

Pega Customer Decision Hub

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.pegasystems.com

Document: Pega Customer Decision Hub Implementation Guide

Publication date: December 08, 2020

Feedback

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

Contents

Application overview.....	5
Purpose of this guide.....	5
Prerequisites.....	5
Pega Customer Decision Hub application stack.....	6
Prepare.....	7
Preparing for the implementation.....	7
Creating your application.....	7
Creating operator accounts.....	8
Updating the Manager field for your organization units.....	8
Verifying installed rulesets.....	9
Optional: Implementing your Pega Marketing application in a CRM configuration.....	10
Building features.....	11
Application data model.....	11
Adding properties to data types.....	12
Implementing the customer class.....	12
Defining the customer class.....	12
Configuring the customer class.....	13
Creating your customer class.....	13
Mapping a customer class to a database table.....	13
Optional: Creating a class key property for an extended customer class.....	14
Adding associated customer entities.....	14
Setting the context for your customer class.....	15
Extending and updating the CustomerSearchResultGrid section.....	16
Optimizing properties to speed up segment creation.....	16
Configuring channels.....	17
Implementing email accounts.....	17
Defining email accounts.....	17
Configuring email accounts.....	18
Implementing outbound SMS accounts.....	19
Defining outbound SMS accounts.....	19
Configuring the outbound SMS account.....	19
Implementing push notifications.....	20
Defining push notifications.....	20
Configuring push notifications.....	20
Specifying delivery time frames.....	21
Implementing the security model and configuring access groups.....	21
Implementing the security model and organization structure.....	21
Authentication schemes.....	22
Implementing your authentication scheme.....	22
Defining your access groups.....	22
Defining access roles and privileges.....	22
Configuring work groups.....	23
Configuring work queues.....	23
Configuring access groups and roles.....	23
Getting started with Next-Best-Action.....	24

Adding business issues and groups.....	24
Importing Next-Best-Action settings from a template.....	25
Defining an alias for a group of actions.....	25
Implementing real-time containers.....	26
Returning additional properties in a real-time container response.....	26
Making custom Strategy Result properties available for real-time container data flows.....	26
Setting the custom property in a real-time container response.....	27
Returning properties of an alternate SR class in a real-time container response.....	29
Modify Service Level Agreement settings for the Container service.....	30
Optional: Granting permission to configure ethical bias policies.....	30
Optional: Configuring repository storage for marketing images.....	31
Configuring custom repository storage.....	32
Configuring default repository storage for Pega Cloud deployments.....	32
Optional: Integrating Pega Customer Decision Hub with Adobe Experience Manager.....	33
Allowing Pega Customer Decision Hub to access images from Adobe Experience Manager.....	35
Implementing Reports.....	35
Implementing reporting.....	35
Reports.....	35
Optional: Implementing Paid Media Manager.....	35
Configuring the connection to a people-based destination.....	36
Matching individuals to their accounts in Facebook, Google or LinkedIn.....	36
Configuring Facebook Ads destinations.....	36
Enabling offline conversion push on Facebook.....	37
Configuring Google Ads destinations.....	37
Enabling device ID targeting for Google Ads.....	38
Configuring LinkedIn Ads destinations.....	39
Configuring the connection to web-based and DMP destinations.....	39
Paid Media web service.....	39
Enable Next-Best-Action targeting on DMP-based destinations.....	40
Configuring DMP destinations.....	41
Enabling elastic search for Paid Media Manager.....	42
Marketing assets configuration.....	42
Optional: Materializing interaction history summaries.....	43
Verifying the System Health page.....	44
Testing a new application.....	44
Testing your application in the Build environment.....	44
Testing in the test or production environments.....	45
Testing in the UAT environment.....	45
Packaging a new application.....	46
Merging application changes.....	46
Packaging an application for migration.....	46
Importing the packaged application.....	46
Production maintenance and monitoring.....	47
Application health monitoring.....	47
Identifying and reporting issues.....	48
Revision management.....	48

Application overview

Pega Customer Decision Hub is a comprehensive marketing automation solution that delivers inbound action management and outbound marketing campaigns on a single platform. Pega Customer Decision Hub uses a unique combination of predictive and adaptive analytics, real-time decisioning, and business process management to dynamically manage cross-channel conversations (from action design to fulfillment), drive revenue, and expand customer lifetime value.

- [Purpose of this guide](#)
- [Prerequisites](#)
- [Pega Customer Decision Hub application stack](#)

Purpose of this guide

A project team that implements a Pega application uses this implementation guide to enable and extend application features to meet client business requirements. You must complete the product installation before starting the implementation procedures. For information about installing your application, see the product installation guides on the [Pega Customer Decision Hub product page](#).

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

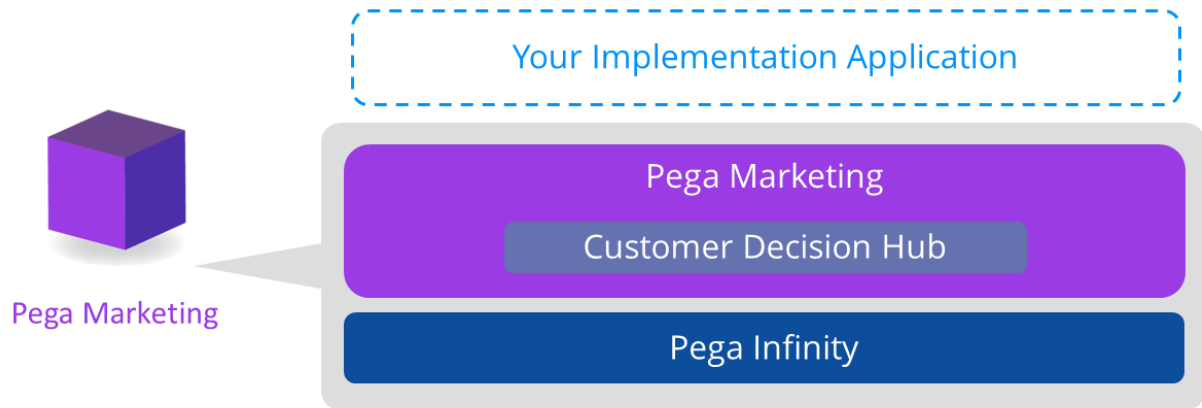
The application includes a case type backlog that supports the procedures in this implementation guide. The case type backlog is an inventory of the application-provided case types that establishes the project scope by defining the changes and additions that are required for the first MLP release and later releases. For more information, see [Capture microjourneys™ in the case type backlog](#) and [Journey Centric Delivery](#).

Prerequisites

Before you start your Pega Customer Decision Hub implementation, ensure that you understand the following information.

- Supported browsers. Review the *Platform Support Guide* on the [Pega Platform product page](#).
- Basic functionality of the Pega Customer Decision Hub application. For more information, see the [Pega Customer Decision Hub Foundation](#) course.
- Basic functionality of [Pega Next-Best-Action Advisor](#)
- Basic functionality of Pega Platform. Enroll in *Pega Platform Fundamentals* on the Pega Academy site.

Pega Customer Decision Hub application stack



Prepare

Preparing for the implementation

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

- [Creating your application](#)
- [Creating operator accounts](#)
- [Updating the Manager field for your organization units](#)
- [Verifying installed rulesets](#)
- [Optional: Implementing your Pega Marketing application in a CRM configuration](#)

Creating your application

Create your application and extend it to meet your organization needs by using the New Application wizard.

Before you begin: Access the Organization landing page and create a customer organizational structure. For more information, see the **Organization chart** tab on the [Organization landing page](#).

When you run the New Application wizard for Pega Customer Decision Hub, it creates a set of Pega-provided access groups. You are required to create your own operators and add them to the required access groups.

 **Caution:** If a Pega Customer Decision Hub application already exists in your system, running the New Application wizard overwrites some existing application settings. This will cause issues with your existing application. Having more than one active Pega Customer Decision Hub application on the same Pega Platform instance is not supported. This includes custom applications built by you and sample applications provided by Pega.

1. Log in to Dev Studio using the operator ID `administrator@pega.com` and the password that you specified for that operator.
2. In the Dev Studio header, click the name of your current application and click **New Application**.
3. Follow the New Application wizard instructions.
For more information, see [Creating an application](#).
4. Click **Go to app**.

To fully set up your application, in the top-right corner of the screen, click **Help > Application Guides** and follow the setup steps.

The New Application wizard creates the application class structure for you. You can reuse the classes and rulesets that the wizard creates in your applications. When you implement your applications, you can apply class hierarchy and inheritance principles to help you improve the maintainability and overall efficiency of your application.

For more information, see [Class layers](#) and [class hierarchy and inheritance](#).

Creating operator accounts

The New Application Wizard for Pega Customer Decision Hub creates the following default access groups:

<your implementation application>:CDHAdministrators

This is an access group for administrators. The default portal for operators in this access group is the App Studio workspace in administrative mode for the Pega Customer Decision Hub portal. This access group also grants access to Prediction Studio, Admin Studio, and the Pega Customer Decision Hub portal.

<your implementation application>:CDHAnalysts

This is an access group for analysts and other marketing users. The default portal for operators in this access group is the Pega Customer Decision Hub portal.

Create operator accounts for users of your implementation Pega Customer Decision Hub application and assign them to the predefined user groups.

1. Log in as a Pega Platform administrator.
2. Create an operator account with one of the above access groups. For more information, see [Operator ID data instances – Completing the Create or Save As form](#). Keep in mind the settings described below.
3. In the **Access settings** section on the **Security** tab, select **Allow rule checkout**.
4. Click **Update password** and enter the password for this operator, then click **Submit**.
5. Click **Save**.
6. Log off and log in as the operator you created.
7. Disable the **Auto checkout on save as** setting in the operator preferences. For more information, see [Setting your user preferences](#).



Tip: After creating the first operator record, open it in Dev Studio and click **Save > Save as**. This creates a new operator record with all the settings and preferences you previously entered.

What to do next: After you create operator accounts for users of your implementation application, you can invite them to log in at any time. To invite users to your application, log in to the Pega Customer Decision Hub portal as a user in the <your implementation application>:CDHAdministrators access group and click **Dev Studio > App Studio** to switch to the App Studio workspace. In App Studio, click **Users > People > Invite people to your application**.

