

PEGA PLATFORM



Pega Enablement

Business Architect Essentials

8.7

Student Guide

SKILL

BUILD
FOR
CHANGE

The Classroom Experience



© 2022 Pegasystems Inc., Cambridge, MA All rights reserved.

Trademarks

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

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

Notices

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

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

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

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

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

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

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

This document is the property of:

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

DOCUMENT: Business Architect Essentials (BAE) Student Guide
SOFTWARE VERSION: 8.7
UPDATED: December 23, 2021

Pega Infinity Overview



SKILL
LESSON

What is Pega?

Introduction to Pega





Overview

Pega's low-code technology allows you to build an application that captures data and initiates business processes, while ensuring you are following organizational best practices and standards.

The screenshot shows the Pega Studio interface with the following elements:

- Header:** STUDIO, Application: MyTown 311, Configure, Launch portal, Create.
- Background:** A dark blue background featuring a night sky with stars, clouds, and a crescent moon.
- Guardrail warnings (last 7 days):**

	Severe	Moderate	Informational
Introduced by you	0	0	0
Introduced by team	0	0	0
- Security status:** 0 out of 33 security tasks completed. Review the Application Security Checklist and Application Guides.
- Bottom navigation:** Issues, Tracer, Clipboard, Live UI, Live Data, Accessibility, Performance.

Every day, Pega powers

MILLIONS
of automated processes
BILLIONS
of customer interactions
TRILLIONS
of dollars of business

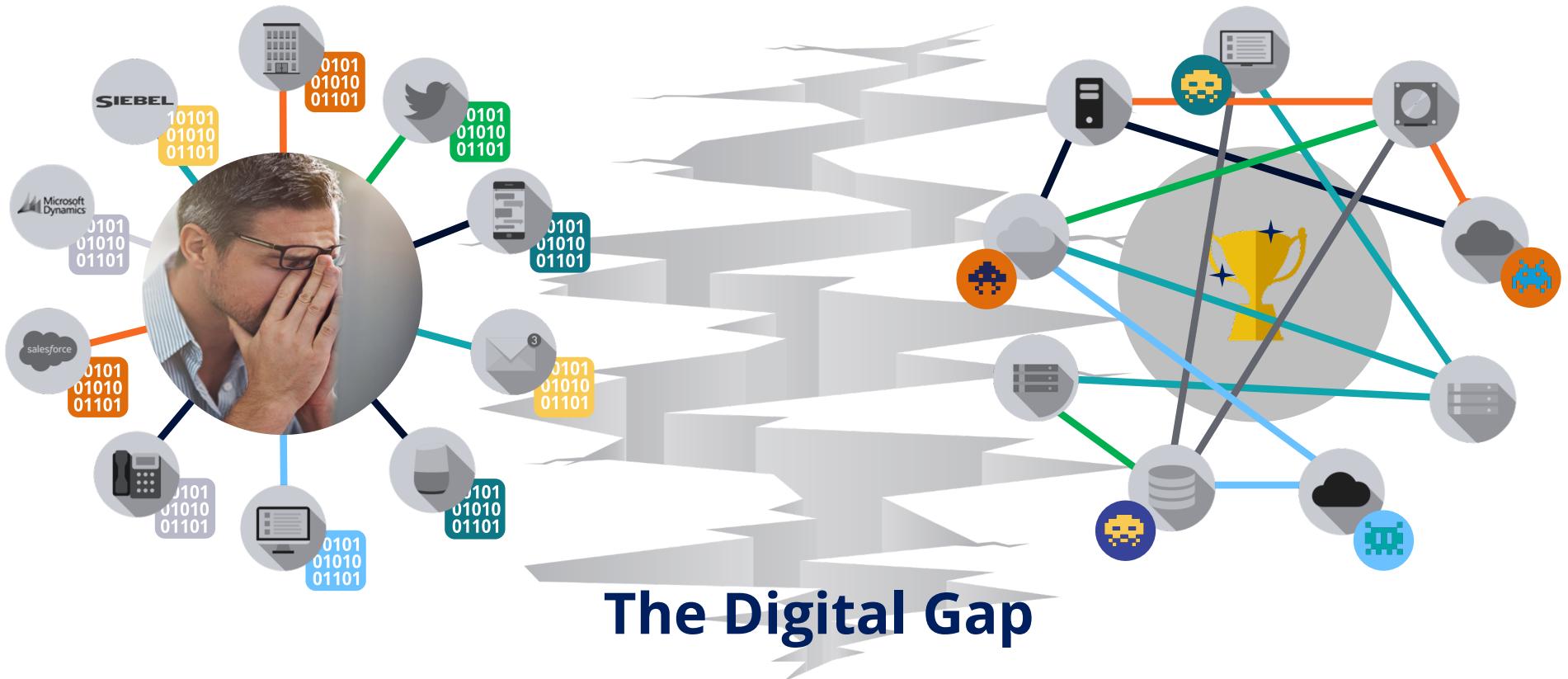
If you've driven a car, used your credit card, called a company for service, opened an account, applied for a loan, accepted an offer, flown on an airplane, paid a bill, submitted a claim, or countless other things you do in your day...

...you've interacted with Pega.





3 well-intentioned mistakes = The Digital Gap



Channels, not journeys

1

SYMPTOMS

- Customers can't move between channels
- Siloed development team's custom-coding logic into channels

RESULTS

- Bad, inconsistent experiences
- Frustrated employees and inefficiency
- Lack of reuse drives increased costs and freezes innovation



Tasks, not outcomes

2

SYMPTOMS

- Lack of common processes across systems, regions, and LOBs
- Siloed investments in robotic band-aids

RESULTS

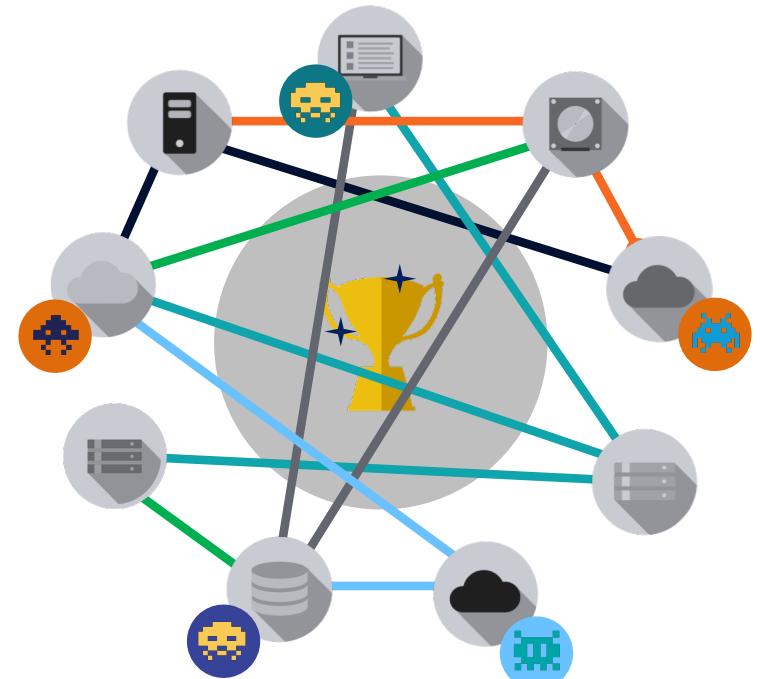
- Disjointed customer experience
- Proliferation of unmanaged bots
- Little visibility into improvement opportunities





Silos, not end to end

3





Pega Infinity™

Revolutionary software that unifies customer engagement and intelligent automation

Pega Customer
Decision Hub™

Pega Customer
Service™

Pega Sales
Automation

**CUSTOMER
ENGAGEMENT**



**INTELLIGENT
AUTOMATION**

Pega Platform™

- Case Management
- Low-code App Dev
- Mobile

Pega RPA™



REAL-TIME,
OMNI-CHANNEL
AI



END-TO-END
AUTOMATION &
ROBOTICS

Industry-leading technology



MICROJOURNEY-
CENTRIC RAPID
DELIVERY



SITUATIONAL
LAYER CAKE™

Start fast and scale



SOFTWARE THAT
WRITES YOUR
SOFTWARE™



CLOUD
CHOICE

Future proof your investment

PEGA DX ARCHITECTURE™



Pega Platform products

Description

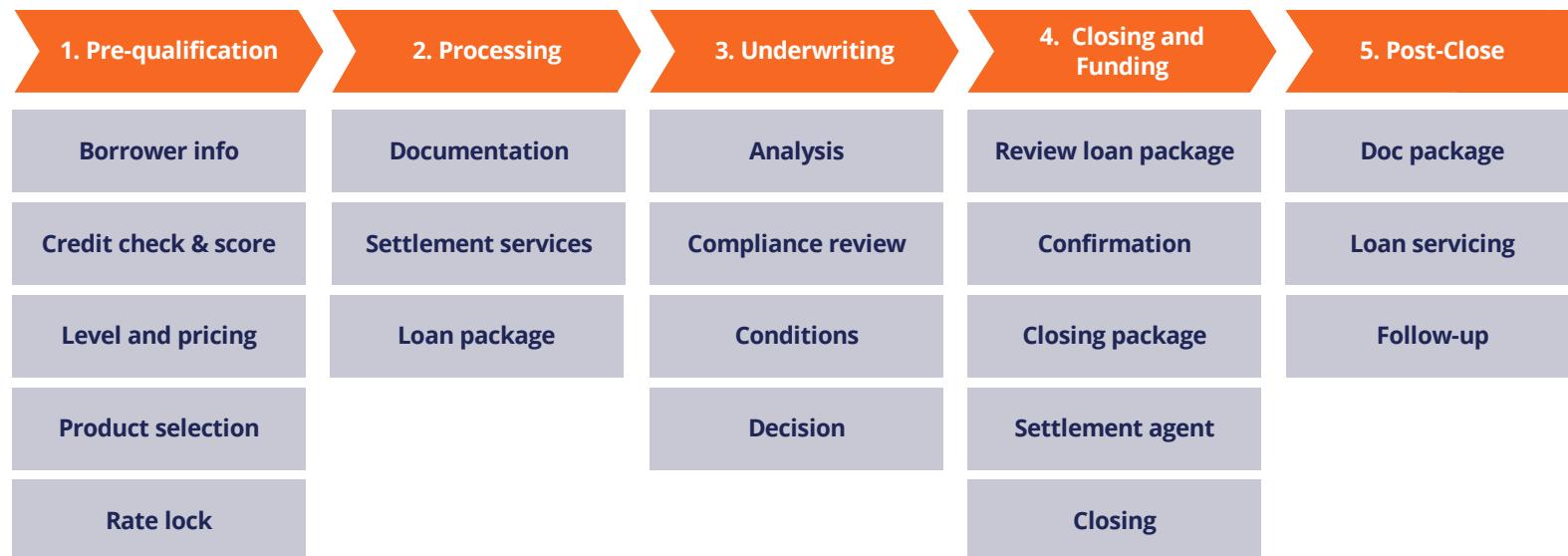
- The main use of Pega technology is to reduce cost and to improve business processes
 - **Pega Customer Decision Hub** helps organizations make an offer, initiate a retention plan, predict a problem before it happens, generate the next-best-action for your customer or business.
 - **Pega RPA** is robots running processes and building bridges between unrelated software systems to automate tedious manual work and repetitive tasks.
 - **Pega Sales Automation for insurance, for financial services and for healthcare** uses AI capabilities to identify the best leads, opportunities, and personalized Next-Best-Actions for each sales representative.
 - **Pega Customer Service for communications, for financial services, for healthcare and for insurance** uses AI to determine the Next Best Action for a customer and guide an agent through an interaction.
 - **Pega Predictive Diagnostic Cloud** an AI-powered technology used throughout the application life cycle to assess the health of your application, notify you of critical issues and resolve performance and stability problems





Design your microjourneys with stages and steps

Account Opening | Loan Origination

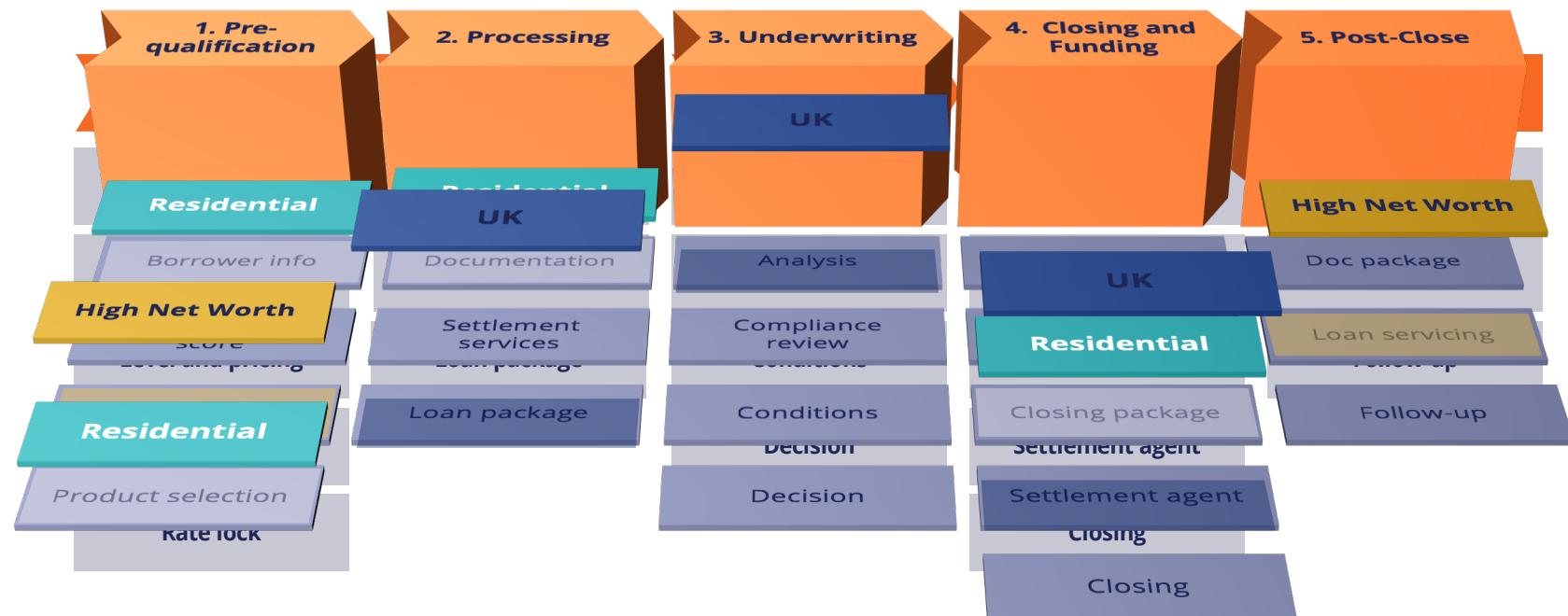


MICROJOURNEY-CENTRIC
RAPID DELIVERY



Situational Layer Cake explained

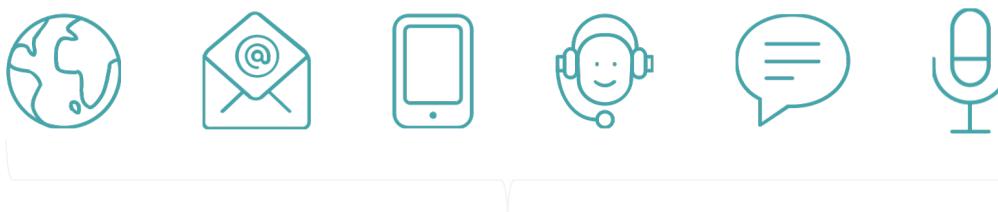
Account Opening | Loan Origination



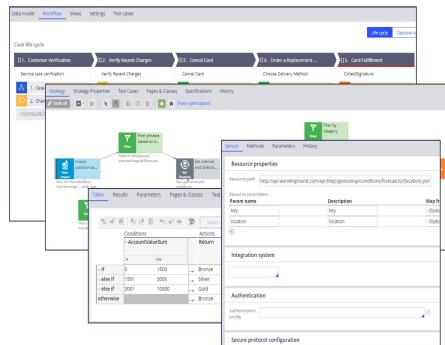


Case-Centric Approach

Enabling consistent customer outcomes across channels



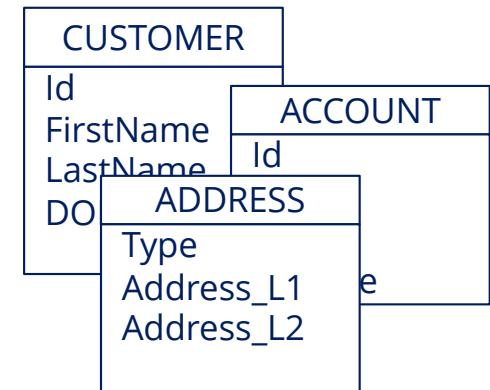
Models



Case



Data Types





Seamless Channel Distribution

Think channel-less

- Think experience/process first
- Easily expose multiple channels to the experience/process
- Reuse and extend existing apps with mashup
- Integrate with existing mobile assets
- Extend to any conversational interface

The screenshot shows the Pega Express application interface for "Internal Travel Approvals". The top navigation bar includes "Pega EXPRESS", "Application: Internal Travel Approvals", "View in channel", and a dropdown menu. On the left, a sidebar has "Design" selected, along with icons for "Cases", "Data", "Users", and "Settings". The main area is titled "Add new" and shows icons for "Web", "Mobile", "Mashup", "Facebook messenger", "Email", "Alexa", "SMS", and "API". Below this, a section titled "Current channels" lists several channels:

- FB travel** (Connected): Described as a channel for customer support to resolve cases involving updating information and common requests such as: Recent transaction, Pay bill and more. It includes "Case types" (Open Account, Update Profile, Lost Luggage Claim) and "Responses" (Check my balance, Chat with agent, Help).
- Support box** (Published): An email channel used for customer support to resolve cases involving updating information and common requests. It includes "Suggested cases" (Lost Luggage, Travel request).
- Employee** (Published): Employee web portal with worklists, and travel cases processing and resolution.
- Approver** (Published): A custom UI channel for approvers.
- Employee** (Published): Another Employee channel, likely for different roles.



