, click **Configure** > **User Interface** > **Skins, interfaces, & templates**.
2. On the **Skins** tab, double-click the skin.

   Note: The **CLM** skin is based on the `pyEndUser` skin.

3. Click the **Component styles** tab.
4. Review each component.

   Note: As you plan your application styles, consider using mixins to provide incremental style changes. For more information, see Using mixins in the skin to drive consistency and efficiency.

Updating the skin

You can update the look and feel of Pega Client Lifecycle Management for Financial Services to reflect the color scheme of your organization. The application uses the standard Pega Platform skinning features in your application.

For more information about skins, see Creating a skin and Skin inheritance.

1. In Dev Studio, click *Application Name* > **Skin**.
2. Update the options as needed, and then click **Save as**.
3. Click **Create and open**.
4. If you give your skin a different name, update the application to reflect it by performing the following steps:
   a. In Dev Studio, click *Application Name* > **Definition**.
   b. Change the value in the **Skin** field to the name of the skin that you created.
5. Click **Save**.

Modifying the welcome pack correspondence email

A retail customer receives a welcome pack email with attached PDFs about their account, including a Privacy Disclosure Notice, Personal Account Agreement, welcome letter, and Bank Signature Card letter. Modify the content of the email to best fit your organization.

1. Log in to Dev Studio using the following credentials:

   User Name: `clmfsretsysadmin`

   Password: The password you specified when you enabled this user.
2. In the Dev Studio header search text field, enter `NotifyCustomerProduceWelcomePack`, and then select the correspondence rule from the results.
3. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
4. Review the template correspondence and modify it to contain information specific to your organization.
5. Click **Save**.
6. Modify the eForm files by following the steps in the Modifying the Welcome Pack correspondence email task.

- Modifying the Welcome Pack email attachments

Modifying the Welcome Pack email attachments

A retail customer receives a welcome pack email with attached PDFs about their account, including a Privacy Disclosure Notice, Personal Account Agreement, welcome letter, and Bank Signature Card letter. Modify the content of the attachments to best fit your customer's needs.

1. Log in to Dev Studio using the following credentials:

   User Name: `clmfsretsysadmin`

   Password: `install`

2. In the Dev Studio header search text field, enter `PrivacyDisclosureNotice`, `PersonalAccountAgreement`, `UPlus Bank Signature Card`, or `UPlus Welcome Letter` and then select the eForm File rule from the results.
3. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
4. Click **Download file**.
5. Review the template correspondence and modify it, if needed.
6. Upload the modified file.

Reports

In addition to the library of Pega Platform reports, Pega Client Lifecycle Management for Financial Services provides several pre-configured reports that are customized for the various work objects, cases, and business processes used in the application. These reports provide you with real-time insights into case inventories and aging so that you can respond with the agility that is needed to comply with your internal and external service level agreements as well as compliance mandates.

To create other reports, see Reporting.

- Implementing reporting

Implementing reporting

Based on your login and portal, your application provides numerous pre-configured reports to monitor case and work inventory as well as monitor workforce productivity and quality. These standard reports are driven by the case and work object life-cycle infrastructure as well as the work and assignment-level service level rules for goals and deadlines used in managing the life cycle of a case.

Based on your login and portal, Pega Client Lifecycle Management for Financial Services provides numerous pre-configured reports to monitor case and work inventory as well as monitor workforce productivity and quality. These standard reports are driven by the case and work object life-cycle infrastructure as well as the work and assignment-level service level rules for goals and deadlines that are used in managing the life cycle of a case. For additional information, see the *Pega Client Lifecycle Management for Financial Services Business Use Case Guide* on the Pega Client Lifecycle Management for Financial Services product page.

Configuring business scorecard rules

Use default scorecard rules to calculate risk based on customers, related products, business codes, and business relations.

If you need to edit these rules do the following:

1. Search for the scorecard rule.
2. In the search results window, click the name of the scorecard. See the Business scorecard rules list below.
3. Edit the rules if necessary.
4. Click **Save as**.
5. Select your Context values.

6. Click **Create and open** to save the rule to your ruleset.

RelatedPartiesRisk

Calculates the risk based on related parties

DurationOfRelationship

Calculates the risk based on duration of business relationship

BusinessCodeRisk

Calculates risk based on business code

CountryRelatedRisk

Calculates the risk based on countries in which business is carried out

ProductRelatedRisk

Calculates the risk based on related products

CustomerRiskAssessmentForOrg

Calculates the aggregated risk for a commercial banking customer based on all risk factors

CustomerRiskAssessmentForInd

Calculates the aggregated risk for an individual customer based on all risk factors

CustomerRiskAssessmentForFund

Calculates the aggregated risk for a fund customer based on all risk factors

BusinessCode

Calculates the business score value based on the business code

InitialWorkUrgency

Generic scorecard rule to assess initial urgency based on several customer and work-related factors

Configuring business map value and decision table rules

You can use default map value rules to calculate prioritization weights based on factors including investor outlooks, customer segments, and product matrices.

If you need to edit these rules do the following:

1. Search for the map value rule.
2. In the search results window, click the name of the map value. See the Business map value rules list below.
3. Edit the rules if necessary.
4. Click **Save as**.
5. Select your Context values.
6. Click **Create and open** to save the rule to your ruleset.

Configuring dynamic system setting rules

Dynamic system settings control the behavior of your application.

If you need to edit these rules do the following:

1. Search for the dynamic system setting rule.
2. In the search results window, click the name of the map value. See the Dynamic system setting rules list below.
3. Edit the rules if necessary.
4. Click **Save as**.
5. Click **Create and open** to save the rule to your ruleset.

Dynamic system setting rules

EnableSpinoffFulfillment

Determines the permission to allow configuration of product fulfillment in parallel or not

bSpinoffFulfillment

Determines whether the functionality for product fulfillment in parallel is enabled or disabled by default

SpinoffFulfillmentBy

Indicates which parallel product fulfillment approach to set by default

OECDResidenceCountry

Indicates whether the country of legal residence of the CLM application hosting financial institution is a member of the OECD list

SkipEmptyKYCAssignment

Determines whether manual assignments for Pega Know Your Customer data collection tasks are required where there is no KYC Type applied or all the ones applied are complete

EnableAsynchronousProcess

Enables or disables asynchronous processing in some parts of the CLM application

DisableCRSFATCAOverrideLink

Enables or disables the possibility for the tax reviewer to override the result of the tax (FATCA or CRS) assessment using a questionnaire

SpinoffTaxCasesForFunds

Enables or disables the functionality for creating tax (FATCA and CRS) subcases for funds in groups

yrsAtPrimaryAddress (Retail)

Threshold to trigger prompting for a prior address

OnlineBranch (Retail)

Indicates the online retail branch for the organization

LicensedToUseKYC

Indicates that the Pega Know Your Customer engine is enabled (licensed separately)

Important: The default setting for the `LicensedToUseKYC` rule is **false**. You must set it to **true** in order to enable the Pega Know Your Customer engine. For information about implementing

Know Your Customer, see the Know Your Customer Implementation Guide on the Know Your Customer product landing page.

Third-party components

Pega Client Lifecycle Management for Financial Services uses third-party components to enrich customer data. These components include MarkIt, Equifax (including eID Verifier), AVOX, and Work-Check.

These components came standard with Pega Foundation for Financial Services but, from version 8.3 onwards, these components are optional. Customers that want to use them can download them from Pega Exchange and include them in the application rule of their implementation layers. With this new approach, customers can use the third-party services as plug-and-play components. When the components are added to the application, it detects them in the application stack. For more information, see Automatic detection of components.

- Automatic detection of components
- Configuring Markit integration
- Configuring Equifax integration
- Configuring AVOX integration
- Configuring World-Check integration

Automatic detection of components

By default, Pega Client Lifecycle Management for Financial Services provides the connection to the component. If your application needs to directly connect to any of them, you must use the automatic detection mechanism to avoid issues when the components are not in the application stack.

The system detects the presence of components in the stack using when rules. Pega Foundation for Financial Services is required in order to use the components. Each component has two associated when rules, one at Pega Foundation for Financial Services level and an override of that rule at the component level. The rule in the component layer is set to return a boolean value `true` (the component is present), whereas the rule present in the Pega Foundation for Financial Services layer returns a boolean value `false` (the component is not present).

When a component is linked to the application, its associated rulesets are put directly above the Pega Foundation for Financial Services rulesets for rule resolution. When the system detects the component, it selects the when rule at the component, therefore confirming its presence. If the component is not in the stack, the system selects the when rule (with a value of `false`) at the Pega Foundation for Financial Services layer and the application will not use the component services.

Customers with applications that have logic strongly coupled to the rules of the components should make use of the rules below. They should invoke them before invoking to the component-related logic to avoid issues at run time due to the absence of the component in the stack.

```
@baseclass IsWorldCheckOneComponentPresent
@baseclass IsMarkitComponentPresent
@baseclass IsEquifaxComponentPresent
@baseclass IseIDComponentPresent
@baseclass IsAvoxComponentPresent
```

Configuring Markit integration

If you are licensing Markit, you must enter your Markit-provided password in the Authentication Profile.

1. Log in to the Financial Services Industry Foundation application using administrator credentials.
2. In the **Records** pane, click **Security** > **Authentication Profile**.
3. Update the password for `GMServiceAuthentication` to the one provided to you by Markit.
4. Click **Save**.
5. Review all the dynamic system settings associated with the `PegaGMInt` ruleset and ensure that they have appropriate values.

Configuring Equifax integration

Instructions for configuring Equifax can be found on the Foundation for Financial Services product page, in the *Equifax Connector Guide*.

Configuring AVOX integration

In order to access AVOX services, the outbound IP of your servers should be whitelisted by AVOX. Please, discuss with your IT department and AVOX how to implement these changes.

After the configuration is completed, review of all the dynamic system settings associated with the `PegaAVOX` ruleset and ensure that they have appropriate values.