Updating the Manager field for your organization units

The New Application Wizard creates the organization structure for your application. For each organization unit the wizard creates, specify the manager operator.

1. Log in as a Pega Platform administrator.
2. Click **Records > Organization > Organization Unit**. A list of all existing organization units is displayed.
3. Click on a unit to select the manager operator.
4. In the **Manager** field in the **Staff** section, enter or select one of the operators with the <your implementation application>:MarketManager access group.
5. Click **Save**.
6. Repeat the procedure for every unit in your organization.

Verifying installed rulesets

The following lists the highest level ruleset versions available after you run the New Application wizard. Verify that your system includes these ruleset versions.



Note: If your system does not contain the following rulesets, the application was not created correctly.

1. Log in as an administrator created for your implementation application.
2. In Dev Studio, click **Configure > Application > Structure > RuleSet Stack**.
3. Verify that the following ruleset versions are listed:
 - <your implementation application>:01.01.01
 - <your implementation application>-NBADTests:01-01-01
 - <your implementation application>-Artifacts:01-01-01
 - <your implementation application>-Rules:01-01-01
 - <your implementation application>:01-01-01
 - <your implementation application>Int:01-01-01
 - PegaMarketing_FieldMarketing:8 (Optional - Will appear only if installed.)
 - PegaMarketing_FieldMarketing-HealthCheck:08-04-02
 - PegaMarketing_FieldMarketing_Core:08.04 (Optional - Will appear only if installed.)
 - PegaMKT-FieldMarketing:08-04-02
 - PegaMarketing_FW:8
 - PegaMarketing_FW-HealthCheck:08-04-02
 - PegaMarketing_Core:08.04
 - PegaMKT-Install:08-04-02
 - PegaMKT-Configuration:08-04-02
 - PegaMKT-MarketingProfile:08-04-02
 - PegaMKT-Portal:08-04-02
 - PegaMKT-Desktop:08-04-02
 - PegaMKT-RevisionManagement:08-04-02
 - PegaMKT-Engine:08-04-02
 - PegaMKT-Integration:08-04-02
 - PegaMKT-MailUtilities:08-04-02
 - PegaMKT-PushNotification:08-04-02
 - PegaMKT-Security:08-04-02
 - PegaMKT-Cloud:08-04-02
 - PegaMarketing_FW_PreReqs:8
 - PegaFW-DashboardCharts:01-01-05
 - Pega-IAC:08-04-02
 - UIKit:13.01
 - UI-Kit-7:13-01-01

- PegaDM:8
 - Pega-DecisionManager:08-04-02
- PegaNPS:8
 - PegaFW-NPS:08-01-99
- PDCClient:01.01
 - PDCClient:01-01-04
- PegaRULES:8

Optional: Implementing your Pega Marketing application in a CRM configuration

Pega Marketing is ready to be used with the [Pega Customer Relationship Management suite of applications](#) to share data and service cases across the Pega Customer Service, Pega Sales Automation, and Pega Marketing applications.



Important: The Pega Marketing application must be created on a separate database cluster from the Pega Customer Service and Pega Sales Automation applications. The Pega Marketing application is a back-end analytical application with a typically smaller number of active users compared to the operational Pega Customer Service and Pega Sales Automation applications. Keeping the operational and analytical clusters separate helps ensure easier maintenance and decreases upgrade complexity.

Building features

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

 **Tip:** Pega Customer Decision Hub provides an interactive *Initial Setup* guide that can walk you through the process of deploying your application. You can complete the steps described in this chapter by following the *Initial Setup* guide and using the *Pega Customer Decision Hub 8.4 Implementation Guide* as reference. To begin, click **Resources** > **Complete your setup** > **Initial Setup Guide**.

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

- [Application data model](#)
- [Implementing the customer class](#)
- [Configuring channels](#)
- [Implementing the security model and configuring access groups](#)
- [Getting started with Next-Best-Action](#)
- [Implementing real-time containers](#)
- [Optional: Granting permission to configure ethical bias policies](#)
- [Optional: Configuring repository storage for marketing images](#)
- [Optional: Integrating Pega Customer Decision Hub with Adobe Experience Manager](#)
- [Implementing Reports](#)
- [Optional: Implementing Paid Media Manager](#)
- [Marketing assets configuration](#)
- [Optional: Materializing interaction history summaries](#)
- [Verifying the System Health page](#)
- [Testing a new application](#)
- [Packaging a new application](#)

Application data model

The application provides a set of data types, data pages, and sample data, which you can use to begin implementing your application. You will need to use the data from your system of record instead of using the sample data provided by the application.

Data modeling involves creating a conceptual model of how data items relate to each other in an application. In Pega Platform, the data model refers to a set of rules that work together to populate the data in your application. The system displays this data to help you process the information and can help you make decisions in your sales processes. The following rule types constitute your data model:

Data types

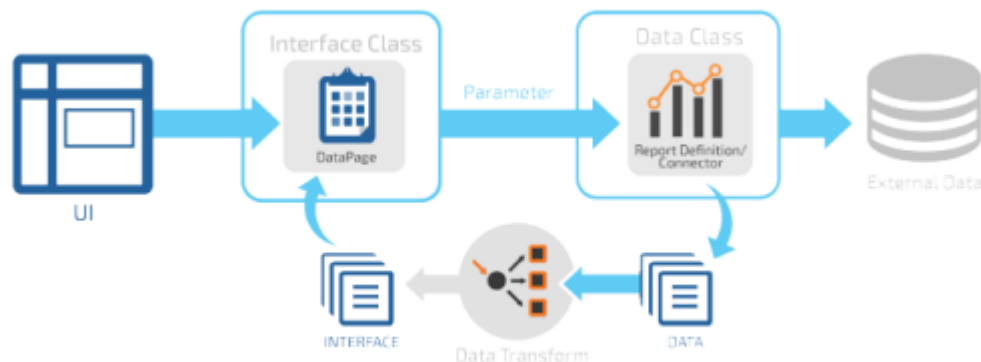
Data type is another name for a class in your application. It holds data that the application uses. A data type has one or more data pages and several property definitions associated with it.

Properties

Properties define the format and visual presentation of data in your application.

Data pages

Data pages define the content of a clipboard page. Data pages also control the loading of that data from a source system, as shown here.



When planning your data model, work with the data modeling resource at your site to understand the attributes of the data types that need to be supported by your application.

- [Adding properties to data types](#)

Adding properties to data types

Data types categorize the data that your application uses. The data types that come with your application have one or more data pages associated with them. Add properties to a data type to further customize it, for example, by adding contact information to a data type that represents a user of your application.

1. In the App Studio navigation bar of the Pega Customer Decision Hub portal, click **Data**.
2. Click the data type that you want to modify.
3. On the **Data model** tab, scroll to the bottom.
4. To add a property, click **Add field**. Add as many properties as your application requires. Repeat these steps for any other data types, as necessary.

For information about data pages, see [Data pages](#).

Implementing the customer class

Pega Customer Decision Hub requires a customer class to support the loading of customer data.

- [Defining the customer class](#)
- [Configuring the customer class](#)

Defining the customer class

Pega Customer Decision Hub requires a customer class to support the loading of customer data.

1. Define a name for the customer class.
2. Identify the customer class key. The default key property is `CustomerID`.
3. Identify the ruleset for the customer class.
4. Identify the associated entities (for example, `PurchaseData`) for this customer class.

Configuring the customer class

To successfully load customer data, a customer class must be configured.

The customer class represents the customer entity in Pega Customer Decision Hub. The base customer class is *PegaMKT-Data-Customer*, which is backed by the *MKT_CUSTOMER* table in the outbound marketing schema. Pega Customer Decision Hub utilizes the external mappings defined on the customer class to load up customer data.

1. [Creating your customer class](#)
2. [Mapping a customer class to a database table](#)
3. [Optional: Creating a class key property for an extended customer class](#)
4. [Adding associated customer entities](#)
5. [Setting the context for your customer class](#)
6. [Extending and updating the CustomerSearchResultGrid section](#)
7. [Optimizing properties to speed up segment creation](#)

Creating your customer class

To support loading customer data, create a customer class that extends the base customer class and has as its only key the property that you identified in [Defining the customer class](#). The default customer class key is `CustomerID`.

1. In the Dev Studio header search text field, enter `PegaMKT-Data-Customer`. From the results, select the `PegaMKT-Data-Customer` class.
2. Click **Save as**.
3. Enter the name for the new customer class, and then click **Create and open**.
For example: `MyCo-Data-Customer`
4. In the **Keys** section, ensure that only the correct key property is listed.
The default customer class key is `CustomerID`.
5. In the **Class inheritance** section, select the **Find by name first (Pattern)** check box, and then as the parent class, enter `PegaMKT-Data-Customer`.
6. On the **External Mapping** tab of the class rule, specify the properties available on the customer record and their backing columns in the customer table. Ensure that these mappings satisfy the following guidelines:
 - The column name does not exceed 27 characters
 - The table contains no empty rows
7. Click **Save**.

Mapping a customer class to a database table

After creating and configuring the class rule for your customer, create a database table instance to map your customer class to a database entity (table or view) in the marketing schema by completing the following steps.

1. In the **Records** menu, expand the **SysAdmin** category.
2. Right-click the **Database Table** node, and then select **+Create**.

3. Enter the name of your customer class and a short description, and then click **Create and open**.
4. In the **Database** section, enter details about the backing database entity. Work with the database administrator to find out what the details are.
5. Click **Save**.
6. Validate the mapping between your customer class and the specified database table by clicking **Test connectivity**.

What to do next:



Note: Whenever changes are made to either the class's external mappings or the columns in the backing database entity, to ensure that the changes are propagated correctly, re-save the database table instance.

Optional: Creating a class key property for an extended customer class

By default, the key of the customer class is the `CustomerID` property that is used to store customer IDs. If your extended customer class uses a different class key, for example, `AccountID`, then you must create two copies of the `CustomerID` property and rename them to the same name as your custom class key.

1. In the Dev Studio header search text field, search for and select the `CustomerID` property that applies to the `Data-` class.
2. Click **Save as**.
3. In the **Label** field, enter the name of the property that you identified in [Defining the customer class](#).
4. In the **Add to ruleset** section, select your Rules ruleset.
5. Click **Create and open**.
6. In the Dev Studio header search text field, search for and select the `CustomerID` property that applies to the `PegaMKT-` class.
7. Repeat steps 2 to 5.