Unified Design-Time Environment

Browser-based authoring of all application components:

- Shared view of application assets
- Consistent “Tooling”
- Role-based Workspaces
- Integrated version control
- Auto managed sandboxes
- Ability to create unit tests/suites
- Support for branching and merging
- Application Quality Reports

The screenshot displays the Pega Platform's unified design-time environment across three interconnected studios:

- PREDICTION STUDIO:** Shows a list of "Predictions" with filters for Type, Status, and Updated by.
- ADMIN STUDIO:** Shows an "Overview" of the application environment.
- DEV STUDIO:** Active studio, showing the "Overview" page. It includes sections for "Channel interfaces" (Case Manager, Pega API), "Case types" (No case types), and "Application Layers". The "Application Layers" section features a 3D perspective diagram illustrating the layered architecture of the Pega Platform.

The interface is designed for role-based workspaces, with a sidebar containing links for Overview, Case types, Data, Interfaces, Pages, Users, and Settings.



No Code/Low Code Development

Description

Business friendly models define application behavior.

Auto-generated code

- HTML5 / CSS3
- JavaScript, Java
- SQL

Benefits

- Fastest path to value
- Simplify communication between the business and technical teams
- Future-proof architecture

The screenshot displays the Pega Platform interface. At the top, a navigation bar includes 'Data model', 'Workflow' (which is selected), 'Views', 'Settings', and 'Test cases'. Below this is a 'Case life cycle' section with five steps: 1. Customer Verification, 2. Verify Recent Charges, 3. Cancel Card, 4. Order a Replacement ..., and 5. Card Fulfillment. Step 2 is currently active. A 'Strategy' tab is selected in the main workspace, which contains a 'Data Import' node and a 'Filter' node connected to a 'Set Property' node. To the right, a 'Service' configuration panel is open, showing resource properties for a service path. The 'Resource path' is set to `http://api.wunderground.com/api/{key}/geolookup/conditions/forecast/q/{location}.json`. The 'Resource parameters' table includes columns for 'Param name', 'Description', and 'Map from'. It has two entries: 'key' (Description: key) and 'location' (Description: location). Below this are sections for 'Integration system', 'Authentication', and 'Secure protocol configuration'.



Future-proof technology

- Business and IT collaborate in a single model-driven environment
- Proven to deliver 12X faster than traditional coding
- Automatically generates documentation

The screenshot shows the Pega App Studio interface with the title "APP STUDIO" and "Application: HR hub". The main area displays a "Microjourney" for a "Job application". The journey is divided into four stages: 1. Collect resume, 2. Screening, 3. Interview, and 4. Decision. Stage 1 has two parallel steps: "Submit web" and "Submit Mashup". Stage 2 has three steps: "1. New submission", "2. Phone screen", and "3. Screening outcome". Stage 3 has two steps: "4. Schedule interview" and "5. Send reminders". Stage 4 has one step: "Decision". Below the journey, there are sections for "Personas & Channels" and "Data". The "Personas & Channels" section lists personas like "Candidate HR portal", "Recruiter HR portal", "Recruiter HR GO mobile", "Candidate Facebook", and "Manager HR portal", each associated with "MLP 2" or "FUTURE". The "Data" section lists data objects like "Applicant info Pega", "Applicant info LinkedIn", "Resume Pega", and "Cost Center SAP", each associated with "MLP 1", "MLP 2", or "FUTURE". On the right side of the interface, there are buttons for "Actions", "Save and run", and "Save". A sidebar on the left includes links for Overview, Case types, Data objects, Channels, Users, and Settings.



**SOFTWARE THAT
WRITES YOUR
SOFTWARE™**



**CLOUD
CHOICE**



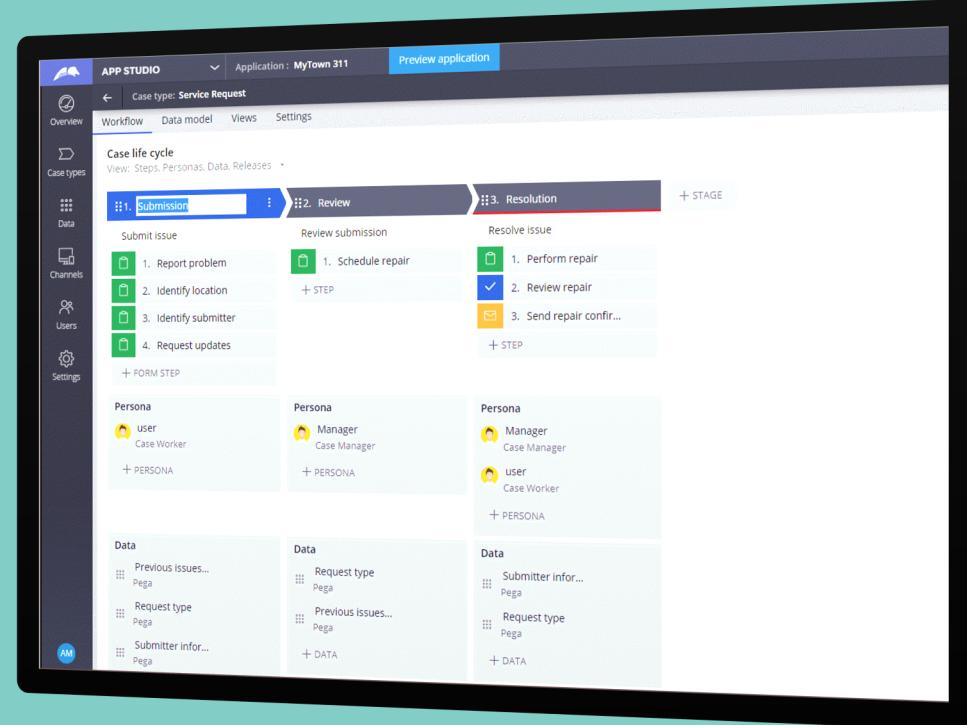
SKILL
LESSON

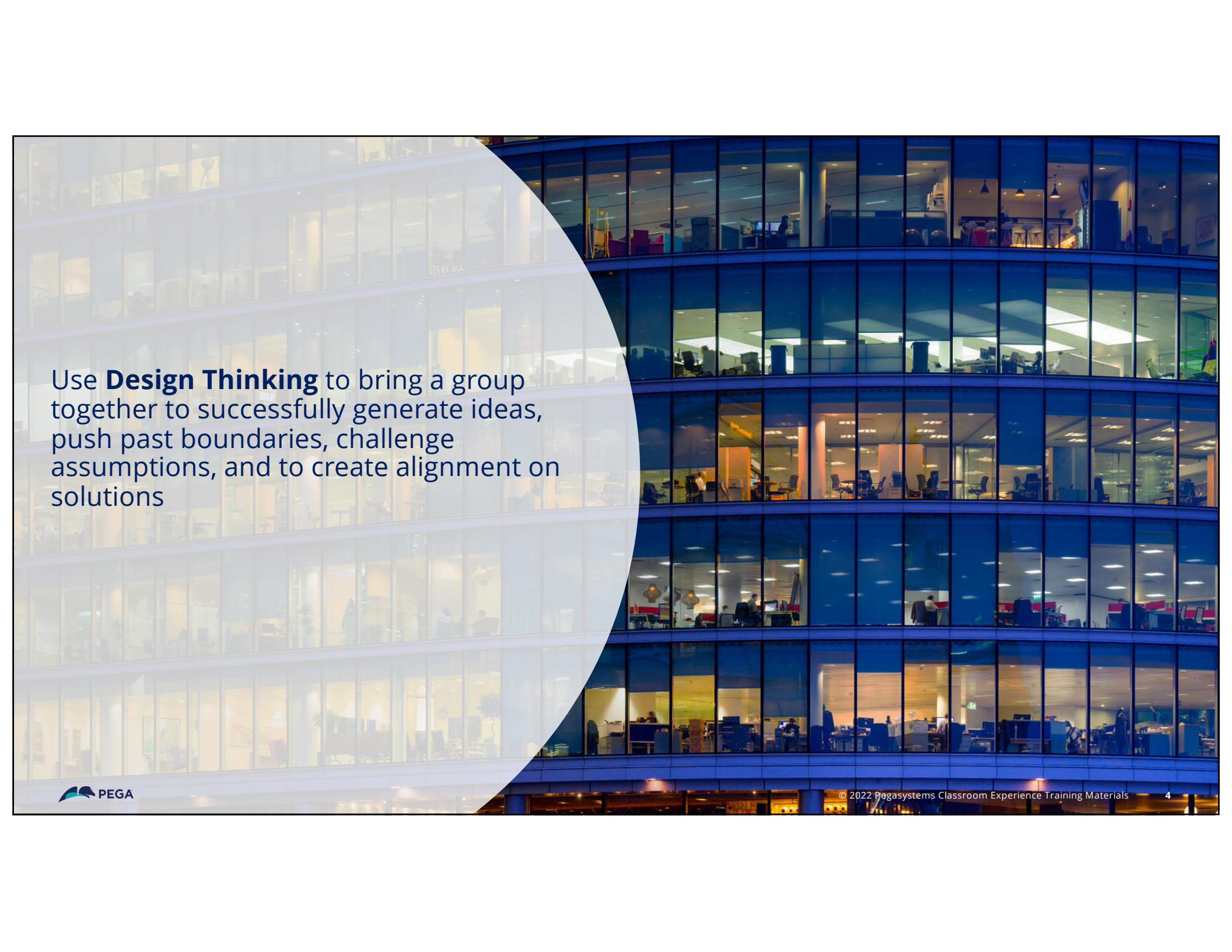
Design Thinking



Overview

Design thinking is a broad term that covers the overall philosophy and approach to problem solving and innovation.



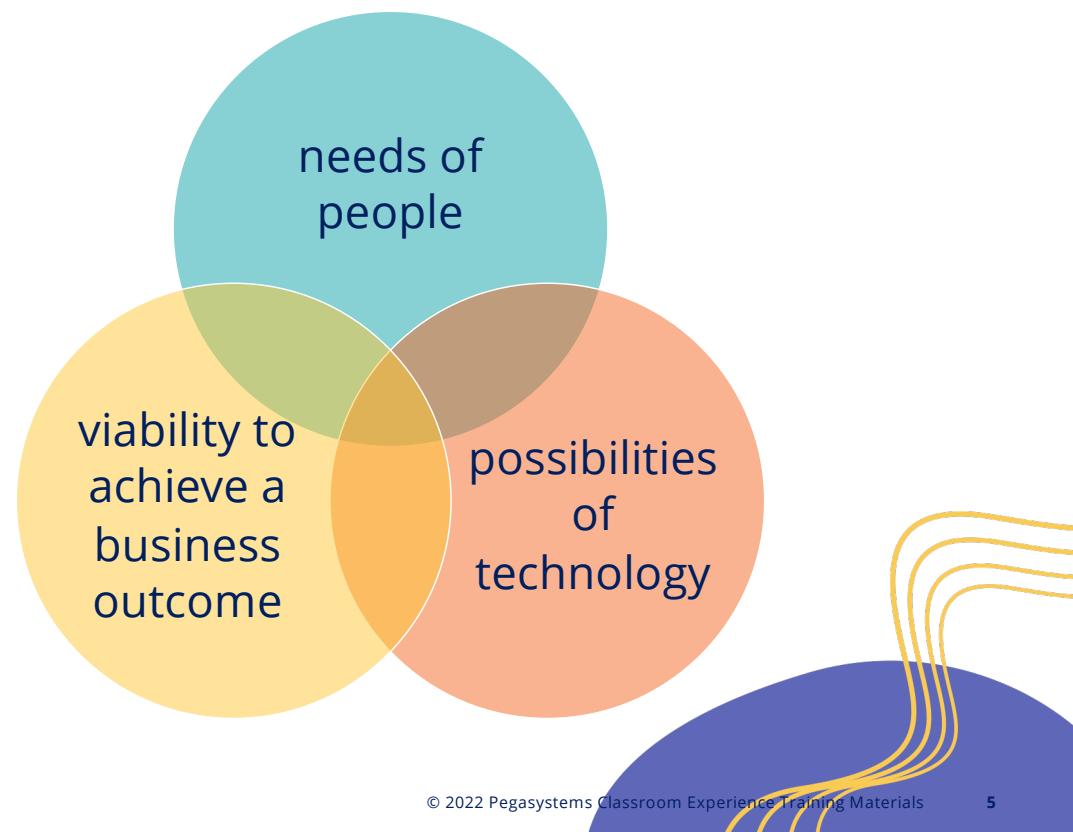


Use **Design Thinking** to bring a group together to successfully generate ideas, push past boundaries, challenge assumptions, and to create alignment on solutions

Design Thinking

Definition

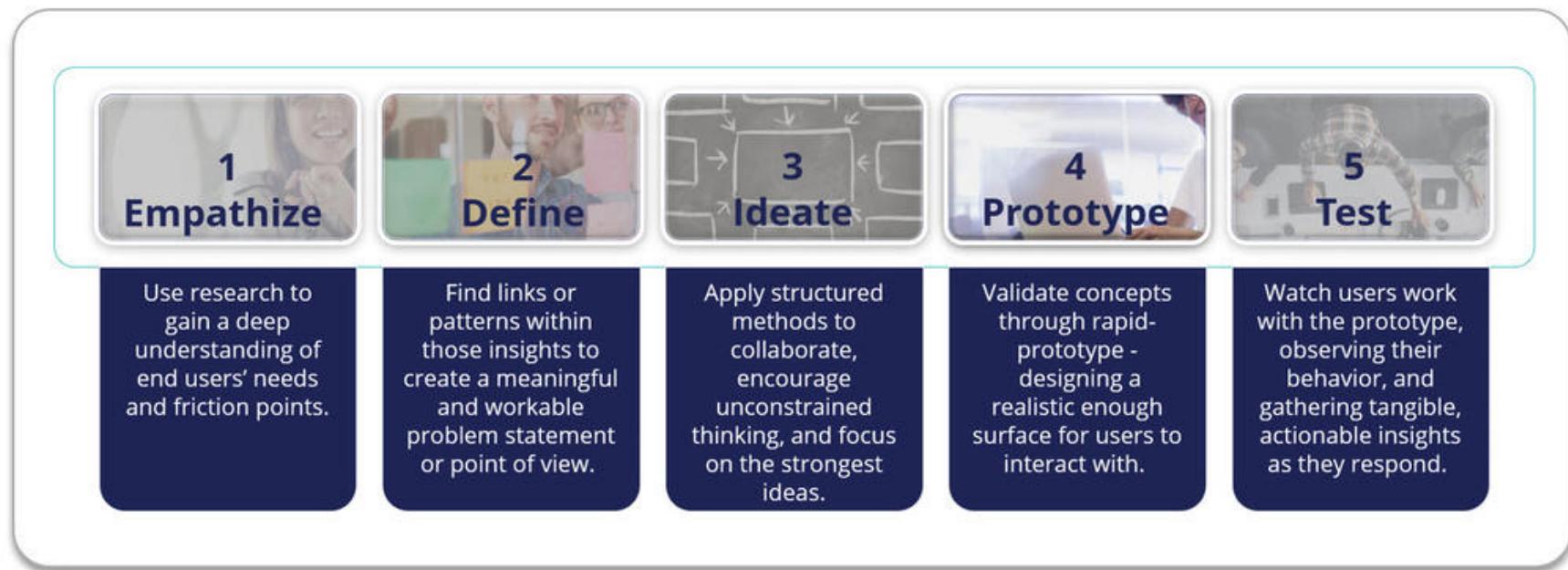
- Design thinking is a human-centered approach to identify and creatively solve problems throughout the project lifecycle
- It incorporates
 - the needs of people
 - the possibilities of technology
 - the viability to achieve a business outcome



Design Thinking - Five Steps

Description

- Design thinking brings a group together to successfully generate ideas, push past boundaries, and challenge assumptions



Design Thinking Tools and Techniques

Description

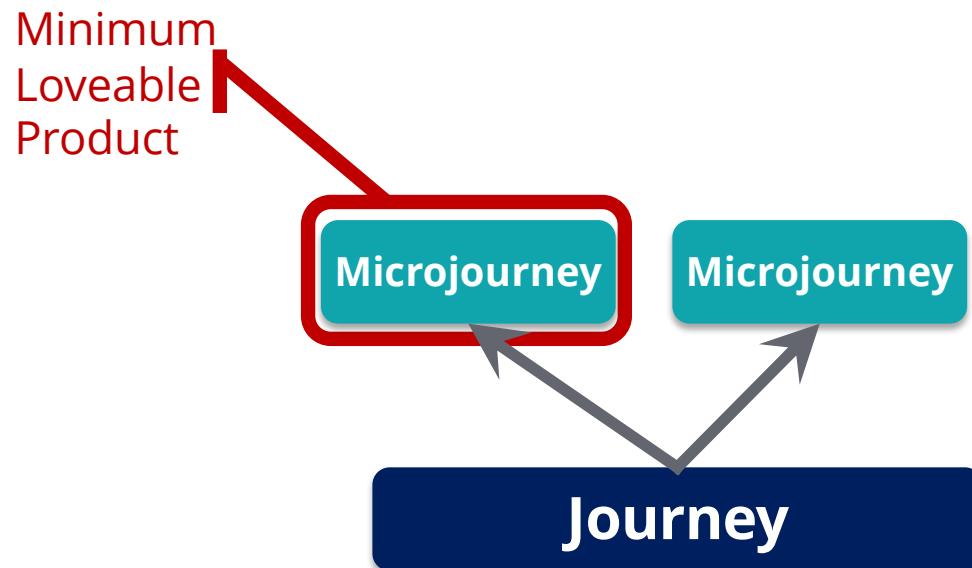
- **Design Sprint and Ideation**
 - A **Design Sprint** is typically a 5-day workshop that uses Design Thinking techniques to solve a problem using creative methods and innovation.
 - It's a way of learning quickly what will and won't work, and it's proven to be very powerful. Amazon, Airbnb, Google make extensive use of Design Sprints
 - A **Design Sprint** is a design thinking technique that is proven to help identify the microjourney solution that best achieves the client's business outcomes.
- **Design Sprints** are a great way of bringing both business, IT and users together to design and prototype a solution. A solution that is simple, intuitive and easier to implement.
- A **Design Sprint** is a concentrated, collaborative set of design thinking activities that inject a burst of energy into the team, launching the project in the right direction early on.