Configuring World-Check integration

Instructions for configuring the Thomson Reuters World-Check integration can be found on Pega Exchange, where both the distribution files and an installation guide are available. Once the component is installed, ensure that the `LicensedToUseEScreening` dynamic system setting is set to **true**. See Enabling eScreening.

Enhancing case searches

Onboarding activities can result in the creation of large volumes of work which are commonly carried out by multiple groups within the organization.

You can use the Relationship Manager Portal to perform searches on cases in your environment. You can add fields to the search results to help refine your search.

For information about how case searches are implemented, see Case searches in Client Lifecycle Management for Financial Services.

To extend the functionality, complete the following steps.

1. Adding the custom property to the report definition
2. Enabling the custom property to appear in searches
3. Including the custom property column in the results
4. Configuring the data indexer

5. Re-indexing the application

Adding the custom property to the report definition

Add the custom property to the report definition.

1. In the Dev Studio header search text field, enter `pyWorkSearch` and select the `Work-` report definition rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Click **Add column**, and type the name of a custom property that you want to add, for example, `pyCustomerName`.
4. Click **Save**.

Enabling the custom property to appear in searches

Enable the custom property to be included in search results.

1. In the Dev Studio header search text field, enter `pysearch` and select the `Work-` custom search properties rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Select the custom property that you added, for instance **pyCustomerName**, to include it in the search results.
4. Click **Save**.

Including the custom property column in the results

Include the custom property column in the results grid in section `pyWorkSearchResults` as required to show in the search results screen.

1. In the Dev Studio header search text field, enter `pyWorkSearchResults` and select the section rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Add columns for your custom property in the grid layouts. For additional information about grids, see Harness and Section forms - Adding a Grid layout.
4. Click **Save**.

Configuring the data indexer

Configure the data indexer.

1. In the Dev Studio header search text field, enter `useDataInstances` and select the `Pega-Searchengine Indexing/usedatainstances` dynamic system settings rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Type `true` in the **Value** field.
4. Click **Save**.

Re-indexing the application

You must re-index all work in order to see the new search fields.

1. In Dev Studio, click **Configure** > **System** > **Settings** > **Search**.
2. Click **Re-index** for All work. For more information about re-indexing, see Rebuilding a search index.

Data traceability

Financial Services institutions are subject to a number of regulations that vary between jurisdictions. In addition to meeting these regulations across the globe, institutions need to prove to regulatory auditors how specific decisions were made. A large volume of data that drives those decisions is captured from a variety of sources, such as internal databases, customer self-service, manual entry by employees, and third party systems. It is important for the business to track where the data was initially captured and how it changes over time.

Use the data traceability feature to identify and track data objects and then configure auditable entries on a particular data object. The data change tracking engine will scan for changes and save them in an exposed, easily accessible data change repository.

To extend data traceability, do the Extending tracked security changes task.

For information about how data traceability is configured, see Data traceability.

- Extending tracked security changes

Extending tracked security changes

Configure auditable entries on a particular data object that you want to track, for example, business goals.

1. In the header of Dev Studio, in the search text field, enter `FSIFTrackSecurityChanges` and select the `PegaFS-Data-Party-MasterProfile` data transform rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Click **Add a row** ⊕.
4. Enter `RelevantRelationshipList.LinkedChildren` for both the **Target** and **Source** fields.
5. Click **Save**.
6. In the header of Dev Studio,, in the search text field, enter `FSIFTrackSecurityChanges` and select the `PegaFS-Data-PartyPartyXref` data transform rule.
7. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
8. Click **Add a row** ⊕.
9. Enter `LinkedChildren` for both the **Target** and **Source** fields.
10. Click **Save**.

Adding properties to customer risk assessment

A risk profile is a qualitative and quantitative analysis of the types of threats an organization, asset, project, or individual faces. The goal of a risk profile is to provide a non-subjective understanding of risk by assigning numerical values to variables representing different types of threats and the danger they pose.

Scorecard logic

A scorecard contains the weighted values for each risk. You can create a scorecard rule to calculate, for example, customer segmentation based on age and income and then map particular score ranges to defined results.

Implementation details

For information about existing risk factors see Customer risk assessment.

To add new risk factors to your risk profile or edit the existing ones, do the following tasks:

1. Adding the new risk factor to the risk profile class
2. Mapping the data to the data transform
3. Creating a declare expression for the new risk factor
4. Creating a scorecard rule for the new risk factor
5. Including the custom property column in the results

Adding the new risk factor to the risk profile class

Add the new risk factor to the risk profile class.

1. In Dev Studio, click **Data types** > **Risk profile**.
2. Click **Add field**.
3. Enter a name and ID and select the field type.
4. Click **Save**.

Mapping the data to the data transform

The risk engine is triggered when changes are made to certain properties. If you extend the risk engine with additional properties, do the following steps.

1. In the header of Dev Studio, enter `SynchronizeRiskProfile` and select the `PegaFS-Work` data transform rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Click **Add a row** ⊕.
4. Enter the **Target** and **Source** values from the property you created in Adding the new risk factor to the risk profile class.
5. Click **Save**.

Creating a declare expression for the new risk factor

Create a declare expression for the new risk factor.

1. In Dev Studio click **Records** > **Decision** > **Declare Expression**.
2. Right-click **Declare Expression** and click **Create**.
3. Type a name in the **Label** field.
4. Enter the **Target Property** values from the property that you created in Adding the new risk factor to the risk profile class.
5. Enter a value in the **Page Context** field.
6. Enter the relevant class in the **Apply to** field.
7. From the **Add to ruleset** menu, select a ruleset.
8. Click **Create and open**.
9. Open the new declare expression and add logic to it by clicking **Records** > **Decision** > **Declare Expression** and selecting it.
10. Click **Save**.

Creating a scorecard rule for the new risk factor

Create a scorecard rule for the new risk factor.

1. In Dev Studio click **Records** > **Decision** > **Scorecard**.
2. Right-click **Scorecard** and click **Create**.
3. Enter a name in the **Label** field.
4. Enter a relevant value in the **Apply to** field.
5. From the **Add to ruleset** menu, select a ruleset.
6. Click **Create and open**.
7. Open the new scorecard rule and add logic to it by clicking **Records** > **Decision** > **Scorecard** and selecting it.
8. Click **Save**.

Including the custom property column in the results

Map the scorecard rule you created in Creating a scorecard rule for the new risk factor to either the individual or organization risk scorecard.

1. In the header of Dev Studio, search for either `CustomerRiskAssessmentForInd` or `CustomerRiskAssessmentForOrg` and select the scorecard.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Click **Add a row** and add the necessary logic.
4. Click **Save**.

Configuring the Case summary feature

The Case summary feature contains a detailed view of the stages, steps, and the most relevant data for a case across its lifecycle.

The process of onboarding a new customer can vary depending on factors, such as the customer type, their location, and the products they use. Increasing regulation in the financial services industry adds pressure to understand the progress of ongoing work, related information, and those main parties involved in onboarding activities. This context-specific view of the overall parent case or separate units of related work must provide relevant data to the user to help them continue their task or understand blockers to progress.

For more information about how the Case summary feature is configured, see Case summary.

To configure this feature, do the following tasks.

1. Customizing the color of the progress gauge
2. Configuring the progress percent value
3. Configuring the passed deadline conditions
4. Configuring data categories
5. Modifying related cases
6. Configuring pending requirements

Customizing the color of the progress gauge

The progress gadget shows the percentage of completion of the case and the status. By default, the indicator is red if the task is delayed and green if the task in on track. If you want to modify the colors, do the following steps.

1. In the Dev Studio header search text field, enter `simple-percentage-chart` and select the HTML or CSS file.
2. Save both rules into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. In the CSS file, change the color properties to a color of your choice.
4. Click **Save**.

Configuring the progress percent value

The progress gadget shows the percentage of completion of the case and the status. You can configure at which percentage of completion a case is considered delayed.

1. In the Dev Studio header search text field, enter `CalculateNavigationProgress` and select the activity rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Modify the **Calculation if case not complete** method parameters.
4. Click **Save**.

Configuring the passed deadline conditions

The progress gadget shows the percentage of completion of the case and the status. You can configure the conditions for when the deadline for completion has passed.

1. In the Dev Studio header search text field, enter `SLADeadlineIsPast` and select the when rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Select the **Advanced** tab and modify the conditions of the when rule, as needed.
4. Click **Save**.

Configuring data categories

You can configure which data categories appear on the Case summary. To modify these categories, edit the data transform and when rules for the categories.

1. In the Dev Studio header search text field, enter `SetAllDataStatusValue` and select the data transform rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Modify the action and the associated when rule, as necessary.
4. Click **Save**.

Modifying related cases

If you are using parallel fulfillment, the system creates the related sibling fulfillment cases for a jurisdiction when all due diligence activities for that jurisdiction are completed.

The related cases are displayed based on the `ObjectType` property which equals `pzinskey` from the main Pega Client Lifecycle Management for Financial Services case.

If you want to change the condition on which related cases should be visible, modify the data page and report definitions.

1. In the Dev Studio header search text field, enter `D_SiblingRelatedCasesByObjectTypeLink` and select the data page rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Modify the data page rule, as necessary.
4. Click **Save**.
5. Open the `RelatedCasesByObjectType` report definition and modify the properties, as necessary.
6. Click **Save**.
7. If you want to change the related cases display properties, open the `RelatedCaseUnit` section rule and modify the section properties.

Configuring pending requirements

At the end of the Capture stage the system creates a unique requirement case for every requirement. You can view a list of pending requirements in the bottom-right corner of the Case summary screen from the Enrich stage onwards.

Depending on the conditions configured in the requirement set rule, the respective requirements are only applicable to the customer.

If you want to update the applicability of the requirement, do the following steps.