Adding associated customer entities

Pega Customer Decision Hub supports associating the customer class with other entities. For example, each customer record can have associated purchase data information. After creating the customer class, create and configure any associated entities.

To add a new association for an entity (for example, `PurchaseData`) to the customer class, complete the following steps.



Note: The relationship with associated entities can be one-to-one or one-to-many.

1. Create a property rule for the associated entity directly under the customer class.
 - a. Ensure that the mode of the property is Page, Page List, or Page Group.
 - b. Ensure that the page class of the property is the class of the entity.
2. Create an association rule for the entity under the customer class.
 - a. In the **Class association** section, enter the class of the entity as the **Class name**.

- b. In the **Class join** list, add an entry to specify the join between the entity and the customer class. The **Type** value is ignored (leave it as *Include all rows...*). Specify the join criteria by clicking **Edit conditions**.

What to do next:



Note: When using an extended customer class, copy the `History` association rule and the `History` property rule from the base customer class (PegaMKT-Data-Customer) into the extended customer class. If the customer class has a class key different from the default `CustomerID` property, then also update the class join in the `History` association rule to reference the correct class key.

Setting the context for your customer class

Use the **Context Dictionary** to review or update the default contexts, configure a different primary context, or add an additional customer context.

The Context Dictionary identifies the customer entities, the relationships between these entities, and any associated data to be used by Pega Customer Decision Hub. By default, the primary context is `Customer`, mapped to the `<your organization name>-Data-Customer` class, and the prospect context is `Prospect`, mapped to the `<your organization name>-Data-Customer-Prospect` class. Additionally, the `Account` context is mapped to `<your organization name>-Data-Customer`.

1. In the App Studio navigation panel, click **Settings > Context Dictionary**.
2. Review the primary context.
 - a. Click the **Configure** icon by the name of a context.
 - b. Review the available properties.
 - c. If you defined a new extended customer class, select it in the **Class** field.
 - d. Review the associated data. If you defined additional associated data for your base customer class, click **+ Add associated data** to enable it.
3. **Optional:** To add a new customer context, click **Add context** and specify the name, customer class, and optional exclusion list. If you defined additional associated data for your new customer class, click **+ Add associated data** to enable it.

You can use additional customer context to build segments and launch campaigns. Using multiple data contexts allows you to better target your marketing efforts at multiple levels.
4. **Optional:** To set your new customer context as the primary Context, click **Change primary** and select the context in the **Select Primary Context** dialog box.

 **CAUTION:** Changing the primary context affects Segments and Campaigns which use it. If you have already defined any Segments or Campaigns, recreate them manually after changing the primary context to ensure that they continue working.
5. **Optional:** Click the **Configure** icon by the name of the prospect context. If you defined a new extended prospect class, select it in the **Class** field.

For more information about the prospect class, see [Configuring the prospect class](#).
6. **Optional:** Change the context storage in the **Context storage** section.

By default, the data resides in a relational database supported by Pega Platform in the form of a database table. If you plan to use segmentation, all data must be synchronized between the decision datastore and relational tables.
7. Click **Save**.

Extending and updating the CustomerSearchResultGrid section

After creating a custom customer class, you must extend the rules that are related to list segments. This is necessary so that your application can use customer data from your extended customer class.

CustomerSearchResultGrid is the default section that can be used to display a segment in a grid based on a segment identifier.

1. In the Dev Studio header search text field, enter `customersearchresultgrid`, and then select the **CustomerSearchResultGrid** section from the search results.
2. Click **Save as**.
3. In the **Apply to** field, enter the custom customer class that you configured in [Creating your customer class](#).
4. In the **Add to ruleset** section, select your Rules ruleset.
5. Click **Create and open**.
6. Click **Check out**.
7. On the **Pages & Classes** tab, in the **Class** column, replace the base customer class name with the customer class name that you configured in [Creating your customer class](#).
8. On the **Design** tab, on the **Table Layout** panel, click the **Gear** icon.
9. Click **Submit**.
10. Click **Save**.

For more information on list segments, see the *Pega Marketing User Guide* on the [Pega Marketing product page](#).

Optimizing properties to speed up segment creation

To improve value retrieval performance when selecting segment criteria from a list, you can add and remove optimizations for properties which you plan to use to create segments. When you select criteria values from a list in a segment rule, you use the **Select items** dialog box to select one or more values for the criterion. If the property has many values, populating the **Select items** dialog box can result in poor performance. By optimizing the property, the number of values that are queried is reduced, resulting in improved retrieval performance.

1. In the **Application Explorer**, in the search text field, enter `Data-Admin-DistinctValueIndex` and press **Enter**.
2. Click **Data-Admin-DistinctValueIndex** to open the Data-Admin-DistinctValueIndex class.
3. Click **Create** to create a distinct value index instance of the property that you want to optimize. You must define the property as a text, integer, or decimal type.
 - a. In the **Short description** field, enter a brief description of the instance.
 - b. In the **Class name** field, enter the name of the class that the property is associated with. The class cannot have security restrictions defined for it.
 - c. In the **Property name** field, enter the name of the property whose values you want to retrieve.
 - d. Click **Save**.

An agent calls the **populateDistinctValues** activity, which searches the specified class and caches the distinct values of the property in a table.

4. Use the optimized property in a segment. For more information, see *Chapter 6: Segments - Selecting Criteria Values from a List* in the *Pega Customer Decision Hub User Guide* on the [Pega Customer Decision Hub product page](#).

Configuring channels

Channels are a means of communication that customers can use to interact with a company. Pega Customer Decision Hub includes integrated, multi-channel support capabilities for managing email, SMS, push notifications and web-based service interactions. Additionally, in-person interactions can be conducted using a mobile device, such as a tablet computer.

Pega Customer Decision Hub allows you to build your process once and reuse it in any channel.

Configure channels by performing the following actions:

- [Implementing email accounts](#)
- [Implementing outbound SMS accounts](#)
- [Implementing push notifications](#)
- [Specifying delivery time frames](#)

Implementing email accounts

You use email accounts to send communication in the form of emails.

- [Defining email accounts](#)
- [Configuring email accounts](#)

Defining email accounts

You can define email accounts for use with your application.

Pega Customer Decision Hub uses the following two Outbound email accounts to manage outbound email transactions.

Default

This system-wide default account for sending emails is also the default account for sending email treatments in actions.

PegaMKT-Work

Used to send campaign-centric emails such as campaign approved and campaign suspended.



Note: You can install prepackaged connectors for email service providers (ESPs) Mailgun, SendGrid, and Mailjet. The connectors help reduce deployment time by automating the configuration of an email account. ESP connector packages are available for download on [Pega Exchange](#).

1. Define the following information for the **Default** email account.
 - Email provider
 - Sender name
 - From email address
 - User ID

- Password
 - Reply to email address
 - SMTP host name
 - SMTP port number
 - Use SMTPS?
 - Keystore instance (message signing)
 - Certificate alias (message signing)
2. Define the following information for the **PegaMKT-Work** email account.
 - Email provider
 - Sender name
 - From email address
 - User ID
 - Password
 - Reply to email address
 - SMTP host name
 - SMTP port number
 - Use SMTPS?
 - Keystore instance (message signing)
 - Certificate alias (message signing)

Configuring email accounts

Configure your email accounts with the required information for your email provider and account.

1. Log in to App Studio.
2. Click **Settings > Channels > Email**.
3. On the Email landing page, in the **Account Name** column, click the email account that you want to update: **Default** or **PegaMKT-Work**.
4. To change the email provider, complete the following steps.
 - a. Click **Select email provider**.
 - b. Select the email provider from the list.
 - c. Click **Submit**.
5. Update the values in the **Sender** section.

For more information, see [Configuring the email account in the Email Wizard](#).
6. To verify that the account information is correct, click **Test connectivity**.

You receive a summary of the test results. Ensure that the **Result** column for each step of the test displays **Success**, and then click **X** at the top right of the results window to close the window. If the test is not successful, correct the account information, and then repeat the test.
7. Click **Save**.

Implementing outbound SMS accounts

Pega Customer Decision Hub uses the Pega-provided Default account to send SMS treatments in actions.

- [Defining outbound SMS accounts](#)
- [Configuring the outbound SMS account](#)

Defining outbound SMS accounts

Pega Customer Decision Hub uses the Pega-provided **Default** account to send SMS treatments in actions.

1. Define the following information for the outbound SMS account.

- Account name
- Host address
- Host port
- User ID
- Password
- Test number
- Sender's number
- Maximum connections
- Restrict sending of action message to a specific time frame?

2. Define the following advanced settings for the outbound SMS account.

- Unit cost
- Source TON
- Source NPI
- Response timeout in seconds
- Max number of retries
- Character encoding type
- Reconnect on disconnect?

3. If required, define the name and value of any additional SMPP properties for this account.

Configuring the outbound SMS account

You can configure the SMS server information for outbound SMS transactions. Use the Pega-provided SMS account named **Default** for outbound SMS transactions.



Note: In addition to the Default account, you can create other outbound SMS accounts by clicking the **Add new connection** button on the SMS landing page. A new account can be referenced as the SMS delivery account in the Send SMS and Send Multi shapes in an action.

1. Log in to the Pega Customer Decision Hub portal.
2. In the App Studio navigation bar, click **Configuration > Settings > Channels > SMS**.
3. On the SMS landing page, click **Edit** to the right of the **Default** (outbound) account, and then update the account with information for your SMS server.
4. Click **OK**.

5. In the **Connection** column, to verify that the account configuration is correct, click **Test**.

You receive a summary of the test results. Ensure that the **Result** column for each step of the test displays **Success**, and then click **X** at the top right of the results window to close the window. If the test is not successful, correct the account information, and repeat the test.

Implementing push notifications

Pega Customer Decision Hub can send push notification messages to registered apps.

- [Defining push notifications](#)
- [Configuring push notifications](#)

Defining push notifications

Pega Customer Decision Hub can send push notification messages to registered apps. Push notifications can be sent while running actions.

For more information, see the *Push notifications* chapter in the *Pega Marketing User Guide* on the [Pega Marketing product page](#).