Identify Your MLP

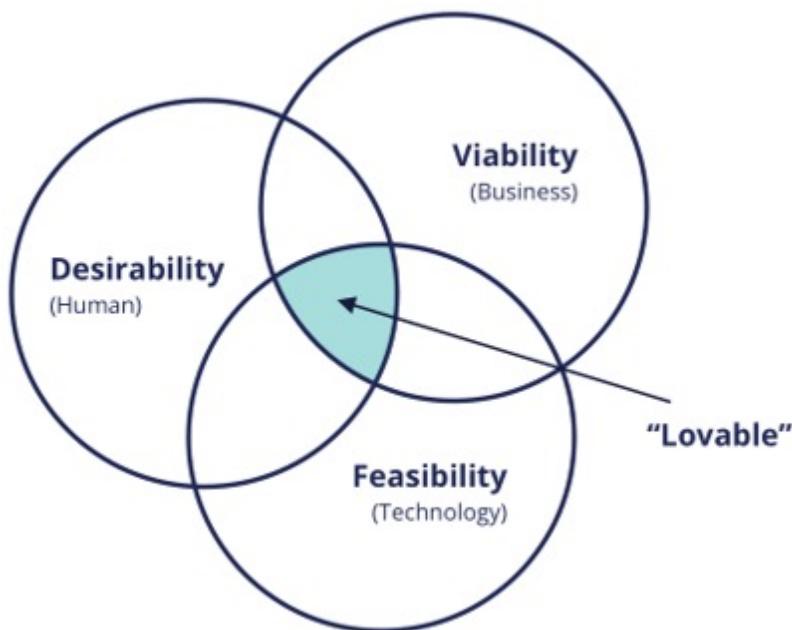
Definition

- Minimum Loveable Product (MLP)
 - The simplest solution that can be delivered quickly to delight your (or your client's) customers.
- MLPs are composed of one or more Microjourneys supported in the MLP release.



Minimum Lovable Product

Description



Design thinking helps unlock innovative solutions to create a **Lovable** solution by evaluating the following:

1. Desirability

- Focus is on designing solutions that are desirable for the people that use them

2. Viability

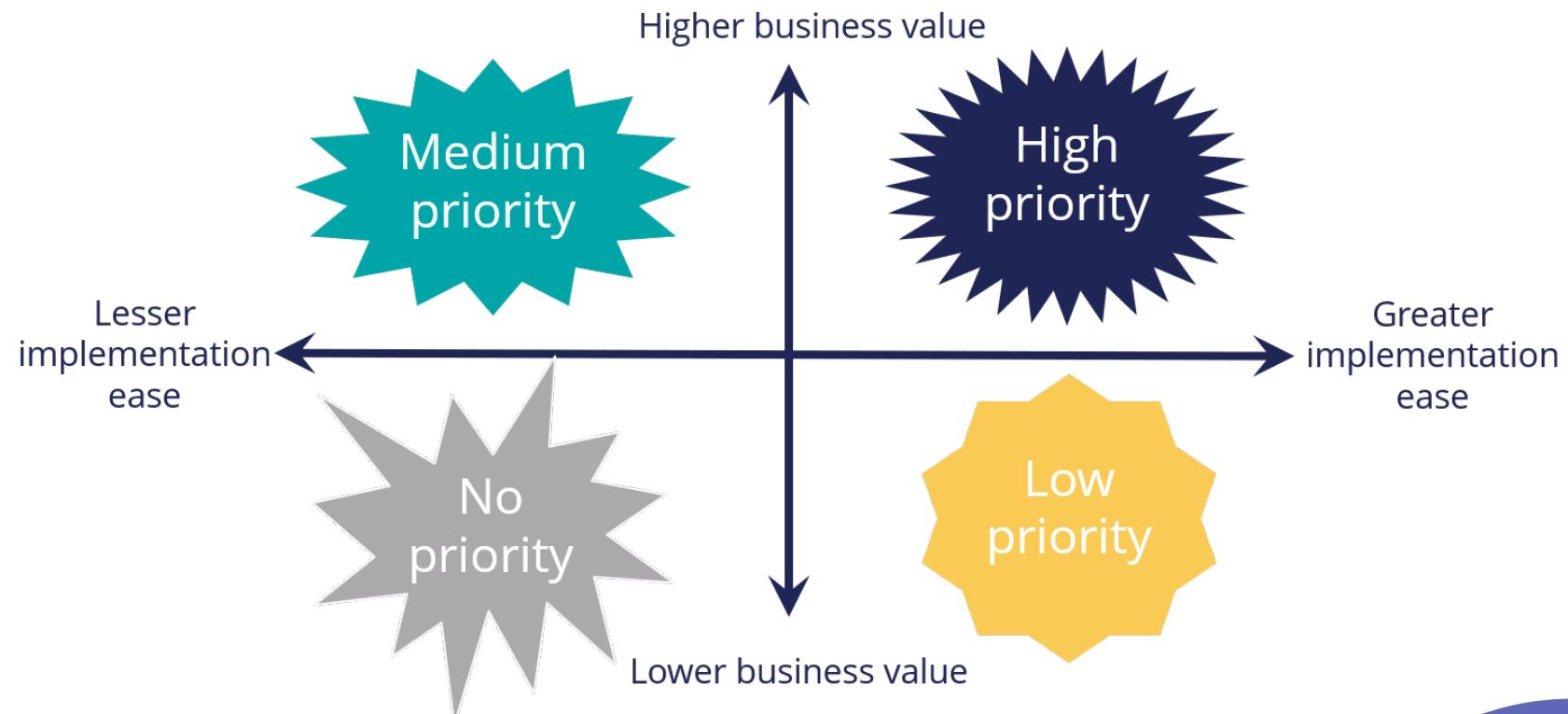
- Focus is on the business viability of projects for which the sponsors are paying to ensure return on investment

3. Feasibility

- Focus is on technical feasibility and effort of developing the software

MLP Priority

Implementation



Direct Capture of Objectives

Definition

- Direct Capture of Objectives (DCO) is a core part of Pega Platform™ and Pega's methodology approach.
- DCO tools are intended to:
 - Facilitate collaboration among project stakeholders.
 - Reduce the time between obtaining requirements, design, and implementation.
 - Focus on desired business outcomes.
 - Speed up and simplify application development.



DCO Application

Description

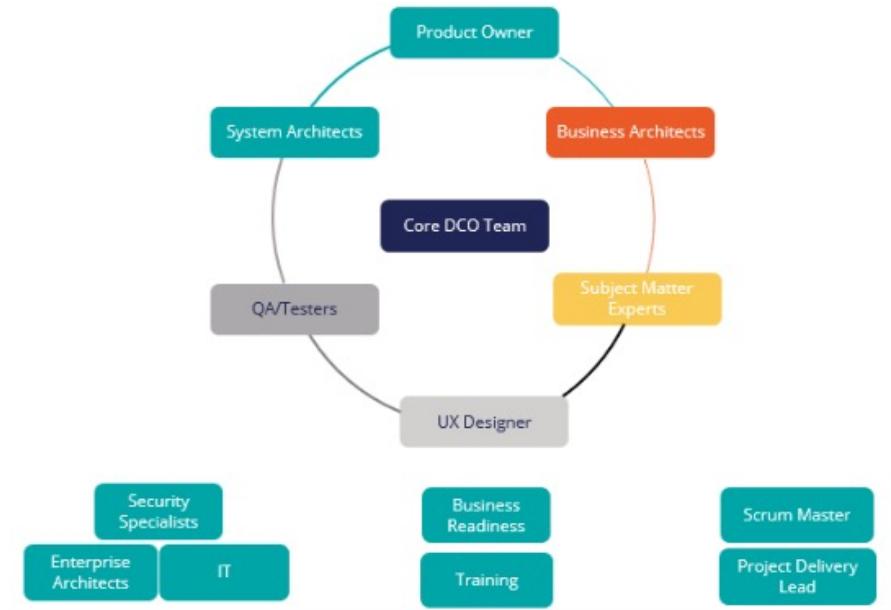
- DCO is used continuously within the Pega Express methodology.
- Use DCO to confirm and convert the three pillars with Pega Express
 - Microjourneys
 - Personas and channels
 - Data and integrations
- Although different activities occur during each phase (Discover, Prepare, Build, and Adopt), all events happen collaboratively, while iteration and validation occur throughout, as illustrated in the following image.



DCO Participants

Description

- The core DCO team must be small, with representation from the business, including the decision-maker (product owner), business architect, system architect, tester, and experience designer.
 - Notice these stakeholders come from a variety of disciplines — business to technology.
 - For some projects, the product owner may need additional support from business specialists.
 - You can also add subject matter experts (SMEs) to your core team.
- In an ideal world, all team members can attend every session, but this may not be feasible.
- Some topics may not require a system architect. Others may not require a experience designer.
- Others may require support from specific SMEs.



Application of DCO tools

Description

DCO-related Tool	Purpose	Benefits
Case Designer	Provides the ability to design the stages and steps of your process or journey. Capture the users and data applicable to each stage, as well as details such as routing and service levels.	Immediately run your process to verify and adjust the case design.
Agile Workbench	Provides the ability to capture and manage features, user stories, defects, and feedback items while working with the application. Integrate Agile Workbench with an agile project management tool such as Pega Agile Studio, or a third-party tool such as Jira, for features such as sprint execution and reporting.	<ul style="list-style-type: none">• Create and edit items while working in the application rather than having to access another system.• Capture a screenshot or video to include with the item.
Agile Studio	Provides agile project management functionality, including the ability to create and manage backlogs, plan and execute sprints, collaborate on backlog items, and monitor results with reporting.	You can support and manage Scrum-based development teams.

Real-time, continuous collaboration

Navigation

The screenshot displays the Pega Case Manager interface on a tablet device. The top navigation bar includes 'APP STUDIO', 'Case Manager', 'Publish', and various system icons. A search bar and a 'Turn editing on' button are also present.

The main content area shows a case titled 'External Training Request (T-15) PENDING APPROVAL'. The status bar indicates '1 day 23 hours from now'. A progress bar at the top shows the workflow steps: 'Request' (green), 'Review' (blue, currently active), and 'Approved' (grey).

Approve (DUE IN 4 DAYS FROM NOW): Please approve or reject this External Training Request. Requested by Dennis Grady.

Review:
Name: PegaWorld 2019 | Start date: 6/2/2019 | Training cost: \$895.00
Type: Conference | End date: 6/5/2019 | Travel cost estimate: \$2,000.00
Link: <https://www1.pegaworld.com/events/pegaworld> | Number of days: 3 | Total cost: \$2,895.00

Business value:

1. A roadmap for Digital Transformation.
2. It's the world's largest Intelligent Automation event.
3. Hands-on access you won't get anywhere else.
4. Unfiltered insights from top brands.
5. Surround yourself with greatness.

Notes: Enter approval or rejection comments.

Case details: Last updated by Dennis Grady (3m ago); Created by Dennis Grady (6m ago).

Open assignments: Get Approval via Email (Current) Administrator.

Attachments (1): External Training Request (T-15) fro... (Jun 01, 2018 02:27 AM)

Participants:
Administrator (A)
Dennis Grady (DG, Owner)
Administrator (A, Manager)
Manage

Skill Mastery

- Understand Design Thinking philosophy and approach to problem solving and innovation
- The composition of a MLP
- Understand the intent of DCO



Case Management Introduction

A business view of work

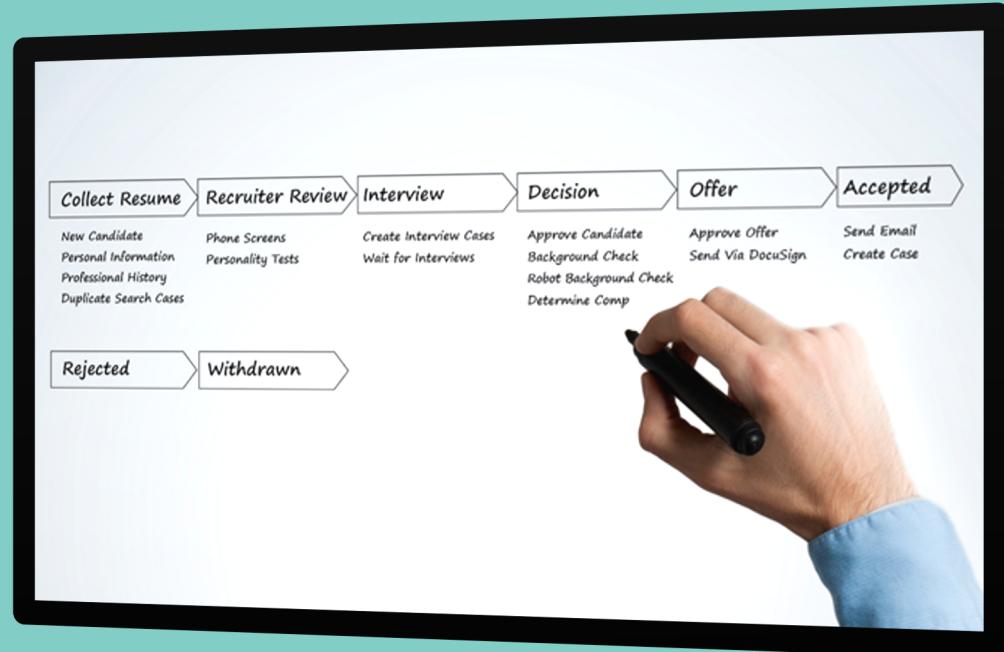
SKILL
LESSON

A decorative graphic on the right side of the slide features a teal base with several concentric yellow arcs radiating from the top right corner. A yellow rectangular callout bubble is positioned in the upper right area of the teal base, containing the text "SKILL LESSON".

Overview

Case management facilitates resolving business processes and guiding CSRs to the information and tasks most relevant to their roles in the case.

The Pega Platform allows you to model, monitor, track, and manage your business processes using a visual representation of your business cases grouped into related tasks and stages.



Use Case

Use Pega's case management to model a business transaction using a non-traditional model driven approach.