1. In the Dev Studio header search text field, enter `OnboardOrganizationBasic` and select the requirement set rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Modify the requirements, as needed.
4. Click **Save**.
5. If you want to apply new requirements for any customer, create appropriate requirement rules in the respective requirement set rule.

Creating a new KYC due diligence case

An important aspect of onboarding a new customer is assessing their trustworthiness. Pega Client Lifecycle Management for Financial Services provides several cases that are designed to collect due diligence information.

If you have additional information that you need to collect, you can create a new subcase based on existing KYC due diligence subcases. Do the following tasks to add a new subcase to the KYC due diligence case. The following tasks serve as an example of how you can add a new due diligence section, Miscellaneous, to the Due Diligence stage. The example section that you add contains a single subcase for collecting Crown Dependencies and Overseas Territories (CDOT) regulation information.

For more information about how the existing case is implemented, see Due diligence case creation.

To build a new KYC due diligence case, do the following steps.

1. Creating a subclass that contains reusable assets
2. Creating a subclass to contain specialization rules for the new regulations
3. Including the CDOT work types and creating a related data transform
4. Adding work parties for the new Miscellaneous case type
5. Creating a flow diagram for the new due diligence subcase
6. Creating an activity to reroute workbaskets
7. Creating a data transform to populate a list of subcases

Creating a subclass that contains reusable assets

Create a `PegaCLMFS-Work-Misc` subclass to contain reusable assets associated with the new subcase.

1. In Dev Studio, in the **Explorer** panel, click **App** > **SysAdmin**.
2. Right-click **Class** and select **+Create**.
3. Enter a label.

4. In the **Class Name**, enter `PegaCLMFS-Work-Misc`.
5. Click **Create and open**.
6. Click **Save**.

Creating a subclass to contain specialization rules for the new regulations

Create a subclass named `PegaCLMFS-Work-Misc-CDOT` to contain the specialization for the Crown Dependencies and Overseas Territories (CDOT) regulations.

1. In Dev Studio, in the **Explorer** panel, click **App > SysAdmin**.
2. Right-click **Class** and select **+Create**.
3. Enter a label.
4. In the **Class Name**, enter `PegaCLMFS-Work-Misc-CDOT`.
5. Click **Create and open**.
6. Click **Save**.

Including the CDOT work types and creating a related data transform

You must include work types for the Crown Dependencies and Overseas Territories (CDOT) case type that you created. You can do this by adding the subcase to the parent case in the case type rule. On the case type rule form, add a data propagation data transform. The data transform will propagate data from the child to the parent class.

1. In the Dev Studio header search text field, enter `pyDefault` and select the `PegaCLMFS-Work-CLM` case type rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. In the **Child case types** section, click **Add a row** ⊕.
4. Enter `PegaCLMFS-Work-Misc` as the name.
5. Click the **Data propagation** for the `PegaCLMFS-Work-Misc` case type.
6. Click **Submit**.
7. Click **Save**.

Adding work parties for the new Miscellaneous case type

The new Miscellaneous case type requires new case-specific work parties. Add new work parties to the `pyCaseManagementDefault` work parties rule.

1. In the Dev Studio header search text field, enter `pyCaseManagementDefault` and select the `PegaCLMFS-Work-CLM` case type rule.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. In the **List of valid parties** section, click **Add a row** ⊕ for each new work party that you want to add.

4. Enter names for the new work parties, for example, `Misc`, `MiscManager`, and `MiscReview`.
5. Click **Save**.

Creating a flow diagram for the new due diligence subcase

You must apply some flow logic to your new subcase. In this example, you will create a flow that prevents the subcase from completing until the parent KYC cases are complete.

1. In Dev Studio, in the **Explorer** panel, click **App** > **CLM** > **Process**.
2. Right-click **Flow** and select **+Create**.
3. Enter, for example, `Misc` as the flow name.
4. Add decision shapes in the flow that prevent the subcase from completing until the parent KYC cases are complete.
5. Include the `Misc` flow in Due Diligence stage and create when rules and service rules to check when the `Misc` flow has to be initiated.

Note: Click **App** > **CLM** > **Process** > **Flow** and look at the `KYC`, `Legal`, or `Tax` due diligence flows to get ideas for creating your flow.

Creating an activity to reroute workbaskets

Create an activity that checks for a valid work party role. If the party role is available on the case then the activity checks for the default workbasket that is associated with that work party and assigns the assignment to that work party. If the work party role is not available on the case then the case is routed to `routing_error@clmfs` and an error message is sent.

For example, if the SSManager role is passed as a parameter, then the activity rule checks the pyWorkParty(SSManager) property rule for a routing workbasket name and assigns the assignment accordingly. If pyWorkParty(SSManager) is not available, then the assignment gets routed to `routing_error@clmfs`.

1. In the Dev Studio header search text field, enter `ToCasePartyWB` and select the `PegaCLMFS-Work-Tax` activity rule.
2. Click **Save as** to copy the rule.
3. In the **Applies to** field, enter `PegaCLMFS-Work-Misc`.
4. Click **Create and open**.
5. Edit the conditions for the when rules, as necessary.
6. Click **Save**.

Creating a data transform to populate a list of subcases

Create a data transform that will populate a list of subcases that are initiated from your new due diligence subcase.

1. In the Dev Studio header search text field, enter `PopulateTaxCasesList` and select the `PegaCLMFS-Work-CLM` data transform rule.
2. Click **Save as** to copy the rule.
3. In the **Label** field, enter `PopulateMiscCasesList`.

4. Click **Create and open**.
5. Edit the actions, as necessary.
6. Click **Save**.

Reviewing Regulatory due diligence case structure

The Regulatory due diligence sub-cases that are required according to the different jurisdictions, products types and regulations are initiated in parallel with Global KYC stage when the Due diligence stage is started. Because cases run in parallel to Global KYC, they may be resolved before the Global KYC stage completes.

Do the following task to review due diligence case structure and makes edits, as necessary.

1. In Dev Studio click **Records** > **Data Model** > **Data Transform**.
2. Search for and select one of the following data transform rules and edit the rule, as necessary.

PropagateDataToLocalKYCRegulatory

The case type includes *PegaCLMFS-Work-KYC* as one of the child cases of the parent CIB case. The *PropagateDataToLocalKYCRegulatory* data transform is used for propagating the data required during the creation of Regulatory cases.

PopulateLocalCasesList

The *PopulateLocalCasesList* data transform is used for preparing the data based on which local and regulatory cases get created. The data transform accepts the *LKYCType* parameter. Based on its value (AML/Regulatory), the system invokes either the logic for creation of AML Local DD cases or to Regulatory Local DD cases.

PopulateLocalCasesListData

The *PopulateLocalCasesListData* data transform is used to create regulatory cases for organization and individual customers.

PopulateLocalCasesListDataForFunds

The *PopulateLocalCasesListDataForFunds* data transform is used to create regulatory cases for funds. This data transform in turn invokes the *CreateLocalCasesListForFunds_Unique* and *CreateLocalCasesListForFunds_Inherited* data transforms which contain logic for the creation of funds based on the *clmfs/SpinoffRegulatoryDDCasesForFunds* dynamic system setting.

What to do next:

Note: The creation of regulatory cases is specialized for funds. Fund regulatory cases are created based on the value of *clmfs/SpinoffRegulatoryDDCasesForFunds*. For more information, see Configuring due diligence cases for funds.

Configuring tax cases for funds

The *clmfs/SpinoffTaxCasesForFunds* dynamic system setting enables you to control which tax cases get created while managing fund managers and funds.

1. In Dev Studio click **Records** > **SysAdmin** > **Dynamic System Settings**.
2. Search for and select the *clmfs/SpinoffTaxCasesForFunds* dynamic system setting.
3. In the **Value** field, enter a value. You have the following options.
   - **Unique**: Tax cases (FACTA/CRS) will be created for fund manager and each fund selected for that particular journey.
   - **Inherited**: Tax cases (FACTA/CRS) will be created only for the fund manager.
   - **If is valid when rule**: Customize the creation by specifying a when rule, for example, *TaxCaseCreationWhenRuleEvaluation*.

   Note: The default is **Unique** if the result of the when rule is true and **Inherited** if result of the when rule is false.

Configuring controlling related parties

Some related parties are marked as "deemed controlling" based on an automated assessment. Those flagged as 'deemed controlling' will be considered during the evaluation of relevant parties, and will determine whether Global due diligence cases are created.

The automated assessment is done based on the *PegaFS-Data-PartypartyXref.RelevantPartyDecision* rule, which is invoked while adding a related party. Based on the assessment, the *IsDeemedControlling* property is either set to `true` or `false`. The property is persisted in the *pp_IsDeemedControlling* column of the *fsf_partypartyxref* database table during the customer data synchronization.

While the preconfigured logic for assessing this flag typically covers most regular cases, there might be some situations where the Relationship Manager might need to override the automated assessment. To set the *IsDeemedControlling* to `true` or `false` do the following steps.

1. In Dev Studio click **Records** > **Decision** > **Decision Table**.
2. Search for and select the *RelevantPartyDecision* decision table.
3. Set the *IsDeemedControlling* property to `false`.

Configuring due diligence cases for funds

The *clmfs/SpinoffRegulatoryDDCasesForFunds* dynamic system setting enables you to control which due diligence cases get created while managing fund managers and funds.

1. In Dev Studio click **Records** > **SysAdmin** > **Dynamic System Settings**.
2. Search for and select the *clmfs/SpinoffRegulatoryDDCasesForFunds* dynamic system setting.
3. In the **Value** field, enter a value. You have the following options.
   - **Unique**: Regulatory cases (LKYC) will be created for each fund selected for that particular journey.
   - **Inherited**: Regulatory cases (LKYC) will be created only for the fund manager.
   - **If is valid when rule**: Customize the creation by specifying a when rule, for example, *TaxCaseCreationWhenRuleEvaluation*.