1. For every app, define the following information.

- Name of the app - how the app is referred to within the Pega Customer Decision Hub application, for example, in action flows
- Description
- Platform of the app - select Android or iOS
- Unique ID - used for validating push notification service requests, for example, for registering and un-registering app users

2. Define additional information.

- For an Android app, define the Google key.
- For an iOS app, define the following additional information.
 - App certificate - an Apple .p12 certificate
 - Certificate name - used for storing the certificate in the system
 - Certificate password - used to validate the certificate
 - Certificate type
 - Monitor Apple's notification feedback service?

Configuring push notifications

You can register an app so it can receive push notification messages from the Pega Customer Decision Hub application. Push notifications allow a server to send (push) a message to the client. For example, mobile apps can send notification messages to their users.

1. Log in to the Pega Customer Decision Hub portal.

2. Complete the instructions in *Registering apps for push notification* in the *Pega Marketing User Guide* on the [Pega Marketing product page](#).

For more information, see the *Push notifications* chapter in the *Pega Marketing User Guide* on the [Pega Marketing product page](#).

Specifying delivery time frames

You can specify delivery time frames for outbound SMS and email accounts. The time frames only apply to the delivery of SMS and email treatments in actions. Other delivery, such as work object-related correspondence, is not affected by these time restrictions.

1. In the **ACCOUNT AVAILABILITY** section, click the **Specify Range** button.
2. Select the start value **From** and the end value **To** of the time range.
3. Ensure that the correct time zone is selected from the drop-down list.
4. Click **OK**.

Implementing the security model and configuring access groups

Security planning involves defining authorization and authentication strategies for your application. It is a best practice to create new access groups and roles that are based on the default access groups and roles that come with the product.

- [Implementing the security model and organization structure](#)
- [Configuring access groups and roles](#)

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:

- [Authentication schemes](#)
- [Implementing your authentication scheme](#)
- [Defining your access groups](#)
- [Defining access roles and privileges](#)
- [Configuring work groups](#)
- [Configuring work queues](#)

Authentication schemes

The Pega Platform offers the following authentication types:

PRBasic

Based on passwords in the Operator ID data instances and the login form. This is defined by the HTML `@baseclass.Web-Login` rule, which your application can override.

PRSecuredBasic

Similar to PRBasic, but passes credentials by using Secure Sockets Layer (SSL) with Basic HTTP authentication. The login form is defined by the HTML `@baseclass.Web-Login-SecuredBasic` rule, which your application can override.

PRCustom

Supports access to an external LDAP directory or a custom authentication scheme.

PRExtAssign

Supports external assignments (Directed Web Access).

J2EEContext

Specifies that the application server in which the Pega Platform is deployed uses JAAS to authenticate users.

Implementing your authentication scheme

Your site can use a centralized, automated means of maintaining operator data instead of maintaining it manually in your application.

1. Discuss [Authentication schemes](#) with your site's security and application server teams.
2. Determine the appropriate authentication type.

For more information on authentication scheme planning, see [Authentication in Pega Platform](#).

Defining your access groups

Define the access groups that you want to add to your application.

1. Identify additional access groups that are needed for your application.
2. Identify portals associated with these access groups.

For more information, see [Viewing access groups and operators](#).

Defining access roles and privileges

You can associate one or more roles with an access group. Roles are additive. The more roles that you add to an access group, the more authorization there is. Privileges can be associated with one or more roles.

1. Determine which roles are needed for your application. You can use the Pega Customer Decision Hub roles as a starting point.
2. Determine which privileges to associate with each role.
3. Associate each role with an access group.

For more configuration information, see [Groups and roles](#).

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 access groups and roles

It is a best practice to create new access groups and roles that are based on the default access groups and roles that come with the product. This insulates your application from changes if the roles and groups change permissions in a future release. Refer to the roles and groups that you identified during the procedures.

To update an access group, complete the following steps:

1. Click **Dev Studio > Configure > Application > Structure > Access Groups and Users**.
2. Select the access group that you want to copy.
3. Click **Save as** to save the role to your application.
It is recommended that you name the role with a different name. Update the roles to use the ones that you created for your application.
4. Click **Create and open**.
5. In the **Definition** tab, map the **Name** and **Version** field details with your application and click **Save**.
6. **Optional:** To update an access role, complete the following steps:
 - a. Click **Dev Studio > Configure > Application > Structure > Access Groups and Users**.
7. Select the access role that you want to copy.
8. Click **Save as** to save the role to your application.
Pega recommends that you give the role a different name.
9. Click **Create and open**.
10. Click **Save**.

Getting started with Next-Best-Action

Define the structure for your business and the actions you want to engage with your customers. Actions, previously referred to as propositions or offers, are the messages you want to communicate to your customers. They are organized into your business structure.

Before using Pega Customer Decision Hub, perform the following tasks to configure the business hierarchy:

- [Adding business issues and groups](#)

Adding business issues and groups

Implement the structure of business issues and groups in your organization.

Business issues represent the aspects of your business on which you want to engage customers. A business issue is the purpose behind the actions you offer to your customers. For example, actions which started with the purpose of retaining existing customers should be grouped under the business issue of Retention. Actions started with the purpose of acquiring new customers belong to the business issue of Acquisition.

Business groups organize customer actions into categories. For example, as part of the business issue of Acquisition, you can create groups for products like Credit Cards, Mortgages, or Personal Loans, with the intention of offering these to potential new customers.

Together, business issues and groups form the organizing structure for your customer interactions. Each action you prepare for your customers is associated with a business issue and a group.

1. Log in to the Pega Customer Decision Hub portal as a user whose user group has the *CanManageNBA* privilege.
By default, this privilege is granted to managers and administrators through the *PegaNBAM_FW:MarketingManager* role.
2. Click **Next-Best-Action > Designer > Taxonomy**.
3. Click **Edit**.
4. To add an issue, click **Configure**.
5. In the **Configure Business Issues** window, click an existing issue to enable it to use Next-Best-Action or click **+ Create** to add a new issue, for example, *Retention*.
6. Click **Apply**. The issue is added to the business structure.
7. **Optional:** If you want the business structure to display the issue description in the Description column, click the **menu icon > Edit issue**.
8. In the pop-up window, fill out the **Short description** field.
9. Click the **menu icon > Configure groups**.
10. In the **Configure Business Groups** window, click an existing group to enable it to use Next-Best-Action or click **+ Create** to add a new issue, for example, *Credit Cards*.
11. Click **Apply** to save your changes. The group is added to the business structure.
12. **Optional:** If you want the business structure to display the group description in the Description column, click the **menu icon > Edit group** by the name of the group.
13. In the pop-up window, fill out the **Short description** field, and then click **Submit**.
14. Click **Save** to save the Next-Best-Action business structure.

- [Importing Next-Best-Action settings from a template](#)

- [Defining an alias for a group of actions](#)

Importing Next-Best-Action settings from a template

To speed up and simplify the process of implementing Next-Best-Action, use predefined templates that contain Next-Best-Action settings.

Pega provides industry-specific templates with the recommended business structure, engagement policy, arbitration, and channel settings for the Communications and Financial Services industries. You can import these templates to quickly set up the basic settings for your organization. In addition to the predefined templates, you can save your own Next-Best-Action settings as a template and then import them into a different Pega Customer Decision Hub instance, for example, to transfer the settings from a stage to a production environment.

1. In the Pega Customer Decision Hub portal, click **Next-Best-Action > Designer**.
2. If you did not already configure Next-Best-Action settings, you can select a template from the drop-down on the Next-Best-Action Designer landing page.
The drop-down lists the templates available on [Pega Marketplace](#), including the Pega-provided Communications and Financial Services templates, as well as any additional templates shared by Pega users.
 -  **Tip:** Templates developed for a specific industry may require certain artifacts to exist in your application. For example, a template created for Pega Marketing for Financial Services assumes that your primary context class is *PegaCRM-Data-Customer-FS* rather than *PegaMKT-Data-Customer*. Make sure that you use templates applicable to your industry, or import the relevant artifacts into your application before you select the Next-Best-Action template.
3. If you previously configured or imported Next-Best-Action settings, you can change the template by clicking **Taxonomy > Actions > Switch template**.
 -  **Important:** Any settings that you previously configured are overwritten by the template change.
4. Review the imported settings and adjust them manually as needed.
5. **Optional:** To save your template settings to a local file, for example, to import them into a production environment after staging tests, click **Taxonomy > Actions > Switch template**.

Defining an alias for a group of actions

By default, Pega Customer Decision Hub uses the term actions to refer to the engagement messages. If needed, you can change this to a more appropriate term, such as Promotions or Nudges.

To change the terminology, edit the business hierarchy in the Next-Best-Action designer.

-  **Tip:** You can use different terms for actions based on the business issue and group to which the product belongs. For example, if your business structure includes a Loans group within the Retention issue, you can rename actions related to that issue and group to *Loan Loyalty Program*. For other issues and groups, you may use other, group-specific terms, or keep the default name.

1. In the **Taxonomy** tab of the Next-Best-Action designer, edit a business group.
2. In the **Action naming (singular)** field, enter the singular form of the name which will be given to actions from this group, for example, *Promotion*.
3. In the **Action naming (plural)** field, enter the singular form of the name which will be given to actions from this group, for example, *Promotions*.

For the selected business group, the terms `Promotion` and `Promotions` are now used for actions. For other groups, the default terms are still used. If required, you can edit the other groups and specify a different alias.

Implementing real-time containers

Real-time containers manage content in real-time channels, for example, on web pages. When a container is loaded, for example, when an individual views an ad for an action in a predefined placement on the Web channel, it triggers Next-Best-Action.

To prepare your application to use real-time container triggers for your actions, do the following tasks:

- [Returning additional properties in a real-time container response](#)
- [Modify Service Level Agreement settings for the Container service](#)

Returning additional properties in a real-time container response

You can make additional properties available for real-time container responses.

1. Create a Proposition or Strategy property under the top-level SR class.

Proposition properties define and store the value on the Action ruleform. Strategy properties are temporary and calculated as part of strategy execution.



Tip: Proposition properties should be created using the Proposition Management landing page. Strategy properties are created from within a strategy ruleform.