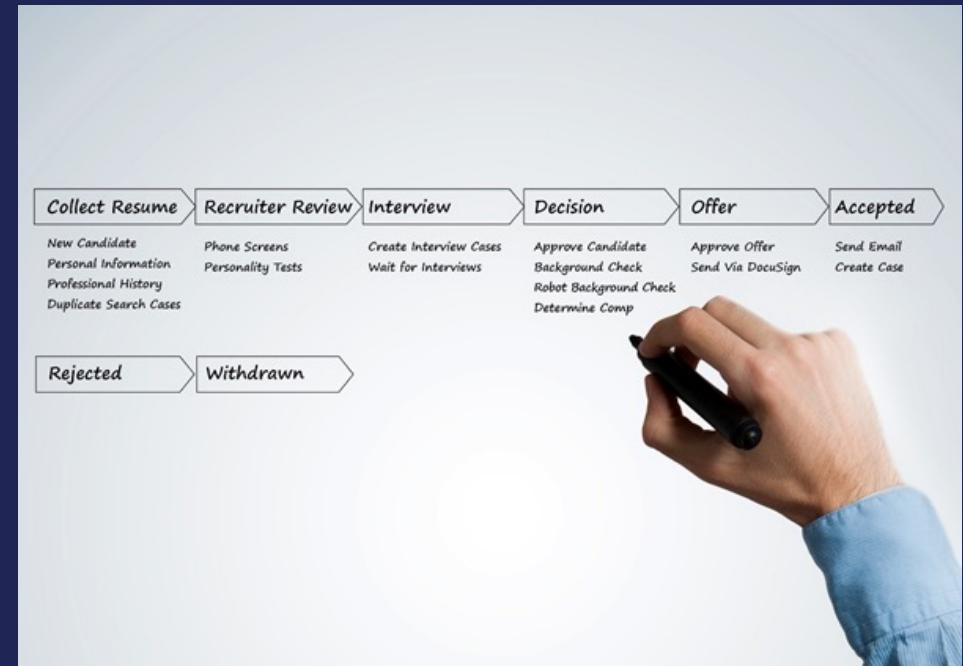
© 2022 Pegasystems Classroom Experience Training Materials



Case Management

Definition

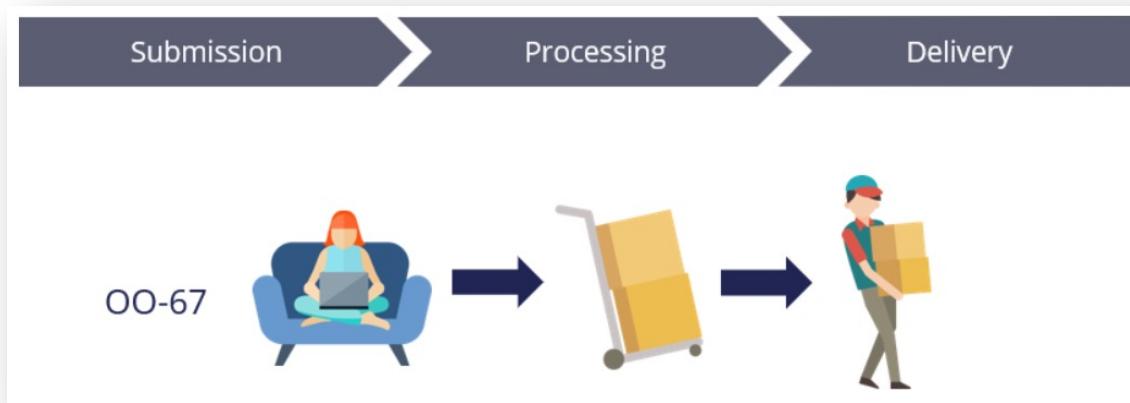
- Design your application to function in the same way that you think about your work
- Define a case life cycle that consists of tasks grouped in a logical and practical way
- A transaction has a life cycle or a process that it flows through
- A Case Life cycle will have:
 - Primary tasks
 - Exceptions and errors
 - Change of ownership
 - Routing
 - Collect and display information
 - Approvals and rejections



Case Management

Description

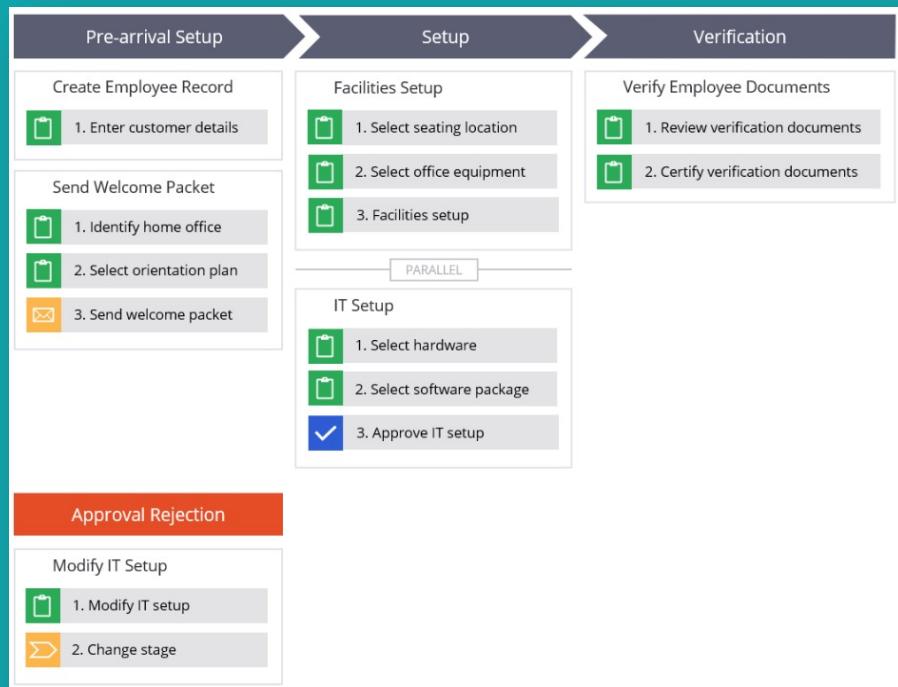
- Pega Platform provides an intuitive editor to define your business process in the way that reflects work that you need to perform
- Users can focus on reaching a specific outcome
- Models the business transaction
- Handles business cases from start to resolution
- Combines human actions with digital automations



Case Type

Definition

- Case types consist of stages, processes, and steps, that the case worker completes to resolve the case
- A visual representation of your business process or transaction
- A template for work that you can reuse for processing multiple instances of the same business case, such as reviewing applications from job candidates
- By creating a case type, you define the path that your work follows, the people who are involved in processing a case, and the data that the case requires.



A Case

Definition

- The case type defines a model of the business transaction and represents the executable code
- A case or case instance denotes the runtime state of the business transaction
 - Unique identifier
 - Status to denote position in lifecycle
- The case state is automatically persisted
 - Provides an audit trail
 - Supports historical analysis through reporting

Dwight Schrute (I-6008) NEW 

1 day from now

Collect Resume
DUE IN 2 DAYS FROM NOW

Personal Information

First Name: Dwight

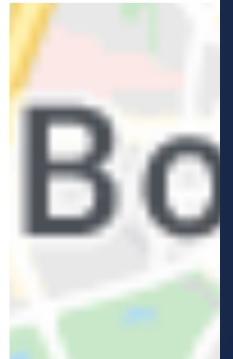
Last Name: Schrute

Phone: 555-123-4567

Email: Dwight.Schrute@beets.com

Skills: Java, Karate, Leadership, Sales

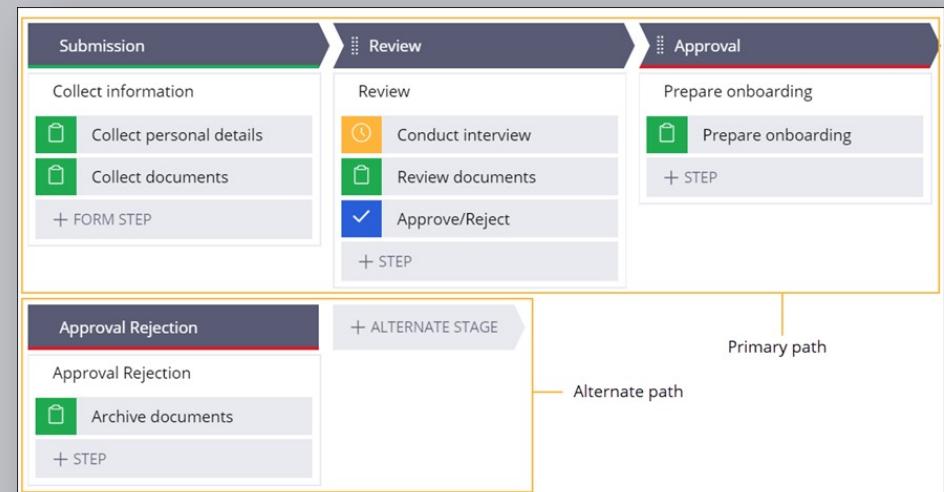
Job Posting

Title: 

Stage

Definition

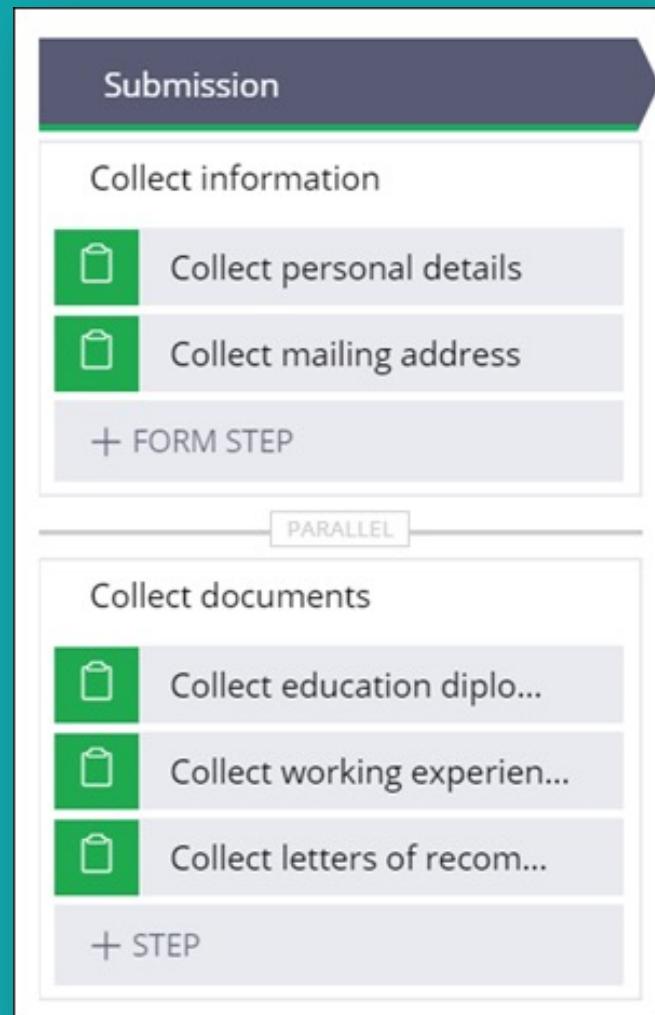
- A stage is the first level of organization for the tasks that are required to complete a work process.
- Stages visualize milestones or significant events in a case life cycle.
- Stages can also indicate a transition of work from one person to another.
- Stages that are necessary to resolve a case by following a default path are **primary stages**
- Cases that alter from the default path may need **alternate stages**
- Default **Create stage**



Process

Definition

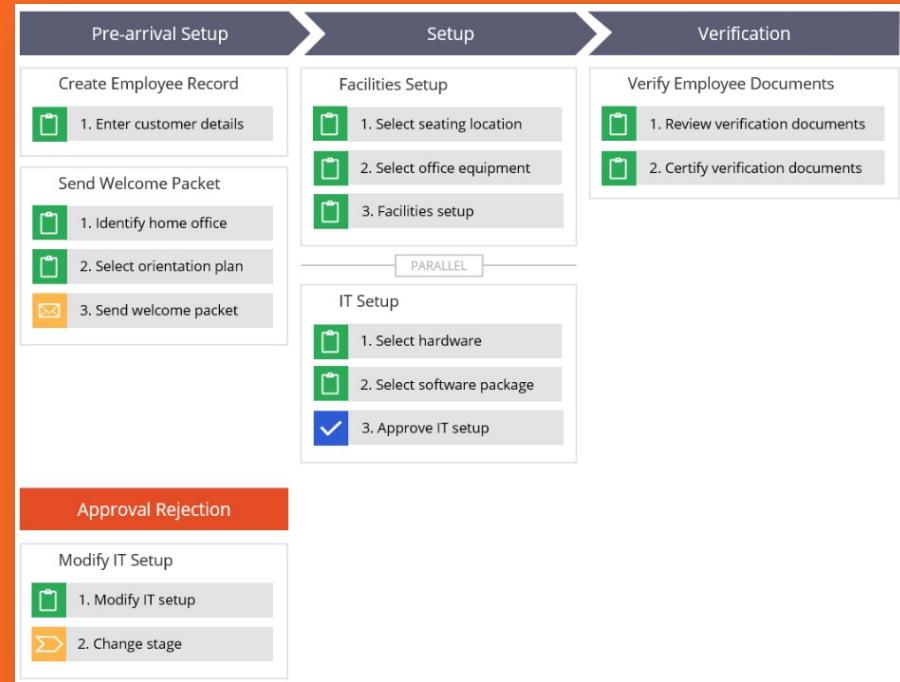
- A process consists of a series of tasks, or steps, and visualizes a set of actions within a stage.
- You can create a sequential process that is a basic set of tasks
- For each stage you can create multiple processes
- A case moves to the next process when all the steps from the previous process are complete
- To speed up case processing, you can also create parallel processes that can involve more case workers simultaneously
- You populate processes by adding steps



Steps

Definition

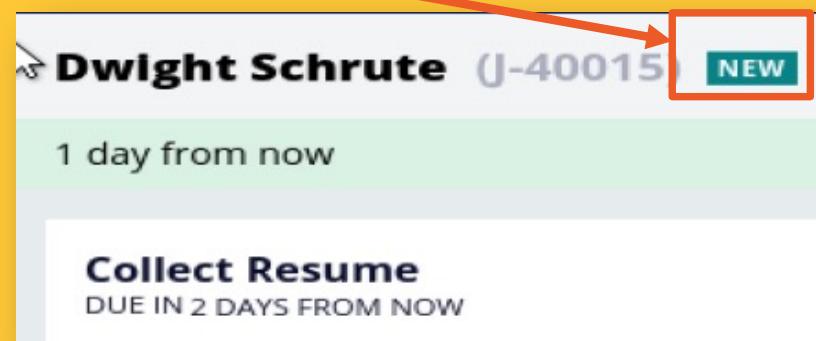
- Steps are the smallest elements of a case life cycle and represent single tasks or assignments
- A step can be a user action or an automation that an application performs.
- The case will include several different types of tasks (steps) to be completed
- Step types include:
 - Collect information, Multi-step form
 - Approval
 - Change stage
 - Send notification
 - Create case
 - Wait
 - Robotic automation



Case Status

Definition

- The case status informs users about the current state of a business process
- Changes to a case status conveys information about the progress of the case towards completion
- Pega Platform provides pre-defined statuses:
 - New, Open, Pending, Resolved
 - Custom status values may be defined



Case ID

Definition

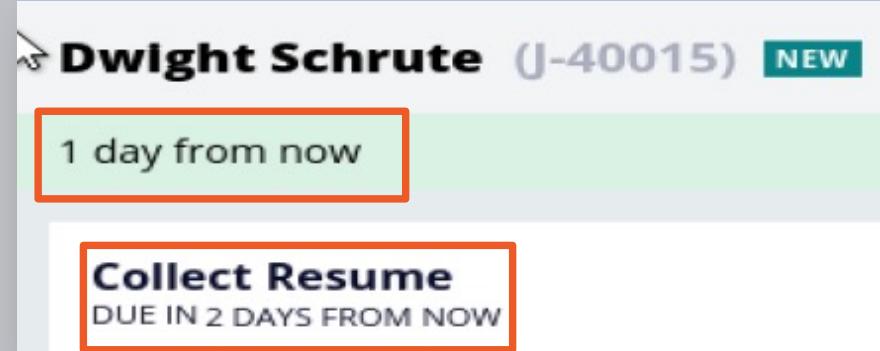
- The case ID is a unique identifier for each case instance
- The case ID helps case workers distinguish between instances of different case types.
- The case ID is automatically generated



Service-level agreement (SLA)

Definition

- The SLA defines intervals or milestones in which work is expected to be completed
 - If not completed within the determined timeframe, the work priority can be increased, and an escalation action can occur
- Many organizations establish service-level agreements to enforce on-time performance
 - These obligations range from informal response-time promises to negotiated contracts



Approvals

Definition

- Case approvals are decision points at which one or more users decide whether to approve or reject a case
- Configure approvals by using the Approve/Reject step
- Defines who is assigned the approval and how the case proceeds if the case is approved or rejected

Sent: 10/10/2019 9:05 AM
From: default@pega.com
To: manager@pega.com
Subject: Equipment upgrade request from Employee is waiting for approval

Hi Manager,

Equipment upgrade request from Employee is waiting for your approval.

Approve Employee's equipment upgrade request?

Reject

✓ Approve

[View Equipment upgrade request](#)

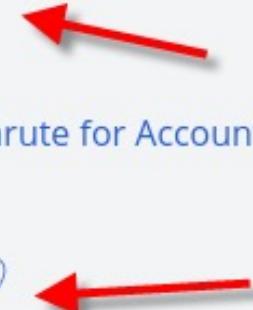
Routing

Definition

- Assigns tasks to operators with the required skills to complete them
- Work tasks may be directed to another operator or work queue explicitly, using business logic or advanced routing rules

Thank you! The next step in this case has been routed appropriately.