   Note: The default is **Unique** if the result of the when rule is true and **Inherited** if result of the when rule is false.

Configuring event-driven architecture

As part of doing business with customers, a wide range of potential events can occur from many different internal and external sources. A financial institution must be able to identify and process a change, such as a customer updating their address or an identification document expiring. Failing to do so can lead to non-compliance and increased risk. Forcing manual checks on every change can delay or suspend revenue generation for the financial institution's customers and incur high operational costs.

You can use event-driven architecture, to define and configure business events. Certain events are processed automatically while others will require manual attention. When an agent rule identifies an event, it immediately triggers the respective validation and decision mechanisms to take appropriate actions.

For example, a financial institution will periodically review customers. Customer X has been onboarded into a financial institution on the 1st of January 2016. Based on the risk profile of the customer and organizational policies, the system sets the next review date to the 1st of January 2018. The system automatically triggers a customer review case on or before 1st January 2018.

1. Overriding agents
2. Creating a new event

Overriding agents

You must save the scheduling agents into your application layer in order use them in an event driven architecture operation.

1. In Dev Studio, click **Dev Studio** > **System** > **Operations** > **Agent Management**.
2. Filter the Queue Class to **PegaCLMFS** and save it into your ruleset.
3. Stop each agent.
4. Click on the name of an agent. The system will open an agent rule that you can copy into your implementation ruleset for changes.
5. Review the configuration of the agent (frequency, access group, and so on) and save the changes.
6. Repeat steps 1-3, but filter for **PegaKYCFS**.

Creating a new event

You can create new events for your financial institution. In this example, the documents provided by the customer during onboarding are set to expire, and the financial institution has to ask the customer to provide the latest documents to adhere to due diligence processes.

This task references rules that are defined in Event-driven architecture. Refer to this article for more information.

To implement this example, do the following steps:

1. Create a new property to store the valid end date of the uploaded document.

   For example, call the property `validenddate`. Place this property on the document collection UI screen. If the documents collected are stored in a content management solution (such as Alfrecsco), make sure that while accessing the customer documents, the `validenddate` property is also brought up in the journey.
2. Create an event code and event type for the document review case and map it in the `MapFSEvents` map value.

3. Create an advanced agent which runs at required intervals.

   These intervals are determined by the duration in which the documents are valid. Use the `CreateCustomerReviewEvents` agent rule as a basis for this rule.

4. Create an activity which retrieves the customer profiles whose documents are expired. Link this activity to the advanced agent in step 3, which calls it periodically.

5. Create a report definition which retrieves the customers whose documents are about to expire, per the `validenddate` rule.

   Use the `UnqueuedExpiredCustomerProfiles` report definition as a basis for this rule. If the documents collected are stored in a content management solution, make sure that, while accessing the customer documents, the `validenddate` is also brought up in the journey which is required for the report definition.

6. Create an activity to check if a case has already been created for customers with documents that are about to expire.

   Use the `QueuedCustomerDetails` report definition as a basis for this rule.

7. Create an activity which creates a document review case and add this activity to the `FSIFEventDrivenProcess` decision table.

   Use the `CreateCustomerScheduleReviewEvents` activity rule as a basis for this rule. The `FSIFAgentProcess` checks the `FSIFEventDrivenProcess` decision table for the activity to be run for the Document review event type, which creates the document review case.

Configuring asynchronous processing

Heavy processes that run during a user's session can create delays between the last action that the user took (for example, submitting an assignment) and the return of the control to the user, either by showing the confirm harness or by opening the next active assignment. For processes where the execution time takes a few seconds and where the current operator is likely to take the next action, this waiting period is acceptable. It is not acceptable, though, in cases where the execution time goes beyond 10 or more seconds and the next task will be likely performed by a different team.

For processes where the waiting period is excessive, the system uses asynchronous processing. This gives control back to the users as soon as they take action on the case, so that they can keep working on other cases while the system processes the slower case. The system works asynchronously until human intervention is required or the case is resolved.

By default, the following cases work asynchronously:

- Navigating from capture Stage to enrich stage (*CaptureToEnrich*)
- Navigating from enrich to due diligence stage (*EnrichToDuediligence*)
- Creation of Local KYC cases (*LKYCCases*)

For more information about asynchronous processing, see the Asynchronous processing article.

Do the following tasks to enable, configure, and extend asynchronous processing.

1. Enabling asynchronous processing
2. Enabling individual asynchronous processes
3. Configuring the underlying asynchronous process infrastructure
4. Extending asynchronous processes

Enabling asynchronous processing

A dynamic system setting enables or disables asynchronous processes throughout the entire application. By default, the system is configured with this setting enabled.

To disable asynchronous processing throughout the entire application, do the following steps.

1. In Dev Studio click **Records** > **Dynamic System Settings**.
2. Search for and select *clmfs/AsyncProcessingSwitch*.
3. In the **Value** field, enter `false`.

What to do next: To enable individual asynchronous processes, see Enabling individual asynchronous processes.

Enabling individual asynchronous processes

If you do not want all of the default asynchronous processes to run, you can specify which individual processes you want to run asynchronously. For example, you can decide that from the processes configured by default, only two of them should be enabled, while others should remain running in foreground as they did in previous versions of the product.

This configuration is stored in a decision table that stores all the trigger points (processes) where asynchronous processing is used.

1. In Dev Studio search for the *AsyncProcessingSwitchTriggerPoint* decision table rule.
2. Enable or disable a certain trigger point by setting the value to `true` (enabled) or `false` (disabled) in the row corresponding to the trigger point. By default, all processes are enabled. If you set the *clmfs/AsyncProcessingSwitch* to `false`, the configuration of this table is irrelevant, and no process will be run in asynchronous. The following asynchronous processes are enabled by default.

   CaptureToEnrich

   Navigates from the Capture stage to the Enrich stage. It includes, as required, the notification to stakeholders after the Capture stage, third party data incorporation, welcome pack generation, and customer investigation.

   EnrichToDuediligence

   Navigates from the Enrich stage to the Due diligence stage. It includes the synchronization with the system of record and the generation of all the Global KYC, Regulatory, Tax, Credit and Legal due diligence cases, including their automated solution, if applicable.

   LKYCCases

   Creates LKYC cases after the completion of the initial stages of Global KYC cases (Related party Global KYC and Customer investigation).

Configuring the underlying asynchronous process infrastructure

The asynchronous processing functionality is implemented using Pega queue processors that process cases according to certain parameters. Customers may need to change the default configuration to meet the needs of their specific implementations.

For more information see the Asynchronous processes Community article.

1. In Dev Studio search for the *AsyncProcessing* or *AsyncProcessingDelayed* rules and select the queue processor.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Make the necessary changes.
4. Click **Save**.
5. To ensure that the new configuration is recognized by the system, ensure that your application is considered by the system by the new System Context mechanism do the following steps.
   a. Open the application model by clicking *<application name>* > **Definition**.
   b. Expand the **Advanced** section and ensure that **Include in background processing** is selected.

Extending asynchronous processes

You can configure your own processes to follow the asynchronous processing pattern by including a reusable sub-flow into your flows and configuring the trigger point.

1. In Dev Studio add the *AsyncProcessing* sub-flow to the flow of the stage to which you want to add asynchronous processing.
2. Double-click the **AsyncProcessing** flow shape, and name the trigger point in its parameters.
3. Search for the *AsyncProcessingSwitchTriggerPoint* decision table, add the new trigger point, and set it to `true` to enable it.
4. Search for the *AsyncProcessingConfig* decision table to configure the amount of attempts, and how often to retry those attempts, and so on.

What to do next: For more information on these extension points, see Asynchronous process extension points.

Asynchronous process extension points

The following rules can be configured to modify the behavior of both default processes and custom processes.

AsyncProcessing

Flow used to run a process in the background using agents and queue processors. It must be invoked as a sub-flow, which creates an event and waits at an assignment shape and proceeds in the background with the help of an agent. The flow requires a *TriggerPoint* parameter, which needs to be passed when the flow is invoked. This trigger point is used to capture the location where the flow has been invoked. The same parameter needs to be configured in the *AsyncProcessingSwitchTriggerPoint* decision table. While customers are expected to add this flow into their own processes, it is not expected that the flow itself is modified to obtain custom behaviors.

AsyncProcessingSwitchTriggerPoint

Decision table used for capturing all trigger points where asynchronous processing is used. When the *AsyncProcessing* flow is invoked in each stage, it passes a parameter from this decision table.

AsyncProcessingPreSetValues

An extension data-transform which can be used to set values on the case before it is queued for asynchronous execution.

AsyncProcessingPostSetValues

An extension data-transform which can be used to set values on the case after it is queued for asynchronous execution.

Policy profile storage

Know Your Customer for Financial Services 8.2, that is shipped with this product, contains some significant changes to the way customer policy profiles are stored. These policy profiles include the KYC questions and answers collected from KYC cases which are stored for audit and reuse purposes.

In previous versions of the application, customer KYC data was stored in their master profile. You can now store the KYC Types associated with a customer in a new policy profile repository. By default, the system uses the new policy profile to store the KYC data of new customers. Existing customers' KYC data that was stored in their master profile will be automatically migrated into the new repository the next time that their master profile is saved. This migration process automatically transfers all KYC data into the new repository.

This change significantly reduces the footprint of the master profile and will improve the overall performance of the application. It is important to note, however, that implementations with custom master profile functionality will need to assess the impact of the changes and act accordingly.

For more details about how to configure this functionality, including how to disable it and restore the legacy mode, see the Configuring the policy profile section of the *Pega Know Your Customer for Financial Services Implementation Guide* on the Pega Know Your Customer for Financial Services product page.

Master profile loading and saving mechanism

The 8.2 version of the application introduces some changes to the way that the master profile is loaded and saved so that the overall performance of the application improves. No changes are required to use the new mechanism, unless some kind of customization has been done in that part of the application. A proper impact assessment is required in that case.