2. Make the property available to real-time container data flows.

For more information, see [Making custom Strategy Result properties available for real-time container data flows](#).

3. Set the custom property in a real-time container response.

For more information, see [Setting the custom property in a real-time container response](#).

4. **Optional:** To add custom properties for a supporting decision that uses an alternate SR class, see [Returning properties of an alternate SR class in a real-time container response](#)

- [Making custom Strategy Result properties available for real-time container data flows](#)
- [Setting the custom property in a real-time container response](#)
- [Returning properties of an alternate SR class in a real-time container response](#)

Making custom Strategy Result properties available for real-time container data flows

Every property output from a decision strategy associated with real-time containers in the Next-Best-Action Designer must be identified in the `pyDefaultWorkingSet` data transform. Update this data transform if you use real-time containers and if you have configured custom action properties.

1. In Dev Studio, create a new data transform called `pyDefaultWorkingSet` in your top level SR class.
2. On the **Definition** tab of the data transform, select the check box **Call superclass data transform**.

Enabling the **Call superclass data transform** option ensures that all properties defined in the built-in `pyDefaultWorkingSet` data transform are available. You do not need to define them again in your new data transform.

3. Add any properties which you want to make available as output properties from Strategy in the generated data flows.
 - a. In the first row, in the **Action** column, enter: Append and Map to
 - b. In the first row, in the **Target** column, enter: defaultWorkingSetPage.pyWorkingSet
 - c. In the second row, in the **Action** column, enter: Set
 - d. In the second row, in the **Target** column, enter: .pyPropertyName
 - e. In the second row, in the **Source**, enter the name of the property which you want to make available.
- The following image shows an example configuration of the *pyDefaultWorkingSet* data transform.

	Action	Target	Relation	Source
▼ • 1	Append and Map to	defaultWorkingSetPage.pyWorkingSet	a new page	
• 1.1	Set	.pyPropertyName	equal to	"AccountType"

4. On the **Pages & Classes** tab, enter the following information:
 - a. In the **Page name** field, enter: defaultWorkingSetPage
 - b. In the **Class** field, enter: Data-MO-DDF-DecisionStrategy
5. Click **Save**.

Setting the custom property in a real-time container response

After you add the new property to the Strategy Result class, ensure that it is returned as part of the Container service JSON response.

The Container service response structure can return two types of results: *RankedResults*, that is, the actual strategy results, and *Actions*, that is, the category or type of results. An example of an Action result is *Make sales offer*.

1. In Dev Studio, create a property under the appropriate class.

 **Tip:** *Int-PegaCDH-Container-Offer* is usually the recommended class to use.
2. Save the *ExtendedContainerProperties* Data Transform rule into your implementation ruleset.

In the **Apply to** field, select the top-level SR class or the corresponding Issue or Group class.
3. On the **Definition** tab of the new *ExtendedContainerProperties* rule, map the property that you created in the *Int-PegaCDH-Container-Offer* class to a property in your top-level SR class.
 - a. Right-click on row 2.2, and then click **Add child**.

Definition Parameters Pages & Classes Test cases Specifications History		
	Action	Target
• 1	Comment	IF NBAD is V2
▼ • 2	When	D_MKTNBAModelReference[pyClassName:Declare_MKTSys
▼ • 2.1	When	@String.equals(ContainerServicePage.Request.Resource,"A
▶ • 2.1.1	Update Page	Response
▶ • 2.1.2	When	@String.equals(ContainerServicePage.Request.Resource,"A
▼ • 2.2	Update Page	Response.RankedResults(<LAST>)
		Save-as this DT @ Issue/Group level property mapping. Also enable
		Set property to Int-PegaCDH-Container-Offer class from SR class, Ta
▶ • 3		

- b. In the new row, in the **Action** column, select **Set**.
 - c. In the **Target** column, enter the property that you created in the *Int-PegaCDH-Container-Offer* class, for example, *AccountType*.
 - d. In the **Source** column, enter the property that you created in your top-level SR class, for example, *AccountType*.
 - e. Click **Save**.
4. To extend the Action results that the container returns, do the following steps:
 - a. In Dev Studio, create a property under the *Int-PegaCDH-Container-Action* class.
 - b. On the **Definition** tab of the new *ExtendedContainerProperties* rule, right-click on row 2.1, and then click **Add child**.
 - c. In the new row, in the **Action** column, select **Set**.
 - d. In the **Target** column, enter the property that you created in the *Int-PegaCDH-Container-Action* class.
 - e. In the **Source** column, enter the property that you created in your top-level SR class.
5. If you previously associated the real-time container with any level of Next-Best-Action hierarchy, do the following steps to ensure that the configuration is updated:
 - a. In App Studio, click **Next-Best-Action** > **Designer** > **Channels**.
 - b. Click **Edit**.
 - c. In the **Triggers** section, remove the real-time container association by clicking **Configure**, and then **Remove**.

- d. Click **Apply**.
- e. Click **Save**.
- f. In the **Triggers** section, restore the real-time container association by clicking **Configure**, and then **Add**.
- g. Click **Apply**.
- h. Click **Save**.

Returning properties of an alternate SR class in a real-time container response

In order to add custom properties to the Container response for a supporting decision that uses an alternate strategy result class, do the following steps:

1. In Dev Studio, create a property under the alternate SR class by using the **Create property** method.



Note: Set the **pyDecisioningItem** custom field for this property to *StrategyProperty*.

2. In Dev Studio, create a new data transform called `pyDefaultWorkingSet` in the alternate SR class.
 3. On the **Definition** tab of the data transform, select the check box **Call superclass data transform**.
Enabling the **Call superclass data transform** option ensures that all properties defined in the built-in `pyDefaultWorkingSet` data transform are available. You do not need to define them again in your new data transform.
 4. Add any properties which you want to make available as output properties from Strategy in the generated data flows.
 - a. In the first row, in the **Action** column, enter: Append and Map to
 - b. In the first row, in the **Target** column, enter: `defaultWorkingSetPage.pyWorkingSet`
 - c. In the second row, in the **Action** column, enter: Set
 - d. In the second row, in the **Target** column, enter: `.pyPropertyName`
 - e. In the second row, in the **Source**, enter the name of the property which you want to make available.
- The following image shows an example configuration of the `pyDefaultWorkingSet` data transform.

	Action	Target	Relation	Source
▼ • 1	Append and Map to ▼	defaultWorkingSetPage.pyWorkingSet	a new page ▼	
• 1.1	Set ▼	.pyPropertyName	equal to	"AccountType"

5. On the **Pages & Classes** tab, enter the following information:
 - a. In the **Page name** field, enter: `defaultWorkingSetPage`
 - b. In the **Class** field, enter: `Data-MO-DDF-DecisionStrategy`
6. Click **Save**.
7. In Dev Studio, create a property under the *Int-PegaCDH-Container* class.
8. Save the *TransformSROutput* Data Transform rule into your implementation ruleset.
In the **Apply to** field, select the alternate SR class.

- On the **Definition** tab of the new *TransformSROutput* rule, map the property that you created in your alternate SR class.

The following image shows an example configuration of the *TransformSROutput* data transform.

	Action	Target	Relation	Source	
▼ • 1	Append and Map ▼	ContainerServicePage.Response.RankedResult	a new page ▼		🗑️
• 1.1	Set ▼	.ClassIdentifier	equal to	"Eligibility"  Select values + 	🗑️
• 1.2	Set ▼	.EligibilityType	equal to	.EligibilityType  Select values + 	🗑️

- Clear the **Call superclass data transform** check box.
- Save and check in your changes.

Modify Service Level Agreement settings for the Container service

The Container REST service triggers real-time container processing. You can modify the default Service Level Agreement settings for the Container service, for example, to change the maximum number of consecutive violations which the Service allows before it invokes a fallback service activity.

- Save the Container service into your implementation ruleset.
- In the **Processing options** section, edit the following settings as required:
 - Maximum duration for service activity (milliseconds)** - The maximum time for a service activity to run. If a service exceeds the specified value, it logs a Service Level Agreement violation alert. The default value for this setting is 500.
 - Maximum consecutive violations** - The maximum number of consecutive Service Level Agreement violations for the Container service. If the number is exceeded, the interval service returns the default set of actions for the Container service for a length of time defined in the **Retry interval (Seconds)** setting. The default value for this setting is 3.
 - Retry interval (seconds)** - The time during which the interval service returns the default set of actions for the Container service. After the retry interval ends, the interval service tries to invoke the actual service activity. The default value for this setting is 10.

Optional: Granting permission to configure ethical bias policies

Configure access privileges to grant selected operators the right to define ethical bias policies.

Ethical bias policies provide a way of testing your strategies for unwanted bias, as well as checking how extensive the bias is. For example, you can test whether your strategies generate biased results by sending more actions to male rather than female customers.

To edit the ethical bias policies, enable the *decision/simulation/biasCheck* dynamic system setting, and then grant the *pzBiasPolicyConfiguration* to an access group.

- Log in to Pega Customer Decision Hub as an operator with access to Dev Studio.
- In the navigation pane of Dev Studio, click **Records** > **SysAdmin** > **Dynamic System Settings**.

3. In the list of dynamic system settings, click the *decision/simulation/biasCheck* setting.
4. Set the value of the *decision/simulation/biasCheck* setting to `true`.
5. Click **Save**.
6. In the navigation pane of Dev Studio, click **Records > Security > Access Groups**.
7. Edit the access group that you want to have the right to configure ethical bias policies for your organization.

 **Tip:** This type of access is usually granted to administrators. If required by company policies, you can create a new access group specifically for the purpose. For more information, see [Creating an access group](#).
8. In the **Available roles** section, click **Add role**.
9. Enter a name for the new role, for example, `<your implementation application>:MarketingBias`, and then click the **Open** icon.
10. On the **Create Access Role Name** page, click **Create and open**.
11. Select **Inherit privileges within class hierarchy**, and then click **Save**.
12. On the **Role** tab, in the **Access class** table, click **Add a row**.
13. In the **Add access Role Object** window, in the **Class** field, enter `@baseclass`.
14. In the **Access Controls** section, set the value of all controls to 5.
15. In the **Privileges** section, click **Add item**, and then enter the following settings:
 - a. In the **Name** field, enter `pzBiasPolicyConfiguration`.
 - b. In the **Level** field, enter 5.
16. Click **Save**.
17. On the access group tab, click **Save**.