Open assignments

- Wait for interviews to complete
(Interview) ⏱
deferred@pega.com
- Interview Dwight Schrute for Account Manager (I-3006)
- New Interview ⏱
Stanley Hudson
- 

Best Practices

Case Management

Follow best practices to ensure that your case types follow rules and policies that you define for your work:

- A case design should be easy to interpret
- Define a case life cycle that orders actions into a sequence
- Use alternate stages to handle errors and exceptions
- Ensure every case is unique by defining conditions that immediately evaluate whether a new case is a duplicate
- Avoid errors and speed up case resolution by assisting users when they provide data for cases
- Track changes that users make in cases
- Create an archival policy for your cases. You can archive inactive cases and remove them from the database to a secondary storage
- Ensure that users interact with relevant case data only, you can configure case access settings

Skill Mastery

Understand:

- Case Management
- A case and case type
- Case ID
- Case status
- Review Service-level agreements
- Review routing and approvals



SKILL
LESSON

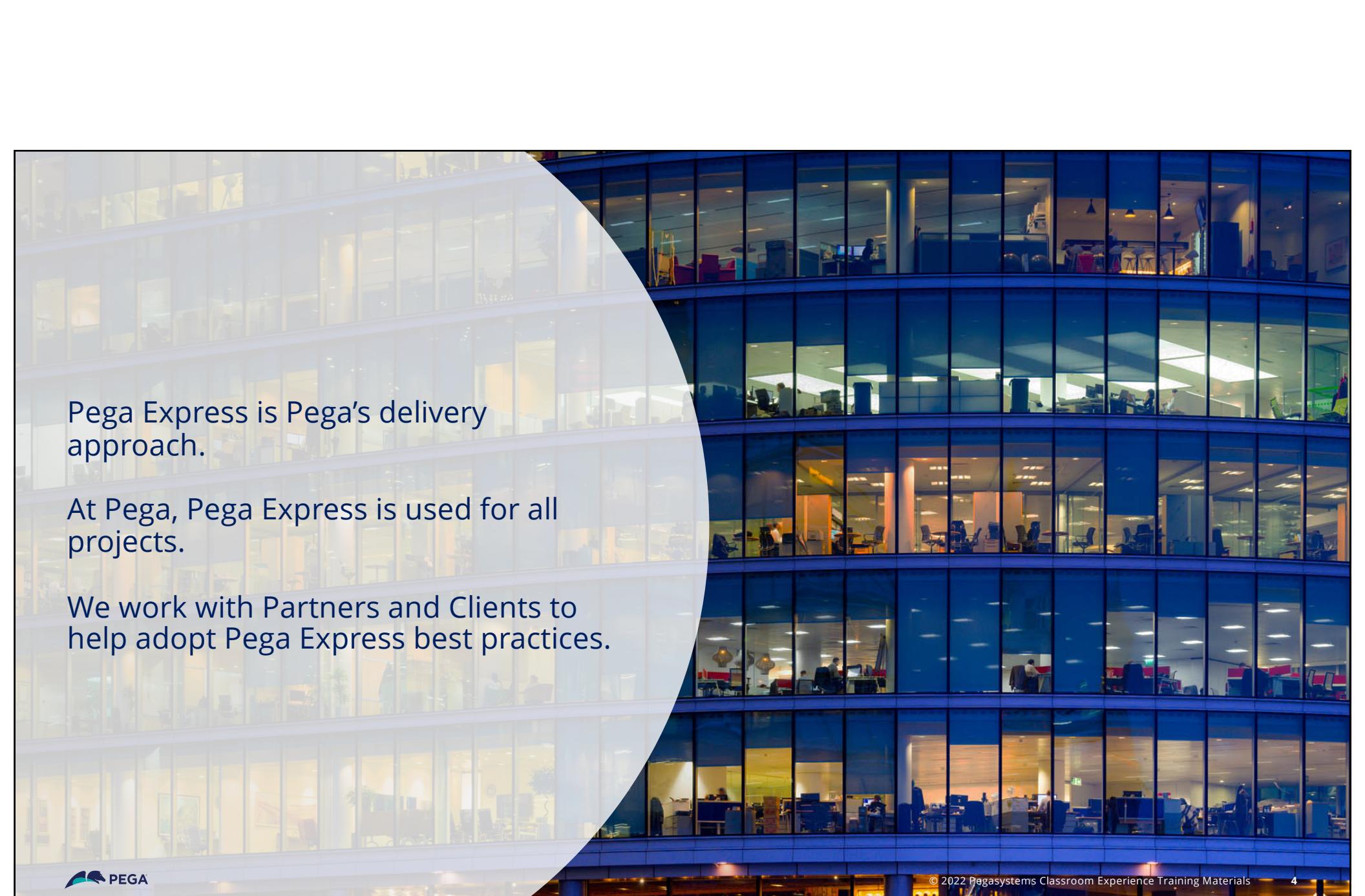
Pega Express Introduction



Overview

Pega Express™ is a light, design-focused delivery methodology that combines Pega's low-code experience, best practices in application development, and Scrum to quickly deliver meaningful outcomes.





Pega Express is Pega's delivery approach.

At Pega, Pega Express is used for all projects.

We work with Partners and Clients to help adopt Pega Express best practices.

Pega Express as a Delivery Approach

Definition

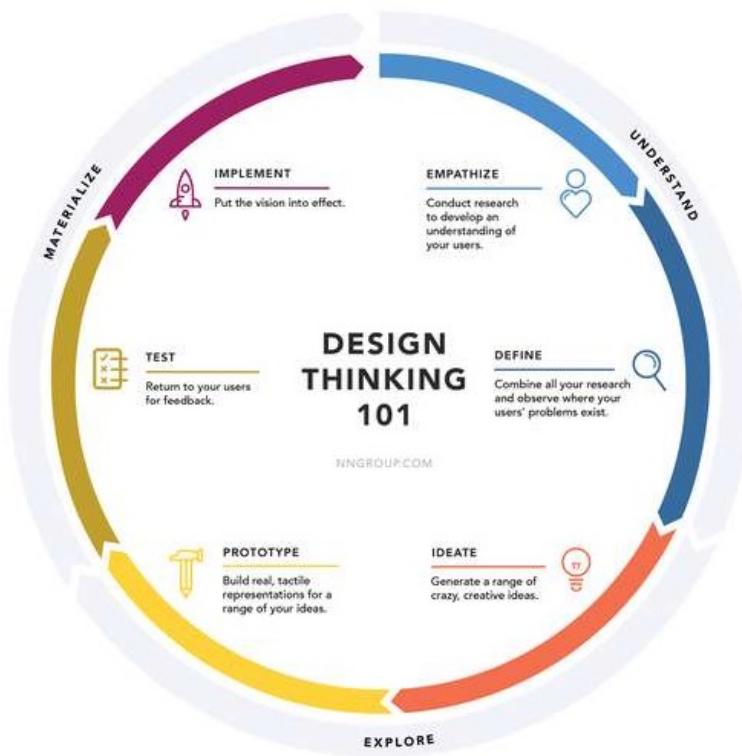
- Pega express Pega's approach to achieving business outcomes rapidly.
- The approach is a fusion of:
 - Pega best practices for example;
 - Directly Capture Objective (DCO) sessions
 - Concept of the microjourney
 - Strong business and IT collaboration
 - Design thinking
 - Pega's Low code capability and features enabled in the platform.
 - Scrum
- Whether big or small, whether CDH, platform or another Pega product, Pega express is our approach for successful deliveries



Pega Express Delivery – Design Thinking

Description

- A Delivery approach that uses design thinking techniques to **break customer journeys** into **smaller**, more manageable pieces called **microjourneys™** that can be designed and deployed in as little as 60-90 days

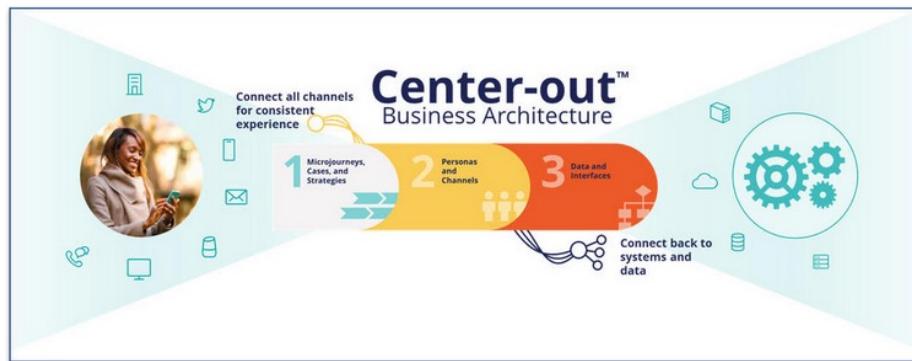


- Microjourneys are business transactions that are delivered to provide immediate value
- Microjourneys are designed to promote and challenge your project team to deliver meaningful client business outcomes quickly

Pega Express Delivery – Center Out

Description

- Pega Express supports the **Center-out approach**
 - Aligns human intelligence and process automation with business logic - beginning at the center of the business
 - Ensures your Pega Platform™ software solution is consistent, seamless, and contextual, providing a great user experience across channels
- A **Center-out business architecture** incorporates the variations within your business
 - variations that extend from your channels to your systems of record
- Pega Platform's layered architecture lets you start small, deploy quick wins, and tackle immediate problems and opportunities
 - You can trust that what you build is scalable as your company grows in scope and size
 - Pega Express reuses microjourneys across the dimensions of customer type, lines of business, or geographies.



Phases of Pega Express – Four Phases

Implementation



Discover

- Use design thinking techniques to clarify the desired outcomes
- Define your Minimum Lovable Product (MLP) and get ready to begin your project



Prepare

- Design a solution and creates a product backlog using Scrum best practices
- Establish governance and enable your team



Adopt

- Readies your application for production and prepares the business for transition to "business-as-usual"
- Analyze application performance while planning your next MLP release

Build

- Iteratively configure and test your application
- Use Scrum project management methods with Pega's low-code platforms and out-of-the-box tools
- Pega Express best practices are built into platforms like App Studio or Next Best Action Designer to define application components and prioritize MLPs quickly

Phases of Pega Express – Four Phases

Implementation

Pega Express: What to do and when

Focus on your outcomes during **Discover**

- Decide on the **microjourney™**(s) to build, using **Design Thinking** techniques
- Use **low code** to capture the three pillars
- Estimate and size your project
- Create a **Day 1 Live Plan**
- Establish a **Coproduction** project team
- Get ready to start your project



Deliver your outcomes during **Adopt**

- Make sure the application is **ready for go live**
- Prepare the business
- Complete your User Adoption Plan
- Release Retrospective
- Handover to BAU
- Analyse Performance
- Plan your **next MLP**

Kick-off your project during **Prepare**

- Validate the overall **project plan**
- Confirm the **application design**
- Use DCO to create 2 sprint's worth of user stories
- Create your **Product Backlog**
- Establish **governance** structures
- Execute your coproduction **enablement Plan**

Build with speed and agility

- Iteratively build & test using **the three pillars**
- Ideation sessions & Lean Usability Testing
- Create User Stories for future sprints using DCO
- Business Testing in Sprint
- Use Pega's **automated testing tools**
- Implement **Pega DevOps**

Best Practices

Best practices and tools on which Pega Express is built

- Pega Express™ serves as a best practice delivery approach for your implementation
- The Pega Platform™ incorporates many elements of Pega Express best practices within the tool to deliver business outcomes faster and more consistently

Business outcomes - value



Design thinking is a human-centered approach to solving problems through creativity and collaboration throughout a project, building empathy to satisfy user needs & realize business outcomes to ensure lovable Microjourneys™



Directly Capture Objectives (DCO) is the Pega Express discipline used to design solutions with clients directly in Pega to foster ongoing collaboration and feedback between IT and business stakeholders



Pega Express is based on **Agile** and **Scrum** to deliver prioritized business outcomes or **Microjourneys™** in increments known as a **Minimum Lovable Product (MLP)** that provide clients speed to value within 90 days

Technical outcomes - quality



Pega's **low-code / no-code studios** allow you to create intuitive applications using the 3 pillars: case types, strategies and automations; integrated with live data; and channels, to provide a lovable experience to your customers



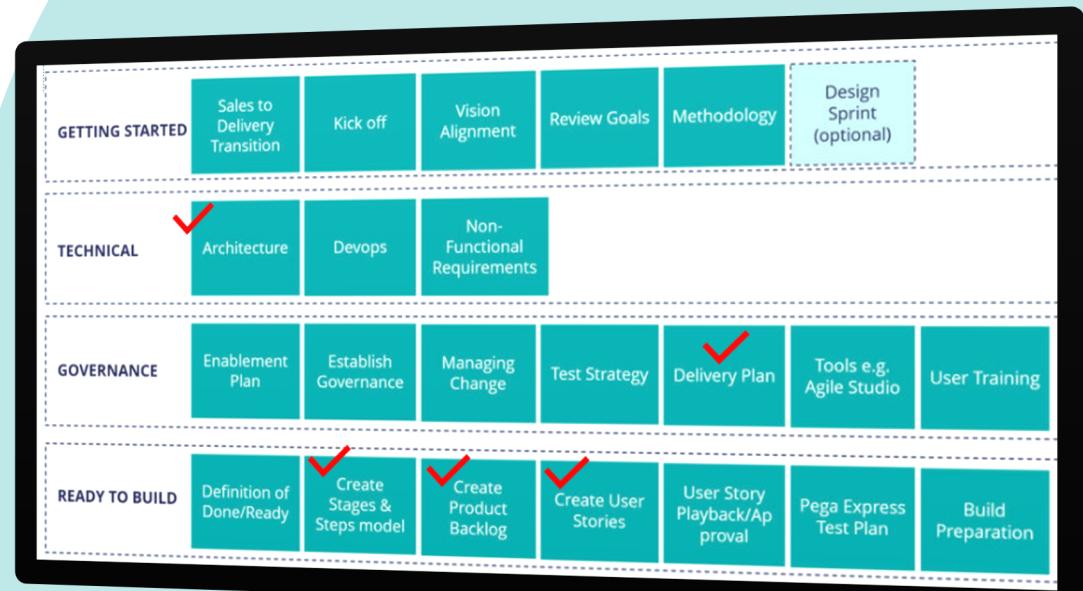
Pega's **DevOps Deployment Manager and automated test suite** enables teams to test early, often, and more efficiently than through manual testing alone so that issues can be identified and fixed as early as possible



Pega's **Application Quality dashboard and Predictive Diagnostic Cloud™** automatically monitor the health of your solution and notify you of potential risks so that you can address them earlier, before they become issues

Overview

- **Architecture and design decisions** in the Prepare phase ready your team for the Build phase.
- You refine the **business and technical scope** into a detailed design for your project.
- You run several **focused workshops** with both business and technical teams.
- Once the **baseline design is built** and your microjourney™ is defined, your team prioritizes **user stories** and creates a **project backlog**.



Case Design

Definition

- The objective of the case design activity is to:
 - Create the foundation case types and the associations between them in preparation for the **Build** phase
 - Validate the functional scope of the microjourney with the product owner
 - Ensure that the case design applies best practices from the start



Backlog

Definition

- The product backlog is an ordered list of project-specific work items, small or large, that are candidates for implementation.
- Using Scrum terminology, these items are referred to as user stories.
- Pega Express™ also uses the term epics to denote groupings of user stories.



User Story

Definition

- The primary purpose of user stories as project artifacts is to help the development team keep track (within the project backlog) of all the features they need to build and implement for the client.
- User stories describe what is required so that features can be designed to satisfy the Product Owner's satisfaction and, ultimately, the end business user. A team creates a user story with the team's collective knowledge and experience.
 - The key features of a user story:
 - Human-centric
 - Easy to understand
 - Clearly reflects the business value the client receives
 - Can be tested

As a...
Call Center Supervisor

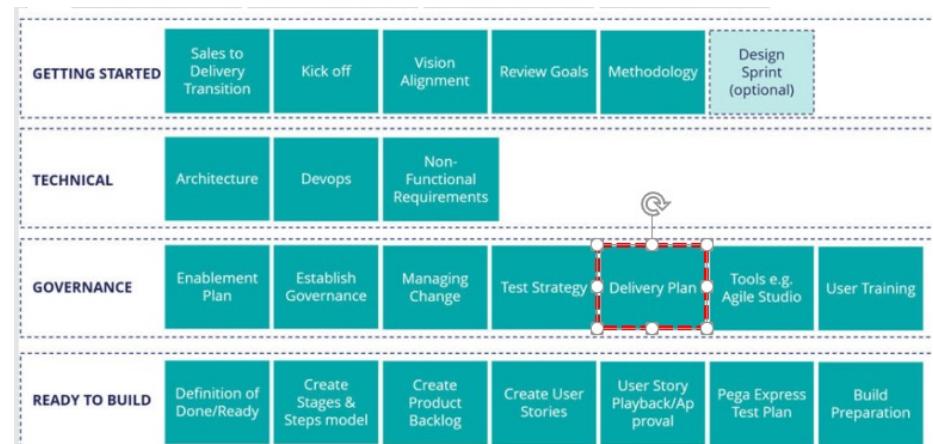
I would like...
to see in real-time how many customers have registered for a promotion

so that...
I can make adjustments as needed to promotion enrollment scripts and awareness.

Validate the Plan

Definition/Description

- By the end of the **Prepare** phase, you know much about the microjourney™ and the first Minimum Lovable Product (MLP) release. Once your microjourneys and MLP are defined, you review the project plan.
- The project plan elements include:
 - **Project team** – Is the team adequately resource based on your understanding of the microjourney?
 - **Sprint plan** – Is your work schedule and timing for your MLP achievable given your team members' capacity?
 - **Key risks** – Are there issues with the technical architecture that need to be addressed (such as integrations that might impact successful delivery)?
 - **RAID log** – Is the RAID log (Risks, Actions, Issues, and Decisions) up to date with new information discovered during the Prepare phase?
 - **Assumptions** – Are the assumptions made during the **Discover** phase still valid; have any new assumptions been raised?