The new model keeps the interface with the rest of the application unaltered. Customers are still expected to retrieve the Master Profile using *GetMasterProfile* (data retrieval plus synchronization) and *D_GetMasterProfile* (data retrieval). The persistence of the data is still expected through *SaveWorkFolder*.

The main changes are implemented at the time of loading *D_GetMasterProfile*, a thread level data page loaded in every interaction. The changes ensure that the master profile is only loaded from the database when data has changed. Otherwise, it reuses information already in memory.

To support this new mechanism, two new data-pages have been created:

D_GetMasterProfileCache

(Requestor) – Data page to store the last version of the master profile that was loaded or saved for a given customer.

D_GetMasterProfileCommitTime

(Interaction) – Data page that returns the last time the master profile was persisted into the database (the last time that changes were made).

In order to make the most of this mechanism and to automatically adopt it in future related features, it is important that customers let the application manage the master profile or, if absolutely required, access it using the existing interface as defined above (*GetMasterProfile/SaveWorkFolder*).

AML CDD profile categorization

During the preparation of the due diligence for a given customer, the system executes a series of rules to determine the anti-money laundering (AML) customer due diligence (CDD) profile. The profile assigned to each customer determines the content and applicability of the KYC questionnaires that the customer must answer. The application currently manages the following profiles.

Contracting party (Main customer)

Profile name	Purpose
Simplified Due Diligence (CPSDD)	Simplified version of the standard due diligence for contracting parties
Standard Due Diligence (CPFDD)	Complete, standard diligence for contracting parties
Enhanced Due Diligence (CPEDD)	Enhanced due diligence applied for contracting parties
No Due Diligence (NoDD)	Exempt from due diligence questioning

Controlling party (Relevant related party)

Profile name	Purpose
Simplified Due Diligence (RPSDD)	Simplified version of the standard due diligence for related parties
Standard Due Diligence (RPFDD)	Complete, standard diligence for related parties
Enhanced Due Diligence (RPEDD)	Enhanced due diligence applied for related parties
No Due Diligence (NoDD)	Exempt from due diligence questioning

At the time of creating the KYC cases for a given party, Pega Client Lifecycle Management for Financial Services uses the AML profile to determine whether the cases should be created (SDD, FDD and EDD profiles) or not (NoDD). If a new case needs to be created, the AML profile information is passed to the KYC engine so that it can configure the applicable KYC Types for that case and show and hide questions according to the profile. For example, an SDD form is usually a leaner version of the FDD form and skips more than half of the questions from the FDD form. See KYC Due Diligence Profile for more details.

- AML CDD profile usage
- Adding a new AML profile to the application

AML CDD profile usage

AML Profile Calculation

The calculation of the profile is done during the synchronization of the Master Profile so that the system can react dynamically to any changes in the data that might trigger changes in the AML profile (for example, a change in risk). The value is usually based on risk but there may be other variables considered.

The logic is invoked through the *AMLCDDProfilesSynchronization* data transform, which provides access to a set of decision tables that manage different scenarios. As different business scenarios require different data, there is a corresponding set of data transforms to ensure that all of the data required is available at the time of execution. These are the decision tables shipped with the application.

Party Type	Decision table name	Data transform name
Contracting party – Individual	*AMLCDDProfileContractingInd*	*AMLCDDProfileContractingIndInit*
Contracting party – Organization	*AMLCDDProfileContractingOrg*	*AMLCDDProfileContractingOrgInit*
Contracting party – Fund	*AMLCDDProfileContractingFund*	*AMLCDDProfileContractingFundInit*
Related party – Individual	*AMLCDDProfileRelatedPartyInd*	*AMLCDDProfileRelatedPartyIndInit*
Related party – Organization	*AMLCDDProfileRelatedPartyOrg*	*AMLCDDProfileRelatedPartyOrgInit*
Related party – Fund	*AMLCDDProfileRelatedPartyFund*	*AMLCDDProfileRelatedPartyFundInit*

If there are changes required in the logic to calculate the AML profiles (for example, a new data variable is required or the values to be returned for a certain combination is different), customers may decide to overwrite these rules and implement their own logic. In addition, if customers want to organize the logic in a different way, for example, if they want to use some other variable instead of using the type of party to categorize the logic across multiple tables, they can overwrite the *AMLCDDProfilePartyRouter* decision table.

KYC Due Diligence Profile

The Pega Know Your Customer Regulatory Compliance for Financial Services application, which is the source of regulatory intelligence for Pega Client Lifecycle Management for Financial Services, contains KYC types that are already configured to manage the same AML profiles that Pega Client Lifecycle Management for Financial Services does (CPSDD, RPSDD, and so on).

These profiles are configured using a new KYC Due Diligence Profile Suite rule type created in Pega Know Your Customer application. The profiles are all recorded under an instance of that rule called *Default* and placed under *PegaKYC-Data-Type-Policies-Global*.

Although they all represent the same business scenarios, the profiles under this rule have different codes than those given by Pega Client Lifecycle Management for Financial Services. The mapping between the Pega Client Lifecycle Management for Financial Services AML Profiles and the Know Your Customer Regulatory Compliance AML Profiles is done in the *PegaFS-Work.DDProfilesMap* decision table. Customers that want to change that mapping, either because they changed the AML profiles in the Pega Client Lifecycle Management for Financial Services layer or in the KYC one, need to overwrite this extension point.

Adding a new AML profile to the application

To add a new AML profile to Pega Client Lifecycle Management for Financial Services, do the following steps.

1. Adding the new AML profile to the decision table
2. Optional: Creating a new decision table
3. Adding the new AML profile to KYC types
4. Adding the new AML profile to the mapping rule

Adding the new AML profile to the decision table

You must first add the new AML profile to the decision table. After these changes, your application will allocate the new AML profile according to the business logic in the modified decision table.

1. In the header of Dev Studio, search for the decision table to be modified (for example, *AMLCDDProfileContractingOrg*).
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Modify an existing row or create a new one if new logic is required.
4. Modify the decision table to either include or edit a value in the **Return** column (for example, *MediumCDD*).
5. In the *CDDProfile_AMLRational* column enter the name of a field value (for example, `@baseclass.property CDDProfile_AMLRational`) that describes the new AML profile value.
6. Save the rule.
7. If the new logic requires data that is not available at the party class (*PegaFS-Data-Party*), make a copy of the data transform with the same name as decision table (in this case *AMLCDDProfileContractingOrg*) and add `Init` to the name (for example, *AMLCDDProfileContractingOrgInit*) and initialize the data there.

Optional: Creating a new decision table

If the logic of your new AML profile does not fit well in any of the existing decision tables, you can create a new ones.

1. Make a copy of any of the AML CDD decision tables listed above in your implementation ruleset with a new name (for example, *AMLCDDProfileContractingDiv*).
2. Remove all of the existing rows and add the logic specific to your business scenario. Be sure to set default values in the **otherwise** row.
3. Save the rule.
4. Create a data transform in *PegaFS-Data-Party* with the same name as the decision table but add the suffix `Init` (for example, *AMLCDDProfileContractingDivInit*).
5. Add any logic to the data transform that may be required to initialize the data that is used by the decision table. Leave it empty if no logic is required, for example, all required data is available at *PegaFS-Data-Party*.
6. Make a copy of the *AMLCDDProfilePartyRouter* decision table and add a row with business logic that triggers the execution of the new table. Ensure that the new row is reachable as expected (the decision table stops executing after one of the rows is applicable).

Adding the new AML profile to KYC types

Add the new AML profile to the impacted KYC types where the new AML profile is used. Open the KYC type rule form and create a new local profile configured as inheritance.

1. Open the **KYCRuleManager** portal and open the KYC type to be modified.
2. In the Profiles list section, click **Add Local Profile**.
3. Enter a new unique identifier (for example `CPMDD`). The unique identifier provided here does not necessarily need to be the same one that was returned by the decision tables modified before.
4. Provide a description and select **Active based on Inheritance**. Submit the changes.
5. On the **Item Definition** tab open the items that will show or be hidden based on this new profile, and configure them accordingly.
6. Save the KYC type.
7. If the majority of your types will be impacted, consider overwriting the Default KYC DD Profile Suite shipped with Know Your Customer Regulatory Compliance for Financial Service and add the profile there instead.
   a. In Dev Studio, click **Records** > **Technical** > **KYC Profiles Suite**.
   b. Open the rule with name *Default* that applies to *PegaKYC-Data-Type-Policies-Global*. Make a copy of it in your implementation ruleset.
   c. Click **Add Profile** and enter the profile data as described before.
   d. Save the new copy of the Default KYC Profiles Suite.

   The new profile is available to all of the KYC types under the *PegaKYC-Data-Type-Policies-Global* class.

Adding the new AML profile to the mapping rule

Modify the *DDProfilesMap* mapping rule to translate the profile as defined in Adding the new AML profile to the decision table to its definition in KYC (Adding the new AML profile to KYC types).

1. In the header of Dev Studio, search for the *DDProfilesMap* decision table).
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Add a row to the table.
4. In the **Customer AML Profile** column, enter the value returned by the AML decision table created in Adding the new AML profile to the decision table for example, `MediumCDD`.
5. In the **Return** column, enter the value entered in the local Profile Suite or in the Default Profile Suite – for example, `CPMDD`.
6. Save the rule.

Configuring eScreening services

In order to comply with laws regarding know your customer (KYC), anti-money laundering, countering the financing of terrorism, and anti-bribery-corruption, banks may want to connect to a varity of eScreening services.

To enable and configure eScreening services, do the following tasks.

- Enabling eScreening
- Configuring eScreening server connection attempts
- Configuring the amount of time that the eScreening data is valid
- Adding data to the eScreening case

Enabling eScreening

To use eScreening services, you must first enable them.