Operators associated with the access group can now configure ethical bias policies in the Pega Customer Decision Hub portal. For more information, see "Ethical Bias Policies" in the *Pega Customer Decision Hub User Guide* on the [Pega Customer Decision Hub product page](#).

Optional: Configuring repository storage for marketing images

You can configure Pega Customer Decision Hub to use repository storage for your Image Library.

The Image Library is used to store and maintain marketing digital assets. By default, the assets are stored in a traditional rule-based Image Library. You can instead configure repository storage for your images. Using repository storage for your Image Library gives you the integrated ability to manage and distribute marketing images, improves usability, and simplifies the launch of email campaigns at scale.

- [Configuring custom repository storage](#)
- [Configuring default repository storage for Pega Cloud deployments](#)

Configuring custom repository storage

For all cloud or on premise implementations you can enable repository storage with your existing S3 repositories (S3, Azure, or local file storage).

Before you begin: Make sure that the IP of the Content Delivery Network you want to use for repository storage is whitelisted for the Pega Web Service, so that the Content Delivery Network can access and serve images from the Web Service.

1. Configure the repository connection.

For more information, see [Creating a repository connection](#) or [Creating custom repository connections](#).

2. In Dev Studio, configure the following Dynamic System Settings:

- **MarketingImageAssets/IsRepositoryStorageEnabled** - Set to `true`.
- **MarketingImageAssets/CustomRepositoryName** - Enter the name of the repository rule which points to your repository. This setting overwrites the value of the setting **MarketingImageAssets/DefaultRepositoryName**.
- Optional: **MarketingImageAssets/Folder** - Enter the name of the folder where Pega Customer Decision Hub will store the image assets. This folder will be located inside the root folder defined in the repository rule. If you do not define a folder for the images, they will be stored directly in the repository root folder.
- Optional: **MarketingImageAssets/CustomBaseURL** - Enter the URL of the Content Delivery Network which you configured, and update the **MKTDefaultSecured** content security policy rule to include **CustomBaseURL** under the **Image-Source** category in the **Allowed Websites** column. If you leave this setting blank, the images will be rendered directly from the Pega Web Service. This setting overwrites the value of the setting **MarketingImageAssets/BaseURL**.

3. Optional: To modify the maximum file size allowed for images, edit the **MarketingImageAssets/AllowedFileSize** Dynamic System Setting.

4. Optional: To allow image types other than .png, .jpg, .jpeg, .tiff, and .heif, save the data transform **AllowedImageTypes** to your application ruleset and add new types as required.

5. Run the **D_RepositorySettings** data page with flush mode on, or clear the cache.
Flushing the DSS settings is required to apply the changes made to the settings.

What to do next: If you replace an image in the repository with a new version, the cached image on the Content Delivery Network must be invalidated and replaced with the new version of the image. Cache invalidation is automatic in the Pega Cloud repository, but when using your own S3 repository, you must configure the invalidation logic, which should be applied on a DELETE event to clear the image cache. For more information, refer to the documentation of your S3 repository.

Configuring default repository storage for Pega Cloud deployments

For Pega Cloud deployments, you can enable and configure the default repository storage.

1. In Dev Studio, configure the following Dynamic System Settings:

- **MarketingImageAssets/IsRepositoryStorageEnabled** - Set to `true`.
- Optional: **MarketingImageAssets/Folder** - Enter the name of the folder where Pega Customer Decision Hub will store the image assets. This folder will be located inside the root folder defined

in the repository rule. If you do not define a folder for the images, they will be stored directly in the repository root folder.

- Optional: **MarketingImageAssets/BaseURL** - Enter the URL of the Content Delivery Network which you configured, and update the **MKTDefaultSecured** content security policy rule to include **BaseURL** under the **Image-Source** category in the **Allowed Websites** column. If you leave this setting blank, the images will be rendered directly from the Pega Web Service.
2. Optional: To modify the maximum file size allowed for images, edit the **MarketingImageAssets/AllowedFileSize** Dynamic System Setting.
 3. Optional: To allow image types other than .png, .jpg, .jpeg, .tiff, and .heif, save the data transform **AllowedImageTypes** to your application ruleset and add new types as required.
 4. Run the **D_RepositorySettings** data page with flush mode on, or clear the cache. Flushing the DSS settings is required to apply the changes made to the settings.

What to do next: If you replace an image in the repository with a new version, the cached image on the Content Delivery Network is invalidated and replaced with the new version of the image. Cache invalidation is automatic in the Pega Cloud repository. The request to invalidate the cache is made immediately, but may take some minutes to fully refresh the image cache.

Optional: Integrating Pega Customer Decision Hub with Adobe Experience Manager

Adobe Experience Manager (AEM) is a content management solution which provides users with an interface to build, deliver, and maintain digital experiences. Use the Pega AEM Connector to connect your Pega actions to Adobe Experience Manager assets, so that Adobe Experience Manager assets can be used for marketing and decisioning.

You can navigate the Adobe Experience Manager folder structure and browse Adobe Experience Manager assets from inside Pega Customer Decision Hub. The Pega AEM Connector uses the Query Builder API provided by Adobe. For more information about the API, see the Adobe API documentation.

1. Download the Pega AEM Connector from Pega Exchange.
2. To open the Application rule form, in Dev Studio, click the down arrow by the name of your application, then click **Definition**.
3. On the Definition tab, in the Enabled components section, click **Manage components**.
4. Click **Install new**, select the file that you downloaded from Pega Exchange, and click **Open**.
5. Select the **Enabled** check box to enable this component for your application, and click **OK**.
6. In the list of enabled components, verify that the Pega AEM Connector is selected, and save your changes to the rule form.
7. In Dev Studio, click **Records > SysAdmin > Repository**.
8. Click **Create**.
9. Enter a description and a name for your repository.
10. Click **Create and open**.
11. In the **Definition** tab, click **Select**.
12. Select **Adobe Experience Manager** as the repository type.

13. **Optional:** By default, Pega Customer Decision Hub can create and delete files and folders in Adobe Experience Manager. If you want to disable this functionality, select **Read-Only**.
14. In the **Repository configuration** section, in the **Publish host name** field, enter the host name of your Adobe Experience Manager repository.
Pega Customer Decision Hub will use this host name to access Adobe Experience Manager.
15. **Optional:** If you want to automatically publish content to Adobe Experience Manager, enter the Author host name of your Adobe Experience Manager repository in the **Author host name** field, and select **Auto Publish**. The **Author host name** field is only available if you did not set the Adobe Experience Manager repository to Read Only in step 13.
16. In the **Root path** field, enter the name of the folder where Adobe Experience Manager assets are stored. The Pega AEM connector will allow you to browse the assets in this folder from inside Pega Customer Decision Hub.



Note: The Root folder path must not start with the / character.

17. In the **Authentication** section, in the **Authentication profile** field, enter the name of a new authentication profile and click the **Open** icon to configure the profile.
The authentication profile stores the credentials that Pega Customer Decision Hub needs to authenticate with the Adobe Experience Manager API.
18. In the **Type** field, select **Basic**.
19. Enter a name and description for your authentication profile.
20. Click **Create and open**.
21. Enter the user name, password, realm and host name required to authenticate with Adobe Experience Manager. For more information, refer to the Adobe API documentation.
22. Select **Preemptive authentication**.
23. Click **Save**.
24. On the **Definition** tab of your repository, click **Test connectivity**. If there are any errors, ensure that the credentials in the authentication profile are correct, and that Pega Customer Decision Hub can access the URLs you entered in the **Publish host name** and **Author host name** fields.
25. Click **Save**.
26. Click **Records > SysAdmin > Dynamic System Settings**.
27. Edit the following Dynamic System Settings:
 - *MarketingImageAssets/IsRepositoryStorageEnabled* – Set the value to `True`
 - *MarketingImageAssets/CustomRepositoryName* – Enter the name of the repository you created
 - *MarketingImageAssets/Folder* – Enter a valid sub-folder of the Root folder. If you leave this field blank, the Root folder is used as the destination folder for your images.
28. Flush the `D_RepositorySettings` data page to make sure the configuration changes are applied.
 - a. Click **Records > Data Model > Data Page**.
 - b. Open the `D_RepositorySettings` data page.
 - c. On the **Load Management** tab, click **Clear data page**.
 - d. Select **Flush All**.
 - e. Click **Submit**.

- [Allowing Pega Customer Decision Hub to access images from Adobe Experience Manager](#)

Allowing Pega Customer Decision Hub to access images from Adobe Experience Manager

Images stored in Adobe Experience Manager are treated as images from an unknown source by Pega Customer Decision Hub. To allow your application to access an image from Adobe Experience Manager, add the Adobe Experience Manager host in the Content security policy.

1. In Dev Studio, click the down arrow by the name of your application, then click **Overview**.
2. Click **Integration & security**.
3. In the Content security section, open the content security policy.
4. Expand the **Image-Source section**.
5. In the Allowed Websites table, click **Add a row**.
6. Enter the URL of your Adobe Experience Manager host.
7. Save the content policy.

Implementing Reports

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.

- [Implementing reporting](#)
- [Reports](#)

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.

Reports

In addition to the library of Pega Platform reports, Pega Customer Decision Hub 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](#).

Optional: Implementing Paid Media Manager

Paid Media Manager extends Next-Best-Action to modern digital advertising platforms, such as Facebook Ads and Google Ads. Each individual's Next-Best-Action is communicated through audiences on ad

platforms. To implement Paid Media Manager in your organization, you must start by configuring the connection to the paid destination.

- [Configuring the connection to a people-based destination](#)
- [Configuring the connection to web-based and DMP destinations](#)
- [Enabling elastic search for Paid Media Manager](#)

Configuring the connection to a people-based destination

Paid Media Manager uses the people-based matching APIs for Facebook Ads, Google Ads, and LinkedIn Ads to synchronize paid audiences. To set up a connection, you must configure both Paid Media Manager and the paid destination to which you want to connect.