Skill Mastery

- Describe the four phases of a Pega Express delivery
- Demonstrate awareness of Pega Express terminology
- Articulate the benefits of a Pega Express delivery
- Describe key architecture and design concepts



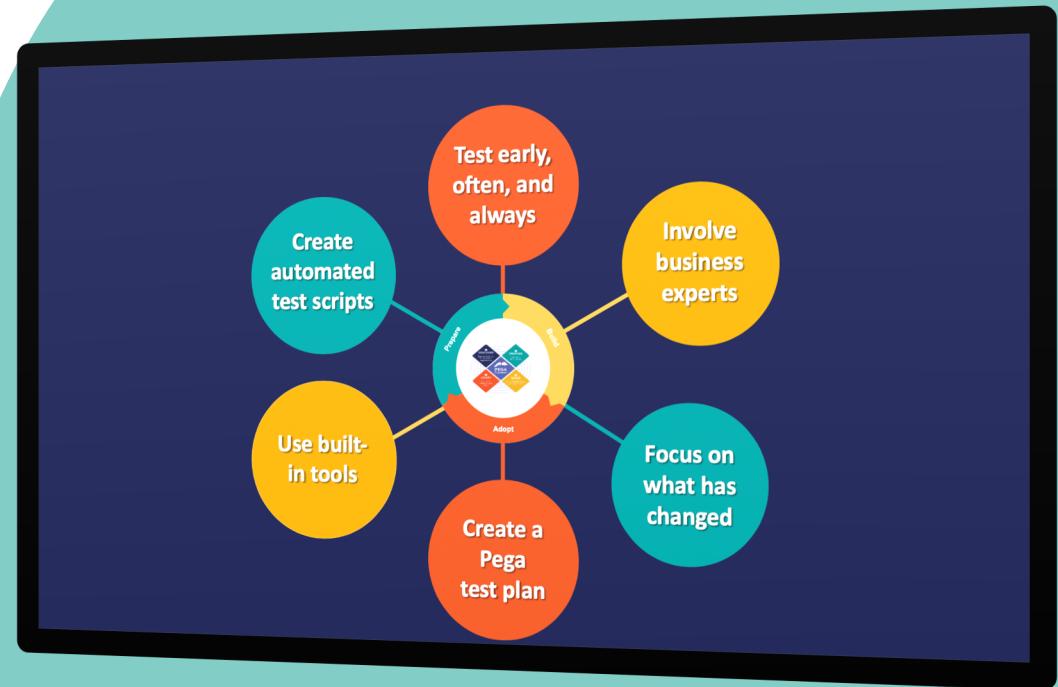
SKILL
LESSON

Testing Best Practices



Overview

Pega Platform™ provides tools and processes to streamline your team's application testing. Improved testing ensures that the solution you create, build, and deliver is one that your client loves — a Minimum Lovable Product (MLP)





To avoid configuration errors
developers, test their
applications.

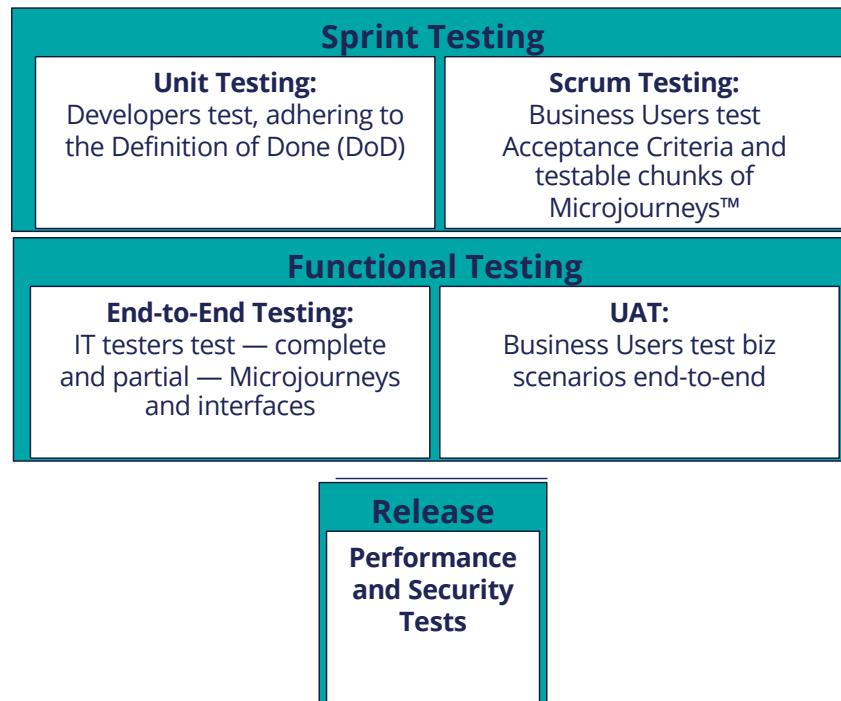
When an error occurs, end users might need to reassign work, or a case might require repair by an administrator. If the case is routed to the wrong operator, it delays the customer during case reassignment



Testing

Definition

- The main purpose of testing is to identify configuration errors in a rule before the errors are compounded in an application when the result of the rule depends on another rule.
- The most basic form of application testing is **unit testing** individual rules.



Business experts should be part of testing

Description



PegaUnit: Access and execute from the landing page

Navigation

3

Run the tests

2

Select the tests

PegaUnit landing page

1

Filter the tests

5

Analyze results

4

View results and click on "Failed"

The screenshot shows the PegaUnit landing page with the following interface elements:

- Header:** Application: Unit testing, Refresh, Help, Close.
- Sub-headers:** Test cases, Test suites, Reports.
- Applications included:** U-Bank (current), Customer Service 8, CRM Base Application.
- Message:** All tests on this page are selected. Select all 432 tests in the application.
- Buttons:** Run selected, Create test suite, Export to Excel.
- Table Headers:** Test case name, Rule type, Rule name, Class, Disabled, Last run, Result, Run history.
- Table Data:** Two rows of test cases, each with a checkbox, rule type (Data Transform), rule name (CreditApplicationDetails), class (UPlus-UPlus_CS-Work-CreditCardApplication), status (No), last run (27 d ago), result (Passed), and a "View" button.
- Test result for TC_AccountOwner:** Date: 19-Sep-2018 04:42:00 AM EDT.
- Test failed:** Icon of a computer with a sad face, message: "Test failed!", Run by: Prasanna Settipalli.
- Tested rule:** Account Owner Party (Data Page D_AccountOwner_Party of class PegaCA-Party).
- Ruleset:** UPlus_CS : 01-01-02.
- Parameters:** AccountNumber="123450000".
- Rule run time/ Expected time:** 2.402/0.043 sec.
- Unexpected results:** Assertion type, Expected Runtime.
- Table:** Property (Expected Runtime), Comparator (is less than or equal to), Expected Value (0.043), Actual Value (2.402).

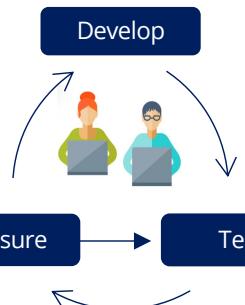
Scenario testing

Navigation

The screenshot shows a Pega application interface with several features highlighted by numbered callouts:

- 1** Records tests by simply running
- 2** Provides support for all controls
- 3** Uses easy to read test steps
- 4** Offers test case types and end-user portal
- 5** Supports dynamic test data

The interface includes a navigation bar with "U+BANK" and "+ New", a search bar, and tabs for "Case details" and "Audit". A modal window titled "Test case for Log Feedback" is open, showing a recording screen with a "Stop and save test case" button and a "Test steps" section containing a single step: "Initiate Complaint". Below the modal, a "Complaint or Compliment (WC-47)" case is shown with options for "Complaint", "Compliment", and "Suggestion". Case details include Case ID WC-47, Urgency 20, Goal 23 hours from now, Deadline Mar 16, 2019, Status Open, and Last update Mar 14, 2019.



Built in tools - Application Quality dashboard

Implementation

The screenshot displays the Pega Application Quality dashboard with the following key metrics:

- Guardrails**:
 - Weighted score: **71** (Orange)
 - Warnings: **656** (Orange)
 - Severe: **6** (Red)
- Unit testing**:
 - Test pass rate: **99%** (Green)
 - 420 of 422 test cases** (Green)
- Test coverage**:
 - Rules covered: **58%** (Red)
 - 376 of 651 executable rules** (Red)

Below these metrics, a table shows application details:

Credit card application	86
Log Complaint	64
Log Feedback	55

At the bottom right, there is a teal decorative graphic.

Built in tools - Leveraging quality metrics in CICD pipelines

Implementation

The screenshot shows the Deployment Manager interface with a pipeline named "UPlus_CS". The pipeline consists of four stages: Development, Quality Assurance, Staging, and Production. Each stage contains various tasks such as "Generate artifact", "Deploy", and "Run Pega unit tests". A callout bubble on the right side of the Quality Assurance stage contains the text: "Measure guardrail compliance, run PegaUnit and Scenario tests, and validate test coverage from Deployment Manager".

Leverage REST APIs to access quality metrics from DevOps pipelines built in 3rd party tools

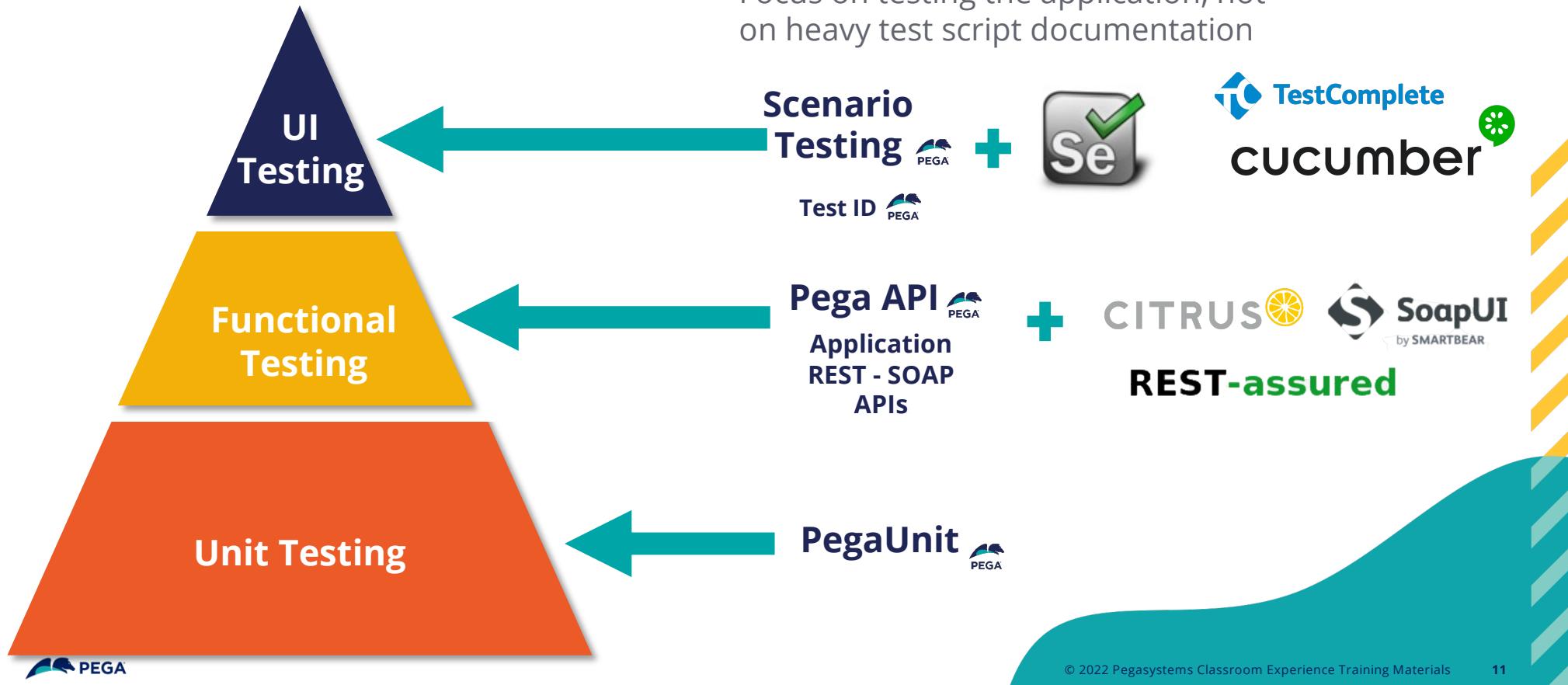
The screenshot shows the DevOps REST API documentation under the "applications" section. It lists several endpoints:

Method	Endpoint	Description
GET	/applications/{ID}/quality_metrics	Get application(s) quality metric details
GET	/applications/{ID}/guardrail_metrics	Get application(s) guardrail metric details
GET	/applications/{ID}/start_application_coverage	Start application coverage
GET	/applications/{ID}/stop_application_coverage	Stop application coverage
GET	/applications/{ID}/test_metrics	Get pegaunit statistics
POST	/applications/test_metrics_for_rules	Get test metrics for given rules

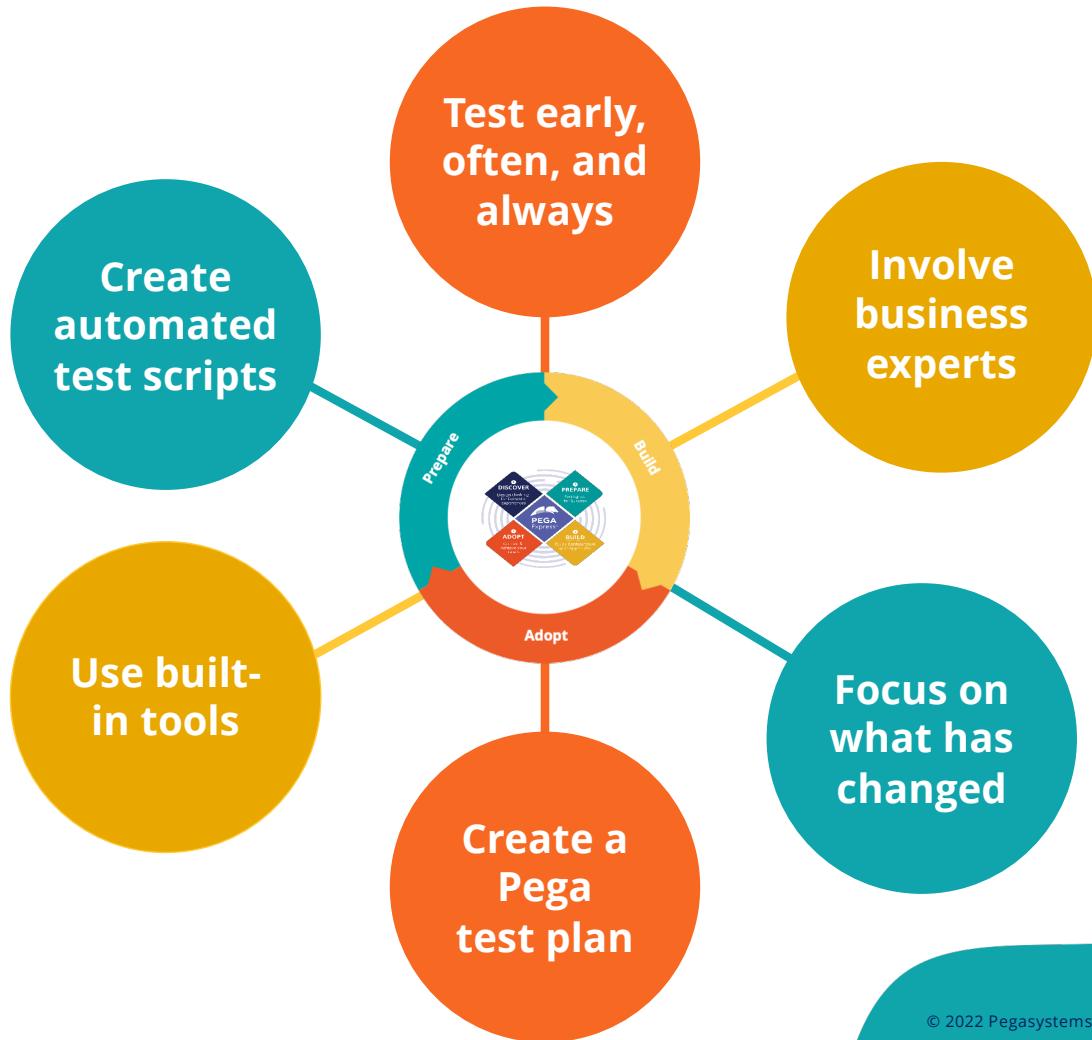
Building your automation test suite



Implementation



Best Practices Testing and Accelerators





SKILL
LESSON

Pega Platform Studios

A business view of work





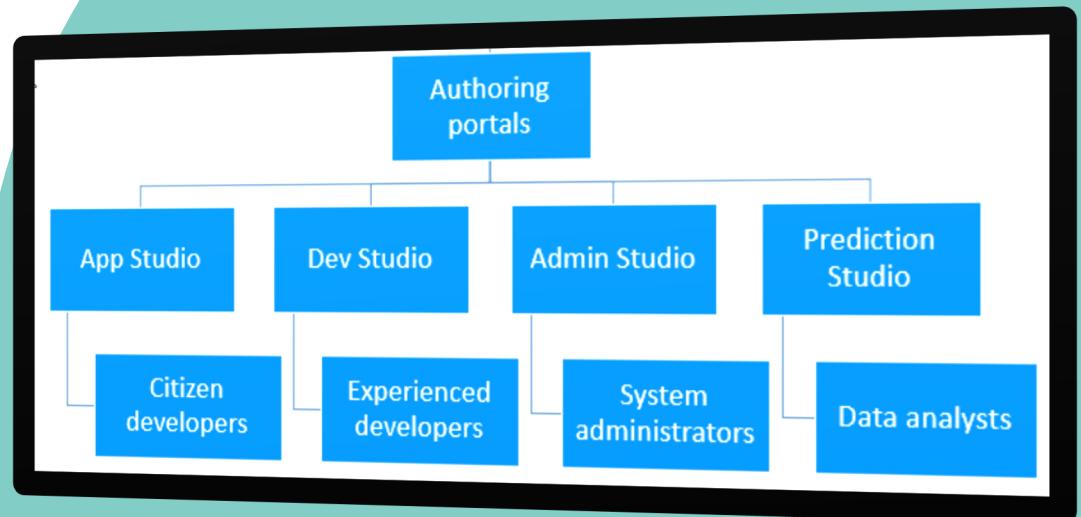
Overview

Pega provides four role-based authoring workspaces known as Studios.

These studios allow for the manipulation of the visual model using low-code development tools that create code.

This module covers:

- Pega Studios
- Creating a request in Pega



Use Case

- Use the Pega Platform studios to collaborate with stakeholders in a common visual language that focuses on explaining the business logic rather than the code.
- Low-code application development platforms bridge the gap between business stakeholders and skilled developers, creating a common visual language to collaborate more effectively.