1. In the Dev Studio header search text field, enter *licensedToUseeScreening* and select the dynamic system setting.
2. Change the value to `true`.

Configuring eScreening server connection attempts

You can configure the number of times the application tries to connect to the eScreening server. If the number of attempts to connect to the server exceed the limit, the screening case is automatically resolved.

1. In the Dev Studio header search text field, enter *eScreeningAttemptsLimit* and select the dynamic system setting.
2. Change the value to your desired amount of attempts.

Configuring the amount of time that the eScreening data is valid

You can configure the amount of time that eScreening data is valid. If the time period is exceeded, a new case is created to get fresh data from the screening server.

1. In the Dev Studio header search text field, enter *ScreeningValidity* and select the dynamic system setting.
2. Change the value to your desired amount of days.

Adding data to the eScreening case

If you want to propagate data from the screening case, in addition to the default data, to the main case so that it can be used during onboarding, due diligence, and so on, you must map this data in the appropriate data transform to include additional fields.

1. In the Dev Studio header search text field, enter *SynchronizeScreeningData_Extension* and select the data transform.

2. Modify the data sources and data fields as needed.

Related party operationalization

As organizations move their customers through onboarding, maintenance, review, and offboarding journeys, it is critical that they consistently understand all parties involved. Complex customers can have a large number of related parties. The due diligence requirements by party are based on many factors such as type, risk model output, and screening results. In order to provide an efficient process, the organization's systems must be able to identify each party, make ongoing assessments, and ensure the necessary level of due diligence activities are carried out. Any business rules driving the identification and assessment are subject to change over time so the rules must be configurable.

Every party in the system is categorized as either a contracting or a related party. Specific logic is then applied to determine an AML CDD profile across the levels of Exempt, Simplified, Full, or Enhanced due diligence. Using eScreening results and the current customer risk assessment, the orchestration engine automatically ensures that the appropriate level of due diligence is carried out. This logic is fully configurable for the institution's business needs.

- Adding an entity type

Adding an entity type

To add a new organization entity type, such as publicly traded company, modify the *AMLCDDProfileContractingOrg* decision table which contains entity types as well as the risk factor and type of due diligence each entity type receives.

1. In the header of Dev Studio, search for `AMLCDDProfileContractingOrg` and select the decision table.
2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Add a row and in the **Organization Entity Type** and specify the entity type.
4. In the **Client Overall Risk** column, specify the risk factor associated with the new entity type.
5. In the **Return** and **CDD AML Profile rational** fields specify the level of due diligence that the entity requires.
6. Click **Save**.

Optional: Installing the UPlus Web Self-Service Application

The UPlus web self-service Retail Banking reference application uses a combination of HTML and JavaScript in conjunction with Pega Web Mashup. If appropriate for this installation, install this web self-service reference site to demonstrate how a customer can initiate the onboarding process from a browser on their phone, tablet, or desktop.

Supported browsers

- Google Chrome™
- Microsoft® Internet Explorer® 10+

- Mobile (IOS and Android)

1. Deploy the `.war` files listed below to your application server. Refer to your application server documentation for details.
   - `WebArchives/fswssportal.war` (Context Root:'fswssportal')
2. Review the following files and update the existing base URL references by searching for instances of `<CLM-CLOUD-URL>` and replace it with your site-specific URL.
   - **\fswssportal\uplusbank\personal.html**
   - **\fswssportal\uplusbank\customernewaccount.html**
   - **\fswssportal\uplusbank\youraccount.jsp**
   - **\fswssportal\uplusbank\openclmcase.jsp**
3. Update the following **.jsp** files where **<PMFS-CLOUD-URL>** is mentioned with your site-specific PMFS URL.
   - **\fswssportal\uplusbank\productdetails.jsp**
   - **\fswssportal\uplusbank\carousel.jsp**
4. Update data instances for the Pega Marketing for Financial Services application to make sure the Pega Client Lifecycle Management for Financial Services and Pega Marketing for Financial Services integration is complete.
   a. Update the `ConnectCLMURL` dynamic system setting with appropriate Pega Client Lifecycle Management for Financial Services instance URL.
   b. Update the `CLMAuthProfile` the username and password with appropriate credentials that are used for creating cases from Pega Marketing for Financial Services.
5. Test the installation:
   a. Navigate to the base URL of your instance: **http://***<host>*:*<port>***/fswssportal/uplusbank/index.html**.
   b. Click the **PERSONAL** tab and ensure that the Deposit accounts are listed.
   c. Log in with the following credentials:

User Name	kelly.deitch@gmail.com

The installation is successful if the screen shows a list of Accounts.

Case navigation

Pega Client Lifecycle Management for Financial Services uses the Enrich stage to capture core data which drives the due diligence activities that are generated as cases in the Due diligence stage. In certain situations, users in the Due diligence stage may need to make changes to that data and use the update core information action to reinitiate the Due diligence stage accordingly. Users can use the action to move the main case back to the Enrich stage to capture the desired data.

When the case moves back to the Enrich stage, the system checks whether any users are actively working in Due diligence cases. If so, the system presents a list of those cases and asks the user to close them. The main case will not proceed back to the Enrich stage until they are closed and unlocked.

When all cases are closed or resolved, the system can continue. It shows a confirmation message explaining the impact that moving to the Enrich stage has on KYC cases that did not complete the review step. The KYC types and type-associated data in those cases are stored in a temporary Policy Memory storage mechanism and is reused when returning to the Due diligence stage. By default, the system deletes policy memory that is older than 30 days (this threshold can be configured by updating the *PolicyMemoryCleanupInDays* dynamic system setting). The KYC types stored in the policy memory are also deleted when the corresponding KYC cases are resolved and the KYC types are synced with the policy profile. For more details, see the Pega Know Your Customer For Financial Services Implementation Guide.

Once the user proceeds, all Due diligence KYC cases will be withdrawn and the main case will navigate back to the Enrich stage. In the Enrich stage only the necessary steps are available and reused. Users can update all data categories as each journey allows. The Monitor Document collection and Business Sponsorship Approval steps will reflect the changes to the data. For example, if the user adds an additional product from a jurisdiction in a new region, the user is prompted for additional documents and a Business Sponsor Approval case is generated for the new region.

Following the completion of the Enrich stage a second time, the case can then proceed to the Due diligence stage again, leading to a recalculation of the appropriate due diligence activities. The newly created due diligence cases are placed in the correct assignments based on the state of the corresponding Due diligence case when it was withdrawn. This behavior is driven by a configuration named STP mode.

The STP mode can have three possible values NOSTP, REVEIEWSTP and FULLSTP. These modes are determined based on the completion of KYC types and the status of the original KYC case when it was withdrawn (this status is captured in the policy memory when the onboarding case navigates back to the Enrich stage). Each of these modes will be active in the following situations.

FULLSTP

This mode implies that the case will be automatically resolved. A KYC case is set to use the STP mode when all the KYC types in the case are complete and approved.

REVIEWSTP

This mode moves the KYC case to the review assignment. A KYC case is set to this mode if its corresponding case was at the review step before navigating back to the Enrich stage.

NOSTP

This mode stops the case at both the Data collection and Review assignments. This mode is activated when the original case was at the Data collection assignment when it was withdrawn. This mode also overrules the other two modes if the KYC types on the case are incomplete.

These modes are not just limited to case navigation but also apply to the KYC case flows.

- Configuring case navigation

Configuring case navigation

As banks onboard and maintain customers, they need to capture data. This data can both be required for general use and essential to determine which due diligence activities must occur. If core driver data needs to be updated later in the Due diligence stage it will create the need to refresh this driving logic and update the new activities accordingly.

The functionality has been designed with a few extension points that can modify the behavior of the module to better suit your needs. If you need to make changes to these parameters, review the rules listed below.

For more information see the Case navigation Community article.

1. In Dev Studio search for the following rules and edit them, as necessary.

Rule name	Rule type	Usage
PegaCLMFS-Work.CaseNavigationResolveDDCases_Ext	Activity	This activity enables you to introduce logic before resolving the Due diligence stage subcases (for example, you can create and insert logic to save temporary KYC data).
PegaCLMFS-Work.CaseNavigationChangeStage_E	Data transform	This data transform enables you to introduce logic before navigating the stage to Enrich
PegaCLMFS-Work.CaseNavigationStageInfo	Data transform	This rule controls which stage to go to after navigation. By default, the application goes to the Enrich stage but customers may want to send the case to an intermediate step.
CaseNavigationCaseListToAbandon	Map value	Contains the list of subcases that are being withdrawn in the Due diligence stage before navigating back to Enrich. You can update this rule to include any other subcases that you want to withdraw before navigation. Subcases not in the list will remain open after navigation.
CaseNavigationDataMapping_Ext	Data transform	Maps or sets any properties or logic which can be used after navigating back to the Enrich stage.
Work-.PolicyMemoryCreator_Ext	Data transform	This rule is an extension point that can be specialized in the implementation layer to perform additional processing while creating the policy memory
PegaCLMFS-Work.DetermineSTPMode	Data transform	This rule is an extension point and can be used for customizing the logic which determines the STP mode or adds new STP modes. Adding a new STP mode requires you to update the flows to recognize the modes.

2. Save your changes.

Note:

For a better understanding of how the functionality works, review the *UpdateCoreInformation* flow action, the *CaseNavigationProcessing* activity, the *DetermineSTPMode* data transform, and the KYC case flows, which has all the core logic of case navigation such as obtaining the list of subcases to withdraw, changing the stage to Enrich, withdrawing the Due diligence stage subcases, creating the policy memory and placing the KYC cases into appropriate assignments.

Configuring client outreach

As a client moves through onboarding, maintenance, or offboarding journeys with a financial institution, a wide variety data and documentation is captured for them and their related parties. Data or documentation can be gathered from third party providers, reused if on file, and still valid or in many cases, sourced directly from the customer.