- [Matching individuals to their accounts in Facebook, Google or LinkedIn](#)
- [Configuring Facebook Ads destinations](#)
- [Configuring Google Ads destinations](#)
- [Configuring LinkedIn Ads destinations](#)

Matching individuals to their accounts in Facebook, Google or LinkedIn

Facebook Ads, Google Ads and LinkedIn Ads destinations enable you to target individuals based on their email addresses, phone numbers, device IDs, and more. To do so, map destination API properties to Pega Customer Decision Hub properties. For example, you can map **Email (EMAIL)** to `.pyEmail1`.

While creating a paid destination, in the **Field mapping** section, map the API parameters to Pega Customer Decision Hub properties. At least one mapping is required. You can map email, phone number, and device ID API parameters to multiple Pega Customer Decision Hub properties.

Configuring Facebook Ads destinations

Facebook Ads Destinations enable ads to be targeted on Facebook, Instagram, Messenger, and third party apps and websites. Paid Media Manager integrates directly with Facebook Ads and the Facebook Custom Audience API.

1. Create a Facebook App to access the Facebook Marketing API. For more information, see [Creating a Facebook Ads app and credentials](#).
2. Log in to Pega Customer Decision Hub as an operator with access to App Studio.
3. In App Studio, click **Settings > Channels > Paid Media > Facebook Ads**.
4. Click **Add destination**.
5. Fill out the following fields. For more information, see [Access and Authentication](#).
 - **Destination name**
 - **Account ID**
 - **App token**
6. In the **Field mapping** section, map the Facebook Ads **Email (EMAIL)** property to a Pega Customer Decision Hub property.

You can also map additional PII properties, such as device ID, address, phone number or date of birth.
7. **Optional:** Use the **Use External Id (EXTERN_ID)** field to specify a unique ID that can be used in place of hashed data for subsequent updates for a given individual.

This feature is enabled and set to use the class key from your Customer class by default. Keep in mind that disabling this setting will result in slower synchronization with Facebook.

The property used as external ID is not hashed during synchronization with Facebook. If you want to change it, select a property that can identify an individual uniquely without exposing their internal data.

8. **Optional:** Configure the additional settings in the **Facebook Ads assets generation** section:

- **Enable campaign generation** - See [Streamlining campaign generation for Next-Best-Action paid audiences](#).
- **Enable offline conversion push** – See [Enabling offline conversion push on Facebook](#).
- **Lookalike audience country** and **Lookalike audience ratio**- See [Prospecting with Next-Best-Action paid audiences](#).

9. Click **Apply**.

- [Enabling offline conversion push on Facebook](#)

Enabling offline conversion push on Facebook

Measure the impact of online advertising on real-world interactions by synchronizing individuals who have accepted actions with an existing offline event set that was previously configured on Facebook.

Facebook Offline Conversions provide a way of measuring the impact of Facebook ads on real-world outcomes, for example, store purchases or call center interactions. You can upload interaction data into Facebook Business Manager in order to measure the effectiveness of your online ads and the return you get on ad spend. Paid Media Manager provides an experimental Offline Conversions push to automatically synchronize conversion data with an offline event set.

1. Create an offline event set in Facebook Business Manager. The name of the event set must have the format *<OfferName_Outcome>*.

For example: If the CaptureResponse service is triggered for when the action name is *DellInspiron* and the outcome is *Accepted*, the name of the event set must be *DellInspiron_Accepted*.

For more information, see [Offline Conversions](#).

2. Log in to Pega Customer Decision Hub as an operator with access to App Studio.

3. In App Studio, click **Settings > Channels > Paid Media > Facebook Ads**.

4. Edit or create a destination.

5. In the **Facebook Ads assets generation** section, select **Enable offline conversion push**.

6. Click **Apply**.

During the next synchronization for this destination, metrics related to action acceptance are uploaded to Facebook Business Manager and linked to the offline event set.

Configuring Google Ads destinations

Google Ads Destinations enable ads to be targeted on Google Search, YouTube, Gmail, and other Google-owned media properties. Paid Media Manager integrates directly with Google Ads and the Google Customer Match API.

1. Create a Google Ads account and credentials.

For more information, see [Creating a Google Ads account and credentials](#).

2. Log in to Pega Customer Decision Hub as an operator with access to App Studio.

3. In App Studio, click **Settings** > **Channels** > **Paid Media** > **Google Ads**.
4. Click **Add destination**.
5. Fill out the following fields. For more information, see [OAuth2 Authentication](#).
 - **Destination name**
 - **Client ID**
 - **Client secret**
 - **Developer token**
 - **Client customer ID**
6. Click **Grant Access** and authenticate in the pop-up window. This generates the authentication token for OAuth.

 **Note:** If an error message shows during the authentication process, close the pop-up window and authenticate again. To provide a more detailed error message for users who might encounter the issue, open the **OAUTH2** service package in Dev Studio and change the **Service access group** value to `PegaNBAM:Agents`.
7. In the **Field mapping** section, map the Google Ads **Email (EMAIL)** parameter to a Pega Customer Decision Hub property.
8. **Optional:** Map the Google Ads **Phone (PHONE)** property to a Pega Customer Decision Hub property.
9. To target individuals based on their advertiser device ID, configure the additional settings in the **Device configuration** section. See [Enabling device ID targeting for Google Ads](#).
10. **Optional:** Configure the additional settings in the **Google Ads assets generation** section.
 - **Enable campaign generation** - See [Streamlining campaign generation for Next-Best-Action paid audiences](#).
11. Click **Apply**.
 - [Enabling device ID targeting for Google Ads](#)

Enabling device ID targeting for Google Ads

For Google Ads destinations, you can target your ads to individuals who use Android or iOS apps.

Google Ads can target individuals with ads in apps for Android or iOS. For example, you can use ads in third-party apps to promote Next-Best-Action-based actions relevant to the individual, using the customer's device ID obtained in your owned app.

To configure device ID targeting for a Google Ads destination:

1. Log in to App Studio.
2. Click **Settings** > **Channels** > **Paid Media** > **Google Ads**.
3. Edit or create a destination.
4. In the **Device configuration** section, configure the following settings:
 - **App ID** - Enter the ID of an Android or iOS app that you own. You can specify multiple apps.
For Android apps, the ID is the application package name from Google Play store. For iOS apps, the ID is a unique number provided by Apple. For more information about app IDs, refer to Android or iOS documentation.
 - **Device** - Enter the Pega Customer Decision Hub property used to identify the type of mobile device that the individual owns. To add more devices, click **Add device**. You can specify multiple devices for each of your apps.

5. Click **Apply**.

Configuring LinkedIn Ads destinations

LinkedIn Ads Destinations enable ads to be targeted on LinkedIn apps and websites. Paid Media Manager integrates directly with LinkedIn Ads and the LinkedIn Matched Audiences API.

1. Create and configure a LinkedIn account. For more information, see [Creating a LinkedIn account](#).
2. Contact Pega Support to authorize the connector. Include the Redirect URL in the Service Request. This is the URL that LinkedIn uses to access your Pega Marketing application. It has the format `http://<hostname>:<port>/prweb/PRRestService/oauth2/v1/redirect`. You can check the redirect URL for your application in the destination settings in Paid Media Manager.
3. Configure the connection to a LinkedIn Ads destination. For more information, see [Configuring LinkedIn Ads destinations in Paid Media Manager](#).

Configuring the connection to web-based and DMP destinations

In addition to modern, people-based platforms, Paid Media Manager also supports pixel-based APIs provided by advertising technology platforms, including demand-side platforms (DSPs), data management platforms (DMPs), and advertising networks. Instead of people-based APIs, these platforms generally rely on third-party cookies to build anonymous profiles and target individuals with ads. Paid Media Manager integrates with these mechanisms to deliver people-based Paid Audiences and Next-Best-Action decisions.

Typically pixel-based mechanisms target individuals with advertisements based on only the current content of the website which the individual is viewing. For example, if Troy goes to `uplusbank.com` and views information on a Platinum credit card, the pixel would then activate and target Troy with ads for the credit card. Paid Media Manager instead bases the ad targeting decision on Next-Best-Action. This takes into account the history of Troy's interactions and the learnings derived from them in order to show him an ad for a product which he has a high propensity to buy.

Unlike with people-based APIs where audience can be updated via S2S API calls at any time, pixel-based audiences can only be updated when an individual is in a live web session and on an advertiser owned website. Many browsers also prevent access by third party cookies, which may further limit the ability to update audiences through this mechanism.

- [Paid Media web service](#)
- [Enable Next-Best-Action targeting on DMP-based destinations](#)

Paid Media web service

Paid Media Manager allows you to include the power of Next-Best-Action decisioning into the pixel calls. You can extend the pixel call with a call to the `Paid Media` web service to determine ad targeting based on the enterprise decisioning and intelligence embedded in Next-Best-Action.

Before calling the pixel-based APIs for each platform, you can call the `PaidAudiencesContainer` REST service to retrieve the paid audiences to which the website visitor belongs based on their customer ID and the destination key. The URL pattern for this service is:

```
http://<host>:<port>/prweb/PRRestService/PegaMKTContainer/V1/  
PaidAudiencesContainer
```

For the GET request, the service expects a JSON Object with the following attributes:

CustomerID

Used to identify individuals based on information captured during their interactions with the destination. For individuals yet to be identified, a unique id is stored in a browser cookie. Once an individual has been identified as a customer, their encrypted customer id is stored. For more information, see "Identity Matching" in the *Pega Customer Decision Hub User Guide* on the Pega Customer Decision Hub product page.

DestinationKey

Because the same individual may have different action priorities and eligibilities per destination, Paid audience membership can vary from one destination to another. To check the destination key for a specific destination, in App Studio, click **Settings** > **Channels** > **Paid Media** > **Web**. Take note of the value in the **Destination key** column.

The paid web service is designed for low latency and does not execute Next-Best-Action for the site visitor which would update the individual's NBA Paid Audiences. However, the real-time container can be called separately to do so.

See below for a sample request and response:

Request:

```
{
  "CustomerID": "101",
  "DestinationKey": "DK6c378a63"
}
```

Response:

```
{
  "ResponseData" : {
    "DK6c378a63" : {
      "PaidAudiences" : [{
        "AudienceId" : "f9d23ce5"
      }],
      "pxSubscript" : "DK6c378a63"
    },
    "CustomerID" : "101"
  }
}
```

The `AudienceId` in the responses is generated by Paid Media Manager and associated with an audience generated in the paid destination. You can see the audience ID for each audience in the **AUDIENCE KEY** field on the Paid Media Audiences landing page.