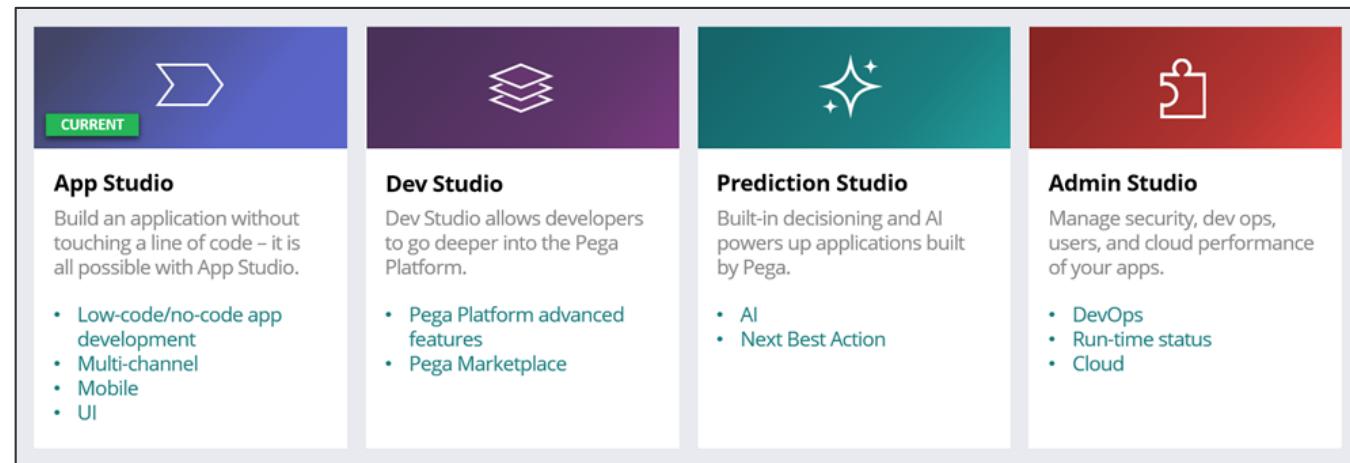




Workspaces

Definition

- A workspace is an environment that provides specific tools and features.
- Different workspaces are used to manage the application and to focus on the tasks that align with the expertise of team members.
- Pega Platform™ provides four role-based authoring workspaces, known as studios:
 - App Studio
 - Dev Studio
 - Prediction Studio
 - Admin Studio



Studios

Description

- **App Studio**

- For prototyping and collaboration
- Used to build an application without coding
- Use to get applications operating quickly

- **Dev Studio**

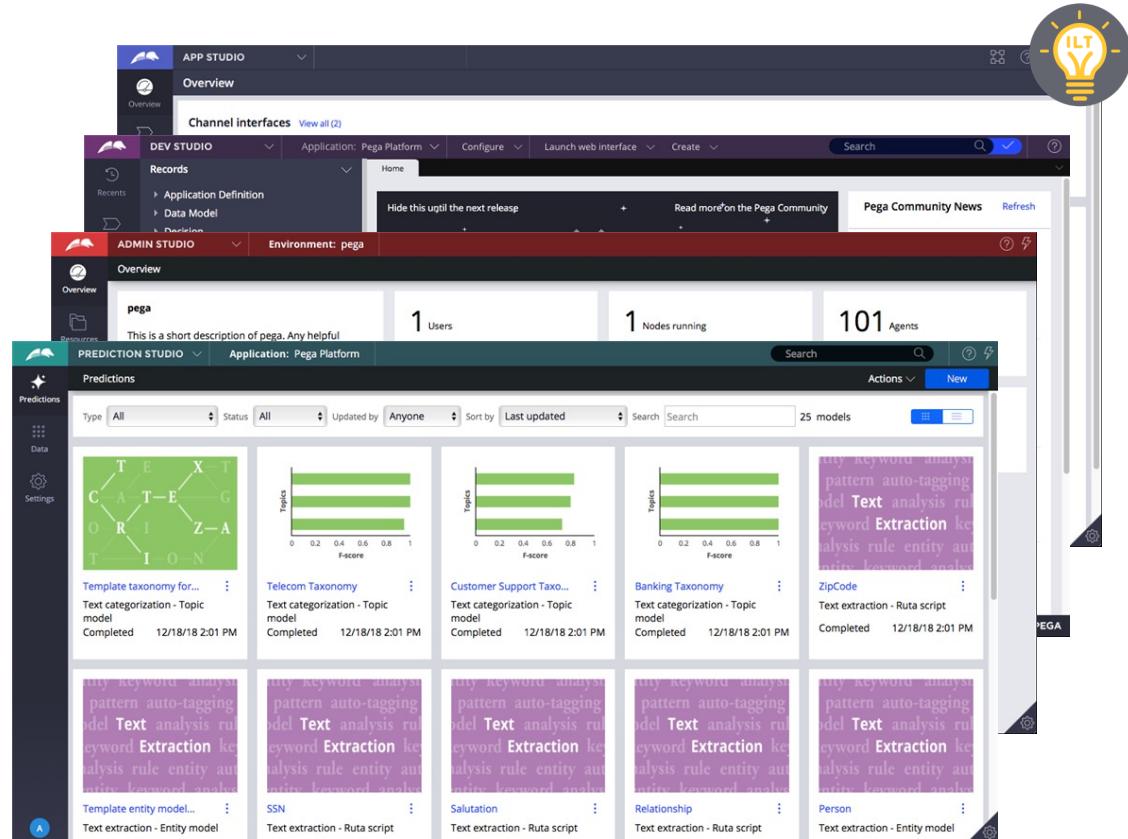
- Advanced development and configuration
- Allow developers to configure rules

- **Admin Studio**

- For System management and monitoring
- Manages security, dev operations, users, and cloud performance

- **Prediction Studio**

- Used by Data Scientists
- Has built-in decisioning and AI capabilities



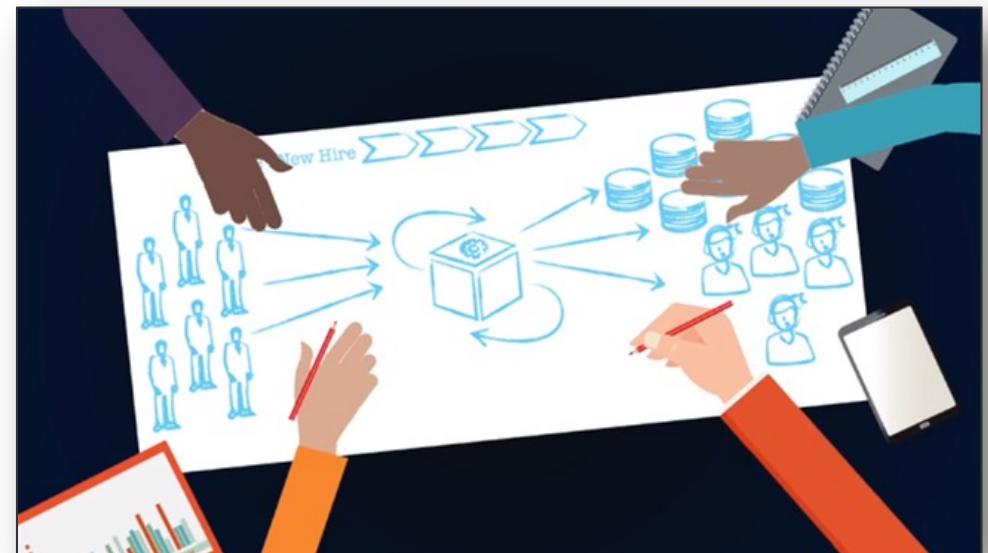
Use the Switch Studio menu in the upper-left corner of the Header to switch between Studios.



App Studio

Description

- Provides core features for application development, such as case design, data management, and user experience.
- Designed for low-code users.
- Typical users include application developers, front-end developers, data engineers, and business analysts.
- Helps visualize the key factors of the desired business process.
- Draft relationships between stages of the process, participating personas, communication channels, and data required for process resolution.





Dev Studio

Description

- Supports advanced rule configuration in applications.
 - Access rule forms directly to address complex or less-common configuration requirements.
 - Provides features for configuring security permissions and access control, managing rules to promote reuse, and addressing the performance limitations of an application.
 - Developers can access all the layers of an application and can extend the rule scope from a single case type to an application, division, or even the entire organization to create a library of standardized, reusable rules.

The screenshot illustrates a two-step process for address entry:

- Step 1:** The user fills out the "Billing address" form, which includes fields for Street, City, State/Province, and Postal code. Below the form is a "Shipping address" section with identical fields.
- Step 2:** The user submits the "Billing address" form. The "Shipping address" fields are automatically populated with the same values from the "Billing address" form.

A central modal window titled "pyWorkflow" provides insight into the data transformation logic:

- Cell Properties:** Shows a "Billing address" row with a "Billing address" column.
- Action Set:** An "Action set 1" is defined with the following details:
 - Applicability:** Both
 - Action:** Run data transform
 - Page:** Auto
 - Data Transform:** fetchShippingAddress
 - Conditions:** When Add a condition



Studio Area

Navigation

Each studio consists of three areas:

- **Header**

- The header contains various menus or tools, depending on which studio is active

- **Navigation panel**

- The navigation pane allows you to access various parts of an application through explorers. For example, the **Case Types Explorer** or **Data Explorer**.

- **Workspace**

- The workspace is used to configure the application behavior.

The screenshot shows the Pega App Studio interface. The left sidebar includes icons for Overview, Case types, Data, Channels, Users, and Settings. The main workspace displays the following sections:

- MyTown 311**: Application details (Version: Pega platform 8.5.0), Application documents (About, Export), Application profile (Manage).
- Personas**: Manager (Generated by New Application API) and User (Generated by New Application API).
- Channels**: Manage (3) - User Portal (Default employee-facing portal for C...), User Mobile App, Pega API (APIs are a set of REST services expos...).
- Case types**: Manage (1) - Service Request (Create, Review, Resolution).
- Data objects**: Manage (No data objects). View data model.

A sidebar on the right shows "Application Layers" with a 3D grid diagram, and a footer lists "MyTown 311 (01.01.01)", "Cosmos (02.01)", "Pega Platform (8)", and "Load More".



End user portals

Definition

- Portal(s) are the workspaces the end-user community use to complete work tasks.

- **Case Worker**

- used by end users who create and process cases

- **Case Manager**

- used by managers of case workers

The image displays two screenshots of the Pega Case Manager application interface. The top screenshot shows the 'Summary for MyTown 311' page, which includes a welcome message for the 'Pega Case Manager', instructions to easily create and manage cases, and links to 'Take a tour' and 'Open Pega Community'. It also features a 'Cases by status' section and a 'Case stages for' search bar. The bottom screenshot shows the 'My Work' page, which lists cases with columns for Name, Case, and Category. It includes a search bar at the top and a 'Create an ad-hoc case' button. The sidebar on both pages includes links for Dashboard, My Work, Pulse, Spaces, Documents, My Teams, Reports, Tags, and Following.



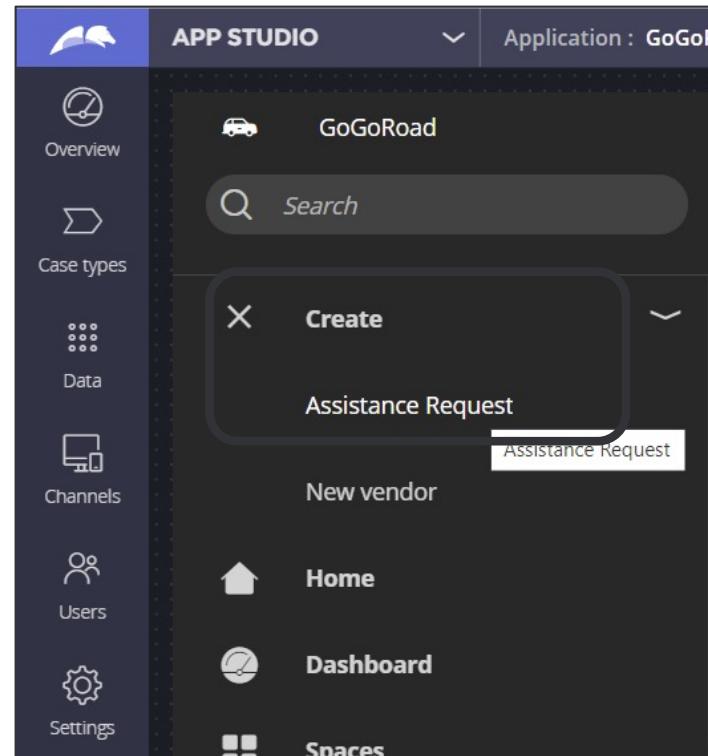
Creating A Request

Navigation

A user creates a request from the Case Worker or Case Manager Portal in Production. The Dev Studio and App Studio can be used to create requests as part of the development process or during testing.

App Studio > Preview Application

Create > New request





Schematic

Design Layer

App Studio
Dev Studio
Prediction Studio
Admin Studio

Rules Layer

Portal

Java HTML CSS XML Javascript JSP

11001010101110001



Skill Mastery

- Understand a workspace.
- Understand how to navigate between Studios.
- Understand usage of the four studios and the associated roles on a Pega project.
- Determine when to configure in App Studio vs Dev Studio.

Case Life Cycle



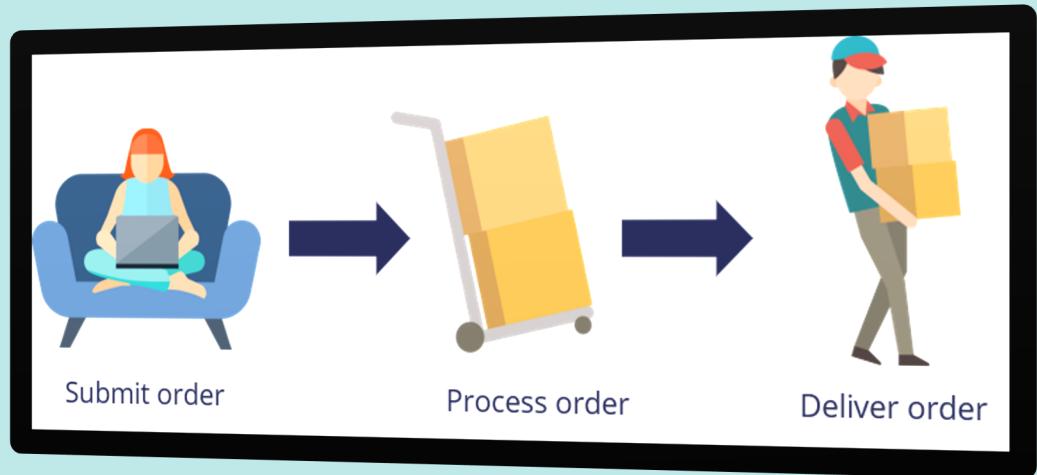
SKILL
LESSON

Case Lifecycle Design



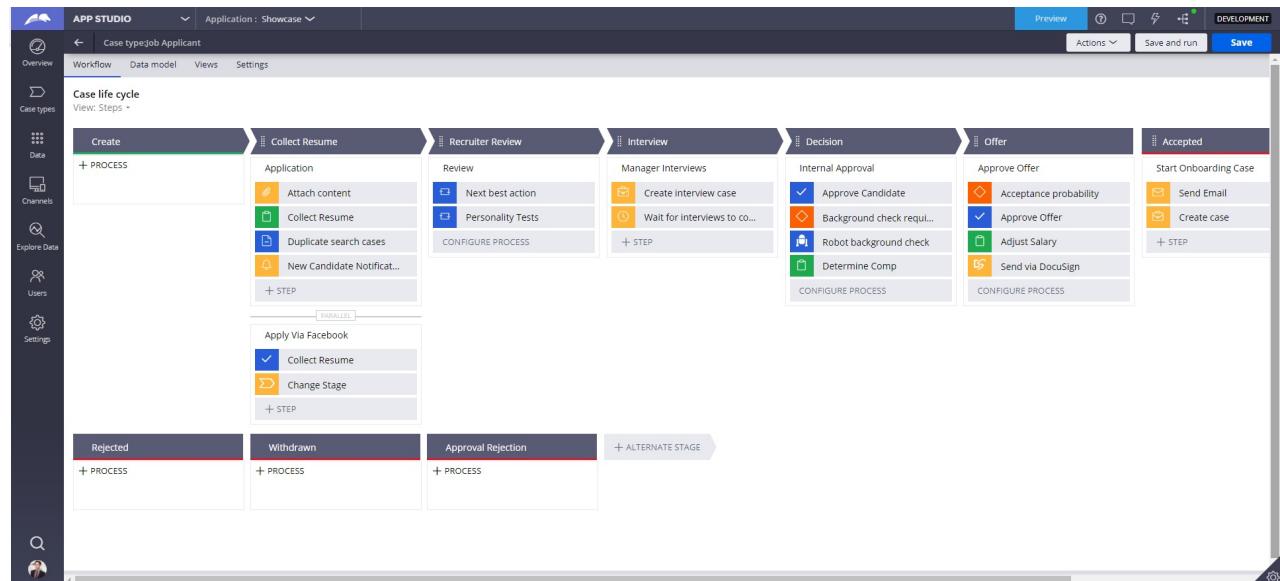
Overview

The Pega Platform uses the case lifecycle design modeling technique to model automated work in the way that business users think about and describe work.