The Client outreach case type enables the client to provide required data or documents through a web self-service mechanism accessed from a laptop, tablet, or phone. Data and documentation received from the client can then be used in the ongoing customer journey.

The following tasks enable you to extend the client outreach functionality. For information about this functionality, see the Client outreach Community article.

1. Extending client outreach case creation
2. Extending active case data management
3. Extending case withdrawal
4. Configuring notifications
5. Extending the list of recipients
6. Extending the class structure

Extending client outreach case creation

During the creation of new Client outreach cases), the system displays a list of customers that the case can be assigned to. By default, the list contains the main customer of the case (base Pega Foundation for Financial Services implementations) or any party associated to the case with a Contact role category (Pega Client Lifecycle Management for Financial Services). Customers may want to change this behavior to send cases to different parties.

1. In Dev Studio search for the *ClientOutreachRecipientListPrepare_ext* data transform.

   Note: This is the extension point for the *ClientOutreachRecipientListPrepare* data transform which retrieves the list of contacts for organizations and the actual client for individuals.

2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Add parties that can be assigned to the Client outreach case.
4. Save your changes.
5. Upon selection of a recipient, the system uses the *ConsolidationKey* property (a combination of customer and recipient identifiers) to determine whether there is already an active case for the current client and the selected recipient and let the user create a new case accordingly. If there are in-flight cases for the same consolidation key than one built after recipient selection, new cases cannot be

created. The consolidation key can be modified to consider customers only or can be removed all together to avoid consolidation.

a. In Dev Studio search for the *CalculateConsolidatedKey _ext* data transform.

Note: When the *ConsolidationKey* is be empty the consolidation logic is not used (the Client outreach case will be created without checking for an existing for customer).

b. Make your changes to the data transform.

c. Save your changes.

Extending active case data management

When an active case is edited, the data captured on the screen – items and instructions – is transferred into the case so it can eventually reach the customer. If additional data is required, changes can be done to the *PropagateEditedData_ext* data transform.

1. In Dev Studio search for the *PropagateEditedData_ext* data transform.

Note: This is the extension point for the *PropagateEditedData* data transform.

2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Add data to the case.
4. Save your changes.

Extending case withdrawal

Client outreach cases can be withdrawn either automatically by the system or manually by the user. In both situations, cases are simply closed, and the assignments created for customers to fulfill the requests are deleted. Some additional logic (for example, to send notification to the recipient or to send a message to the front-end system to remove notifications for customer) can be added to the application by extending the two rules listed below.

1. In Dev Studio search for the *ClosePendingClientOutreachCases_ext* activity.

Note: This is the extension point for the *ClosePendingClientOutreachCases* activity used to automatically withdraw the Client outreach case.

2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Make changes to the behavior of automatically withdrawn cases.
4. In Dev Studio search for the *WithdrawPost_ext* activity.

Note: This is the extension point for the *WithdrawPost* activity used to manually withdraw the Client outreach case using a local action.

5. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
6. Make changes to the behavior of manually withdrawn cases.
7. Save your changes.

Configuring notifications

The system generates an email notification to the recipient when the case is created and when there are changes made to the case. In addition, the system can send extra notifications when the case reaches, in the Fulfillment stage, a certain goal or deadline service level agreement (SLA). The content of all these notifications can be modified by changing the rules listed below.

1. In Dev Studio search for one of the following correspondence rules.

Rule Name	Type	Details
NotifyRecipientCO	Correspondence	Correspondence sent to notify recipient in the Client outreach case that there is additional data or documentation required.
NotifyContactCODeadline	Correspondence	The extension for the *WithdrawPost* activity. Correspondence is sent to notify recipient that the Client outreach case has reached its deadline SLA.
NotifyOriginatorCOGoal	Correspondence	Correspondence sent to notify recipient that the Client outreach case has reached its goal SLA.

2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Make changes to the correspondence rules.
4. Save your changes.

Extending the list of recipients

The logic to build the list of recipients for a given case can be modified or extended. In the Pega Client Lifecycle Management for Financial Services layer, the list is built using the *D_ClientOutreachRecipientList* data transform, which pulls any associated party from the case with a role category of *Contact*. To change this logic, modify the data page or the data transform that feeds its content.

1. In Dev Studio search for the *ClosePendingClientOutreachCases_ext* activity in the *PegaCLMFS-Work* class.

   Note: This Populates from *pyWorkParty(Customer).*

2. Save the rule into your implementation layer. For additional information about locked and unlocked rulesets, see Copying a rule or data instance.
3. Make changes to the recipient list logic, as necessary.
4. Save your changes.

Extending the class structure

All the extension points for the Client outreach case can be override using a ruleset specialization (saved in the same class, using a different ruleset). However, if customers want to introduce new types of items – the system currently supports only basic data and documents – they would need to create their own classes to support those types. These classes need to be created either under the same hierarchy as the default ones, or in a new class structure. To keep a clear distinction between base product (Pega Foundation for Financial Services and Pega Client Lifecycle Management for Financial Services) and customer implementation, the second approach is recommended.

Pega Client Lifecycle Management for Financial Services provides an example of how these classes can be extended. New classes have been created (all under *PegaCLMFS* rulesets), to add new types, or change the behavior of the existing types. Customers can follow a similar pattern.

Class	Parent Class	Usage
PegaCLMFS-Data-ClientOutreach	*PegaFS-Data-ClientOutreach*	Wrapper data class for client outreach
PegaCLMFS-Data-ClientOutreach-Item	*PegaFS-Data-ClientOutreach-Item*	Wrapper class for different items
PegaCLMFS-Data-ClientOutreach-Item-Document	*PegaFS-Data-ClientOutreach-Item-Document*	Wrapper class for all document related items
PegaCLMFS-Data-ClientOutreach-Item-Document-Basic	*PegaFS-Data-ClientOutreach-Item-Document-Basic*	Used for holding all basic document items

Entity types

Pega Client Lifecycle Management for Financial Services comes with reference data to support different entity types. The data is stored in two different tables of the database that can be used by customers to add entity types that they may need to manage in their applications.

Data entity/table	Description
Entity types *fsf_ref_partysubtype*	List of all entity types available in the sytem. For example, Bank, Mutual Fund, Insurance Company, and so on.
Entity types per party *fsf_ref_partysubtypexref*	List of entity types per party type (organization/fund). For example, Bank can be used for organizations while Mutual Fund can only be used for Funds.

Reference data about entity types has been enhanced so that reference data more closely resembles the structure of financial institutions. During this enhancement process, some entity types and relationship codes and their attributes have been removed from the application reference data as they are no longer applicable. Removed data will not appear in new installations, but in those environments upgraded from

previous versions of the application (8.2 and before), the deprecated entity types and relationship codes need to be filtered out during retrieval so that they are not available to the users.

If, for any reason, your organization needs to make use of some of the codes that were removed, look at the following rules and adjust them as required.

Data entity	Filtering rule
Entity types *D_PartySubTypeXRefDescSummary*	*PegaFS-Int-FSF_REF_PARTYSUBTYPEXREF.IsRecordSetAsDeleted*

Relationship codes and roles

Pega Client Lifecycle Management for Financial Services comes with reference data that supports different relationships between parties. The data is maintained in different database tables.

Data entity/table	Description
Relationship codes *fsf_relcodes*	Table used to hold all the relationship codes and their categories irrespective of party type. For example, Primary Contact, Beneficial Owner, and so on.
Relationship codes per party *fsf_partyrelcodexref*	Table used to hold the role codes based on party types (individual or entity). Used by the application show different roles based on the party type. For example, show Beneficial Owner only for individuals.
Relationship code attributes *fsf_partyrelatttype*	Table used to hold all reference data of attributes based on relationship code. It is used by the application to maintain different attributes for different relationship codes. For example, the percentage of ownership for a Beneficial Owner.

If your organization need some additional values in these tables, they can be added and will become immediately available in your application. The data-pages used for the retrieval of this data are listed in the following table.

Data entity/table	Description
Relationship codes	*D_RelCodeSummary* - Retrieves all relationship codes *D_RelCodeOpen* - Opens a particular relationship code *D_GetRelType* - Fetches a list of relationship types and their description

Data entity/table	Description
Relationship codes per party	*D_PartyRelCodeXRefOpen* - Opens a particular code and party type *D_PartyRelCodeXRefSummary* - Retrieves list of relationship codes based on party type
Relationship code attributes	*D_PartyRelAttribTypesOpen* - Opens a particular attribute *D_PartyRelAttribTypesSummary* - Retrieves list of attributes based on relationship code

Reference data about the modeling of relationships (codes and their attributes) has been enhanced so that reference data more closely resembles the structure of financial institutions. During this enhancement process, some entity types and relationship codes and their attributes have been removed from the application reference data as they are no longer applicable. Removed data will not appear in new installations, but in those environments upgraded from previous versions of the application (8.2 and before), the deprecated entity types and relationship codes need to be filtered out during retrieval so that they are not available to the users.

Data entity	Filtering rule
Relationship codes *D_RetrieveRelcodeByCategoryAndDesc*	PegaFS-Int-FSF_RELCODES. IsRecordSetAsDeleted
Relationship code attributes *D_PartyRelAttribTypesSummary*	PegaFS-Int-FSF_PARTYRELATTTYPE. IsRecordSetAsDeleted

Enabling users to simultaneously work on cases

By default, when a user opens an assignment for action, the system locks the case that the assignment belongs to and its parent case to avoid conflicts created by other users acting on those cases. The lock persists until the assignment is saved, canceled, or submitted. You can enable multiple users to simultaneously work on the same assignments and cases.

For information about locking cases, see Locking Cases.