Using the key instead of an audience name obfuscates the ad decision from the targeted individual. The actual audience name can be configured in downstream advertising platforms consuming the audiences.

Enable Next-Best-Action targeting on DMP-based destinations

Data Management Platforms (DMPs) provide a mechanism to update cookie-based audiences outside of a client web session. After a DMP establishes a cookie for an individual who visits an advertiser's website, subsequent updates can be made through S2S updates to the DMP. These updates can then be delivered to a separate ad platform responsible for selecting and bidding ad inventory, such as a Demand Side Platform (DSP). Each of these media buying platforms should have a separate destination to account for differences in paid action priority for an individual across different destinations.

Paid Media Manager provides out-of-the-box support for the Adobe Audience Manager DMP. Paid Media Manager generates the appropriate traits and audiences within Adobe Audience Manager (AAM) through the AAM API. After a scheduled run completes, a file is also generated with the audience updates

produced by Next Best Action. This file is delivered via S3 bucket or FTP to Adobe. Adobe Audience Manager can take up to 48 hours to propagate these updates to downstream DSPs and other third-party cookie-based platforms. In addition to Adobe Audience Manager, you can also configure paid destinations for a generic DMP.

For DMPs, updates are only made in response to scheduled runs and segments.

- [Configuring DMP destinations](#)

Configuring DMP destinations

You can configure one or more paid destinations for Adobe Audience Manager and other DMPs under Paid Media Settings in App Studio.

1. Log in to App Studio.
2. Click **Settings > Channels > Paid Media > Adobe Audience Manager**.
3. Click **Add destination**.
4. Fill out the destination name and description.
5. Select the destination type.
 - **Generic DMP**
 - **Adobe Audience Manager**
6. For Adobe Audience Manager, fill out the following required fields:
 - **Destination name**
 - **DP ID**
 - **Data source ID**
 - **API user name**
 - **API password**
 - **API client ID**
 - **API secret key**
7. In the **Sync Configuration** section, map the Adobe Audience Manager **Sync ID** property to a Pega Customer Decision Hub property that holds the value of the universally unique identifier (UUID) that identifies the individual.

The UUID property is implemented as part of the Adobe ID synchronization and works as a synchronization key, allowing the integration to match known individuals to their accounts and target them with the right ads on the right platform.
8. **Optional:** Configure the additional settings in the **Sync Configuration** section.
 - **Full synchronization** - Enable this setting if you want to send all Paid audience information to DMP destinations every time that the synchronization runs. Enabling full synchronization results in larger files being uploaded into the DMP. This can slow the synchronization. This setting is not enabled by default. When full synchronization is not enabled, the synchronization only sends information which has been updated in Pega Customer Decision Hub since the last synchronization date.
9. In the **Delivery Configuration** section, configure the location where Paid Media Manager sends files with synchronization data to be uploaded into the DMP.

This location may be an FTP location or an Amazon S3 bucket for the files.
10. Click **Apply**.

Enabling elastic search for Paid Media Manager

To improve the performance of the paid media synchronization and audience count reporting, enable elastic search on the paid media master database table.

Elastic search for paid media is used for reporting purposes. The elastic search index contains data about the paid media synchronization and audience counts, excerpted from the master database table used by Paid Media Manager. In this way, reports can draw upon a smaller set of data, which increases their performance.

The PaidMediaESCheckAndReindex job scheduler periodically checks the elastic search index for any conflicts with the master database table and rebuilds the index if required, to ensure that reporting data remains up-to-date. By default, the job scheduler is disabled and set to run once a day once you enable it.

1. Log in to Dev Studio.
2. Click **Configure > System > Settings > Search**.
3. In the Security section select **Customize full text search**.
4. Search for and open the PaidMediaESCheckAndReindex job scheduler.
5. Review the schedule and adjust it as required.
6. Click **Enable Job Scheduler**.
7. Save your changes.

Marketing assets configuration

Most Pega Customer Decision Hub rules are not executable. This is because they are metadata rules and have no generated run-time behavior. The metadata that these rules carry with them is used to do work on a user's behalf.

Implementation layer rules reside in one of the two implementation layer rulesets: *Rules* and *Artifacts*. To configure your marketing assets, you must create instances of any of the rule types listed below within the relevant Business Issue/Group context. For more information on configuring marketing assets, see the [Pega Marketing User Guide](#) on the [Pega Marketing product page](#).

Rule types in the Artifacts ruleset

The first set of rule types reside in the Artifacts ruleset and can change based on the needs of the marketing team. These rules can be divided into two groups, depending on where they are automatically published.

You can find these rules in the Application Explorer. For more information, see [Finding rules by class](#).

- Rule types that are automatically published to an internal Pega Customer Decision Hub class *PegaMKT-Work-Offer-**<issue>-<group>***:
 - Offer: *Rule-Obj-Flow-Offer*
 - Email: *Rule-Obj-Treatment-Outbound*
 - SMS: *Rule-Obj-Corr-SMS*
 - Passbook: *Rule-PegaMKT-Passbook*
 - Volume Constraint: *Rule-Obj-VolConstraintGrp*
 - Real-Time Event: *Rule-PegaMKT-Event*
 - Real-Time Container: *Rule-PegaMKT-Container*

- Output Template: *Rule-Obj-Template*
- Geofence: *Rule-PegaMKT-Geofence*
- Rule types that are automatically published to your configured customer class:
 - Segment: *Rule-Obj-Report-Definition-Segment*
 - Strategy: *Rule-Decision-Strategy*
 - Analysis Project: *Rule-Obj-Analysis*
 - Sample Population: *Rule-Obj-Report-Definition-Segment-Sample*
 - Contact Policy: *Rule-PegaMKT-ContactPolicy*

Revision management

Revision management enables marketing users to revision, package, and import marketing artifacts. They can make controlled changes in the business sandbox and test these in production within the boundaries defined by IT. The users can see their assigned change requests in the My Work and Revision Management widgets.

For more information, see [Revision management](#).

Rule types in the Rules ruleset

The second set of rules are saved to a Rules ruleset in your configured customer class and are typically not changed after the implementation is complete.

You can find these rules by following the hierarchy and rules under the newly-created customer class in the Application Explorer.

- Customer: *Rule-Obj-Property*
- Segment: *Rule-Obj-Association*



Note: If you change the customer class midway through the implementation process, the previously-created segments or strategies no longer display in the Marketing portal. To access them in Dev Studio, you must navigate to the original customer class.

Optional: Materializing interaction history summaries

Pega Customer Decision Hub contains interaction history summaries, which are shipped without preaggregated data. To save processing time, turn on preaggregation for each interaction history summary by materializing them.

Preaggregated data sets save processing time because they include the latest interactions. Data sets that are not preaggregated do not include the latest interactions and therefore they query the database.

1. Log in as an operator with access to the Summaries landing page.
Access to the Summaries landing page is limited to users whose access group includes the CustomerDecisionHub:DataAdmin user role. For more information about access roles, see [Defining access roles and privileges](#).
2. In App Studio, click **Data > Summaries**.
3. On the Summaries landing page, find the data set for which you want to turn on preaggregation, and click **More** **> Materialized**.

Verifying the System Health page

After building your implementation features, check the System Health page for any errors or warnings.

1. Log in to the Pega Customer Decision Hub portal.
2. Click **Configuration > Settings > System Health**.
3. Review and address any errors or warnings.

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. 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.
3. 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.
4. Test all integrations, both independently and with associated integrations.

Test integrations for any optional Pega Customer Decision Hub 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.

5. 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 Customer Decision Hub 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.

Additionally, the revision management capability allows business users to make controlled changes in the business sandbox and test the changes in a production environment within the boundaries defined by IT. For more information, see [Revision management](#).

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:

- [Application health monitoring](#)
- [Identifying and reporting issues](#)
- [Revision management](#)

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.
- Monitor social accounts (for example, your Facebook and Twitter accounts) to ensure that they do not expire.
- Regularly perform a health check using the System Health diagnostic tool. Resolve any issues shown as errors, and review warnings. For more information, see the System Health appendix in the *Pega Marketing User Guide* on the [Pega Marketing product page](#).
- Ensure that you are aware of the agents and their characteristics. This helps assess the impact of any possible failures in these agents. For more information, see [Pega Marketing agents](#).
- Regularly check the email accounts configuration. This helps avoid unnecessary exceptions and errors. Ensure that all email accounts are connected to a working email server, and that there are no outbound email accounts configured that are not needed.
- Ensure that all Pega Platform operator accounts used by your Pega Customer Decision Hub application have a valid email address. This also applies to the accounts of managers, system administrators and application administrators.
- Configure the System Management application with access to the nodes in the system. For more information, see [Managing your system](#).

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.

Revision management

Revision management is a Pega Platform feature that enables business users to take ownership of parts of the enterprise application. Users can make controlled changes in the business sandbox and test the changes in a production environment within the boundaries defined by IT.

Most marketing artifacts can be included and updated in revisions. Revision managers can start a new revision and create change requests within the revision. A change request can contain marketing rules and be assigned to a marketer.

These two widgets can be used by marketers to view their assigned change requests:

- The My Work widget
- The Revision Management widget
 - Added to a user's Home page in the Pega Customer Decision Hub portal
 - Displays change requests that are assigned to the user, and lists each rule within the request
 - Enables a user to directly access the rules within an assigned change request

There are three roles involved in revision management:

- `DecisionManager:RevisionManager` - includes the ability to create a revision and change request.
- `DecisionManager:StrategyDesigner` - can be assigned a change request to work on.
- `DecisionManager:PegaMarketing` - includes both of the above roles to allow running campaigns from the application overlay context.



Note: Even when revision management is used, an artifacts ruleset with an unlocked ruleset version is still required to store rules generated at runtime.

For more information, see the *Revision Management* chapter in the *Pega Marketing User Guide*, available on the [Pega Marketing product page](#).

In addition to revision management, 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. After the rule has been delegated, you can access it by clicking **your operator profile** > **My rules**. For more information on rule delegation, see [Delegating a rule or data type](#).

Managers can also create reports in the **Reports** section of the portal. For more information, see [Report Browser features](#).