1. Ensure you have a copy of the Pega Client Lifecycle Management for Financial Services case types in your implementation layer rulesets. Although it is not mandatory, it is recommended as well to have them in your class structure.
2. Copy the *pyOptimisticLockingEnableParentLock* application setting (*Pega-ProcessArchitect* ruleset) into your rulesets.
3. Enable the configuration for all the production levels. This activates advanced optimistic locking settings in the case type rule.
   a. From the navigation panel, click **Records** > **SysAdmin** > **Application Settings**.
   b. Select *pyOptimisticLockingEnableParentLock*.

c. Select the check boxes to set the value to **True**.

d. Save the application settings.

4. Enable simultaneous work on your top-level case types.

a. In the navigation panel, click **Case Types** and select the case in which you want to enable simultaneous work.

b. Click the **Settings** tab.

c. Click **Actions** > **Open**.

d. Click the **Advanced** tab.

e. Select **Allow multiple users**.

f. Select the option **Lock parent case when an action is performed on the case** to ensure that there are no data integrity issues when two users work simultaneously on two cases from the same parent (for example, Local KYC cases).

g. Save the application settings.

5. **Optional:** If there are conflicts during the resolution of assignments, the system uses the *pyShowConflict* section (included in the *pyCaseHeaderOuter* section). If you have customized the default perform harnesses, you should update them and ensure that this section is included. The harnesses are:

- *PegaCLMFS-Work.Perform*
- *PegaCLMFS-Work-Requirement.Perform*
- *PegaCLMFS-Work-Requirement.RequirementHarness*

Note:

When two users access the same assignment in the same case at the same time, and the first user completes the assignment, the second user receives a message saying that the case has been modified. The system, however, leaves the second user in the original assignment instead of checking the validity of the assignment. It directs the second user to a review harness with a message stating that the assignment was already performed by someone else.

Note:

When a background process, such as the one that manages service level agreements or correspondence, takes control of the case for few milliseconds to update the case (for example, to increase the urgency), users that have the assignment open, will get an error saying that the case was modified and needs to be refreshed to regain control of it. Refreshing the case may result in the loss of some data entered in that assignment.

KYC Surgical policy updates

Pega Client Lifecycle Management for Financial Services is required to keep the KYC cases updated with the latest KYC polices so that the application can remain compliant with regulatory rules. Pega Client Lifecycle Management for Financial Services uses the Surgical policy update (SPU) feature from the KYC Engine to constantly push the new policies to in-flight KYC cases, when such policies are created or imported into the system.

Any KYC case in Pending-KYC, Pending-KYCReview, or Pending-KYCRMReview status is classified as an in-flight case and considered for a policy update. Pega Client Lifecycle Management for Financial Services uses the SPU in both the background and in manual update mode and thus ensures in-flight cases receive changes to regulatory policies.

When a policy update is applied to a KYC case, an in-flight case in the review stage might be rendered incomplete due to the application of new or updated policies. These cases must be reverted to the data collection assignments for the KYC operator to enter the required information. Restart the in-flight case to ensure that the STP mode determines the correct assignment for the case. See Case navigation to learn more about the STP mode.

When a policy is updated on a case, the assignee is shown a message on both the confirmation and the data collection screens to ensure that the assignee is informed of the update and can perform the next actions as required.

Approved and resolved KYC policies are isolated from posterior changes in the rules by using the KYC Engine policy-freezing mechanism. Thus the resolved cases and the KYC types present in the master profiles present the KYC types in the form that they were approved in the case.

For more information about Surgical policy updates, see the Surgical policy update topic in the Pega Know Your Customer for Financial Services Implementation Guide.

Testing a new application

Testing a new application involves testing in different environments.

- Testing your application in the Build environment
- Testing in the test or production environments
- Testing in the UAT environment

Testing your application in the Build environment

Test a new application in the Build environment before migrating the application to a test or production environment. Testing in the Build environment enables you to verify that basic functionality and interfaces work correctly and that performance is acceptable.

1. Run functional tests to test specific features from the end-user perspective.
2. Test features used by all service intents, such as: security, eligibility, search, and data loading. For this unit testing, automated scripts are recommended but not required.
3. Use the Performance tool to measure the performance of the application. For more information, see Track system utilization for a requestor session with Performance Analyzer.
   - Prior to extending your site-specific Pega implementation, establish a performance benchmark (baseline) by using the Performance tool. This allows subsequent, iterative performance tests against the baseline to help identify any degradation in performance resulting from development efforts.
   - Use the Performance tool to check the performance of the following features:
     - Search
     - Account selection
     - Loading of components
     - Kickoff of all service intents. For this unit testing, automated scripts are recommended but not required.
   - Save the test results so that you can compare them to future test results to determine whether an application update has a performance impact.
4. Verify that the Pega-provided reports and your custom reports run successfully, and that they show your implementation layer data, rather than the default demonstration data. This can be an automated test.
5. Test all integrations, both independently and with associated integrations.

   Test integrations for any optional Pega Client Lifecycle Management for Financial Services components and other applications that you plan to use. See the product documentation for the component or application to determine which product components to test.

6. Test security. Test the most common roles to ensure that the required access groups are configured and point to the correct software version.

Testing in the test or production environments

After you import the application to a test or production environment, test the application in the new environment to verify that it works correctly in that environment.

1. Verify that the source and the destination files are the same.
2. Run functional tests to test specific features from the user perspective.
3. In the test or production environment, run the Application Guardrails Compliance Score to ensure that the application meets guardrails.
4. Verify that the Pega-provided reports and your custom reports run successfully, and that they show your implementation layer data, rather than the default demonstration data. This can be an automated test.
5. Test all integrations, both independently and with associated integrations.

   Test integrations for any optional Pega Client Lifecycle Management for Financial Services components and other applications that you plan to use. See the product documentation for the component or application to determine which product components to test.

6. Verify that the integrations point to the correct system of record, and not to the system of record for the Build environment.
7. Test security. Test the most common roles to ensure that the required access groups are configured and point to the correct software version. Use these common roles in your smoke tests (see next step).
8. Run a smoke test to compare the source and destination environments. Verify that all tests that pass in the Build environment also pass in the test or production environment. If anything fails, compare the environments to determine whether a difference in environment could cause the test to fail. If the environment caused the failure, either fix the issue that caused the failure or adjust the test as appropriate for the new environment.
9. Run performance tests to verify that performance meets expectations. Pega recommends automated performance testing. Save the results so that you can compare them to future performance test results to determine whether an application update has a performance impact.

Testing in the UAT environment

After you complete testing in a Test environment, it is common to perform User Acceptance Testing (UAT) in a designated UAT environment, which could be a pre-production environment. UAT ensures that users will be able to successfully complete work and meet business objectives.

Note: Organizations that use Scrum for application development will complete less formal UAT as part of each sprint cycle.

1. Verify the integrity of the UAT environment.
2. Have the end-users (or business analysts acting the role of end-users) run scripts to test all scenarios including boundary and exception testing. The end-users (that is, the trainers, managers, and directors), perform the following steps during UAT:

   a. Verify that there are no major issues.

   b. Review changes to understand the features.

Packaging a new application

To migrate a new application to a different environment, you must first package the application so that it can be imported to the new environment.

- Merging application changes
- Packaging an application for migration
- Importing the packaged application

Merging application changes

If you developed your application features in separate branches, use the **Merge Branches** wizard to merge the branches before you package the application. The wizard shows any merge conflicts so that you can correct them before you merge the branches.

Packaging an application for migration

Before you can migrate a new application to a different environment, you must package the relevant data instances and rulesets into a product rule. The product rule is an instance of `Rule-Admin-Product`, and it is referred to as the RAP file.

1. In the header of Dev Studio, click **Configure** > **Application** > **Distribution** > **Package** to start the Application Packaging wizard. For information about using the wizard, see Product rules.
2. Complete each page of the **Application Packaging** wizard.
3. On the last page of the wizard, click **Preview**.
4. Review the contents of the generated RAP file.
5. On the last page of the wizard, click **Modify** to make any changes.
6. When the RAP file is complete, click **Export**.
   The wizard creates a .ZIP file in the **ServiceExport** directory on the current application server node.

Importing the packaged application

To deploy a new application to a different environment, import the .ZIP file that contains the packaged application to the new environment.

1. In the header of Dev Studio, click **Configure** > **Application** > **Distribution** > **Import**.
2. Use the Import wizard to import the target .ZIP file. For more information, see Importing rules and data from an archive by using a wizard.

   For information about how to swap the database connection pointers to your production database after an import to a production environment, see the Pega Platform Upgrade Guide at Deploy Pega Platform.

Production maintenance and monitoring

Production maintenance and monitoring include the following procedures:

- Business rule maintenance in the Production environment
- Application health monitoring
- Identifying and reporting issues

Business rule maintenance in the Production environment

You can give managers the ability to update other rule types in the production environment. For example, managers can update the Goals and Deadline for a certain case type. These rules must be delegated in Dev Studio first. For more information on rule delegation, see Delegating a rule or data type.

Application health monitoring

Pega Autonomic Event Services is an application that automatically monitors, retrieves, and organizes the alert data from one or more clustered systems throughout the enterprise. Pega also provides the Pega Predictive Diagnostic Cloud, which allows you to benefit from Pega Autonomic Event Services without installing it locally. Pega Predictive Diagnostic Cloud is a Software as a Service offering of Pega Autonomic Event Services.

Implementing the following best practices in your application can help to ensure optimal response times and overall application health:

- Segment application agent processing to a dedicated JVM (Java Virtual Machine). This configuration ensures that end users do not have to share resources with background processes.
- Monitor integration response time. Over time, slow integration points can cause average handle times to increase. When queues start to grow, it becomes very difficult to recover, which might require usage of offline services or a backup application.

Identifying and reporting issues

As with any application, your users will encounter issues that they need to report as they begin to use the application. When deploying your application to the production environment, complete the following steps:

1. Identify operational staff who will be responsible for responding to issues reported in the production environment.
2. Establish procedures with those resources to triage, respond to, and escalate issues.
3. Determine procedures for delivery of changes to the production environment.