Use Case

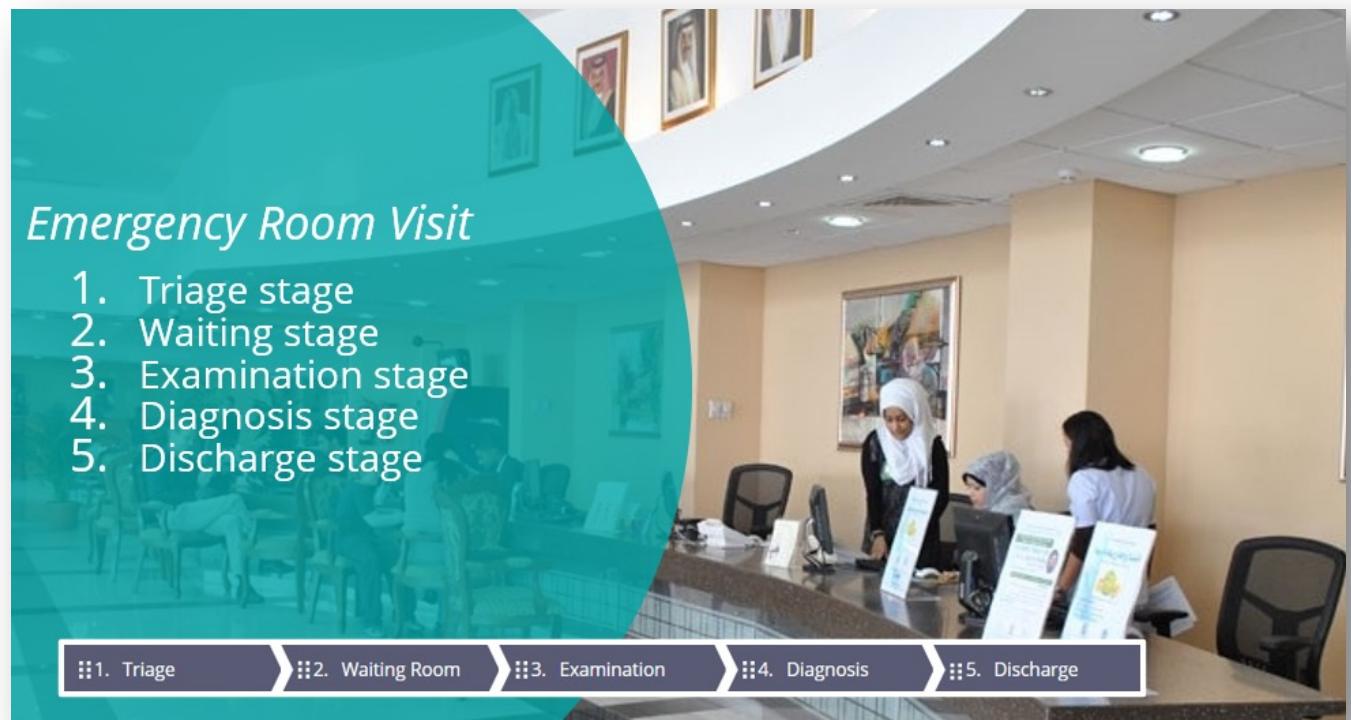
- A case is needed when:
- The work should be assigned, processed, and completed.
- The work should be tracked by a unique ID.
- The work has a repeatable set of tasks.
- The set of data processed will eventually need a status of complete, done or finished.



Case life cycle design

Definition

- Case life cycle design is a visual representation of a business transaction that groups related tasks into stages that can be monitored till completion.
- Case type is an abstract model of a business transaction that is repeatable and leads to a specific outcome.
- Case is a specific instance of a business transaction.



Case life cycle design

Description

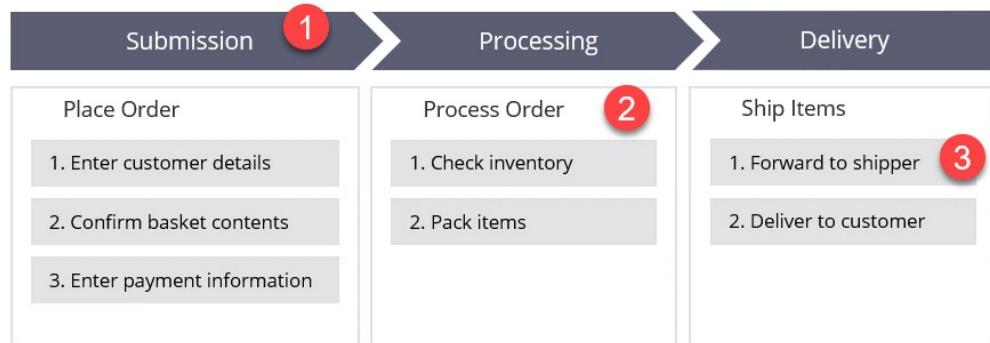
- The case life cycle design is used to model repeatable business transactions.
- A Pega application uses the case life cycle design to create a case type to help automate work to achieve specific business outcomes.
- Pega thinks applications should function the same way the user thinks about and describes work.
- Define the case life cycle to help visualize the work that must be completed as part of the desired business transaction.



Model -case life cycle design

Implementation

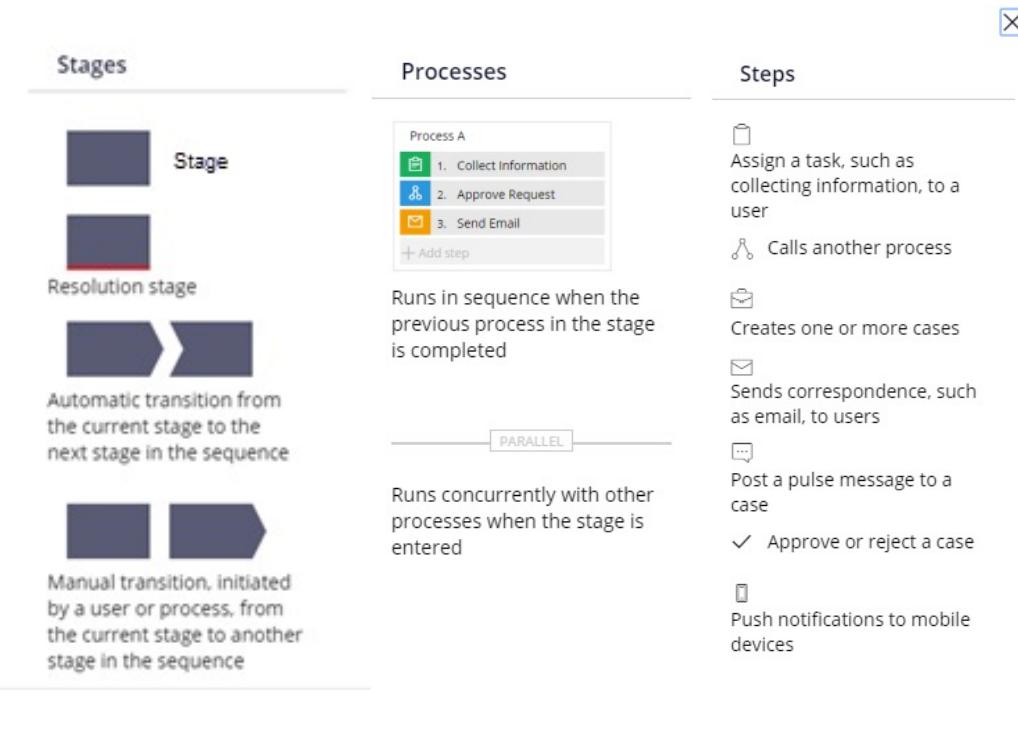
- Begin designing a case life cycle by organizing work into stages.
 - Stages represent the case transfer from one caseworker to another or a significant change in the case status.
- Next, add processes which contain a series of tasks, or steps, that users complete as they work on the case.
 - Each stage can contain one or more processes.
- Finally, add steps which is either a user action or automated action within a process performed by the application.
 - Steps performed by the system are referred to as automation steps.



Case designer legend

Definition

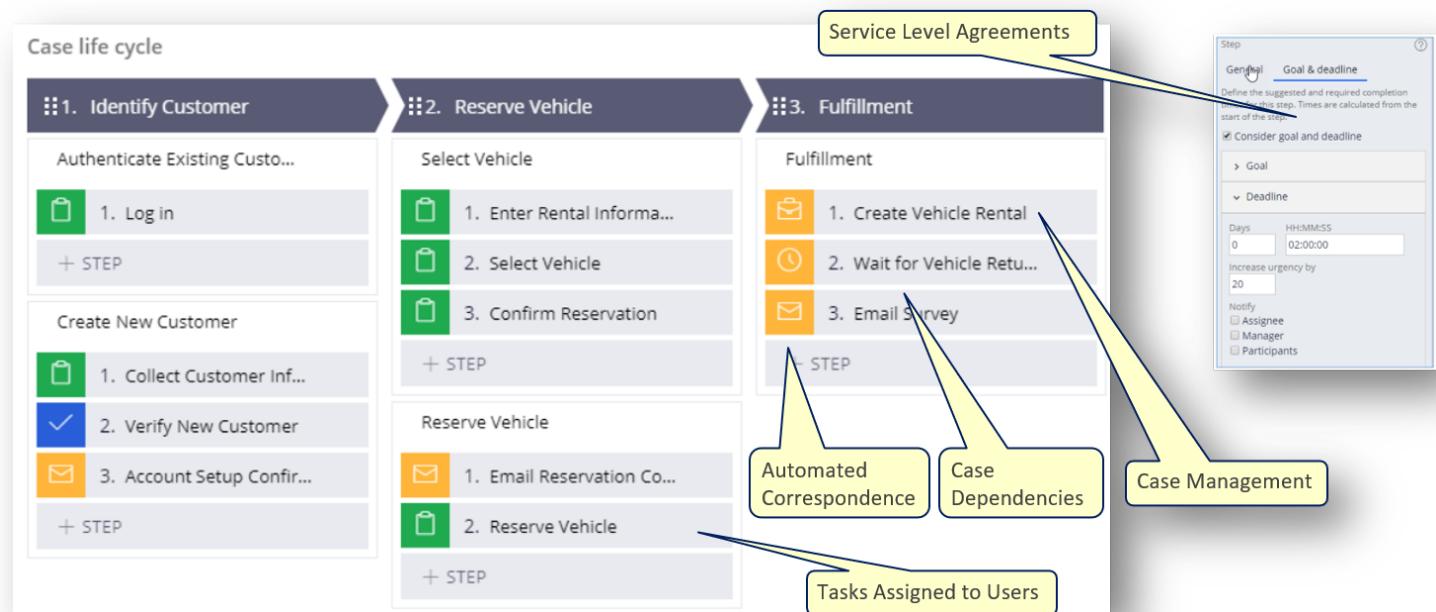
- Case Designer is an informal name for the environment used to configure a case type.
- Using the Case Designer allows quick updates to a case type without having to manage the underlying rules that support it.
- The case designer is used to add stages, processes and steps to a case life cycle design.



Case designer configurations

Description

- Selecting a stage, process or step in the case designer will open the right pane for further configuration.
- The right pane can be used to configure the behavior of a stage, a process or a step.
- The right pane can also be used to configure SLAs, automated correspondence, case dependencies and more



Case designer

Navigation App Studio

Open Existing

1. Select the Case types explorer
2. Click on the existing case type name to review or configure a case type design

The screenshot shows two instances of the App Studio interface. The left instance displays the 'Case types' section with a search bar and a list of case types: 'Evaluate and Sell a Vehicle', 'Interview', and 'RV Calculation'. The right instance shows a detailed view of the 'Assistance request' case type. It includes a navigation bar with 'Data model', 'Workflow' (selected), 'Views', and 'Settings'. Below this is a 'Case life cycle' diagram with four stages: 'Submission', 'Validation', 'Service', and 'Resolution', each with its own set of steps.

Create New

1. Select the Case types explorer
2. Click *+Add a case type* or *New* to create a new case type

This screenshot shows the 'Case types' section of App Studio for the application 'CaseLocking'. It features a search bar and a table listing two case types: 'Benefits Enrollment' and 'Candidate'. A red box highlights the 'New' button in the top right corner of the interface.

Use Case

Create stages as the first level of organizing work in a case type. It contains the tasks, or steps, that users perform before they can move a case to the next phase in its life cycle.

Stages of an “Emergency Room Visit” case type

Primary stages

1. Triage
2. Waiting Room
3. Examination
4. Diagnosis
5. Discharge

Alternate stages

- Isolation
- Cancellation
- Admission



Isolation

Cancellation

Admission



Stages

Definition

- A stage defines the order of related steps and processes in a case.
- The case lifecycle design model allows business users to begin by organizing work into stages.
- A **primary stage** (happy path) is a high-level phase in the lifecycle of a case that leads to a desired outcome.



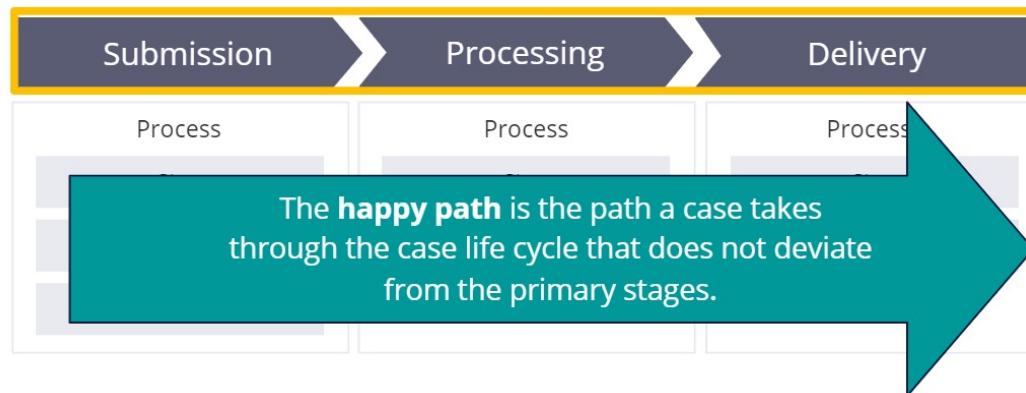
- An **alternate stage** represents “alternate” course events, an exception to primary processing, or an interruption to the “normal” course of events.



Primary stages

Description

- Primary stages visualize the main phases of a business process.
- Cases typically follow primary stages in sequential order.
- Sequential transition from stage to stage can be set as automatic or manual transition.
- Stages represent the transfer of a case from one authority to another or a significant change in the status of the case.



Alternate stages

Description

- Cases enter alternate stages when an error or an exception occurs.
- Alternate stages are not sequential, these stages can only be sequenced manually.
- Adding an Approve/Reject step will automatically create an alternate stage named *Approval Rejection*
- Do not delete or rename this stage, because users will not be able to reject cases.

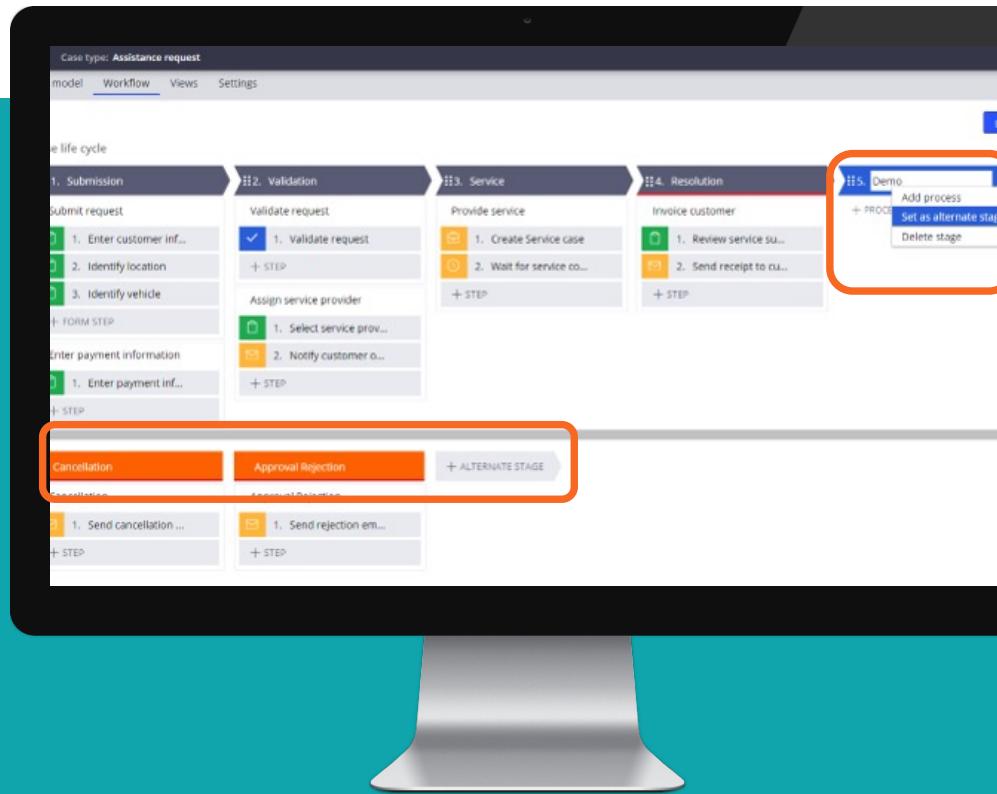


Creating alternate stages

Navigation

From App or Dev Studio:

1. Click the Case types explorer
 2. Click the case name to open
 3. Click Alternate stage
- Or convert a primary stage to an alternate stage
4. On the menu next to the stage name, click Set as alternate stage



Defining resolution stages

Definition and description

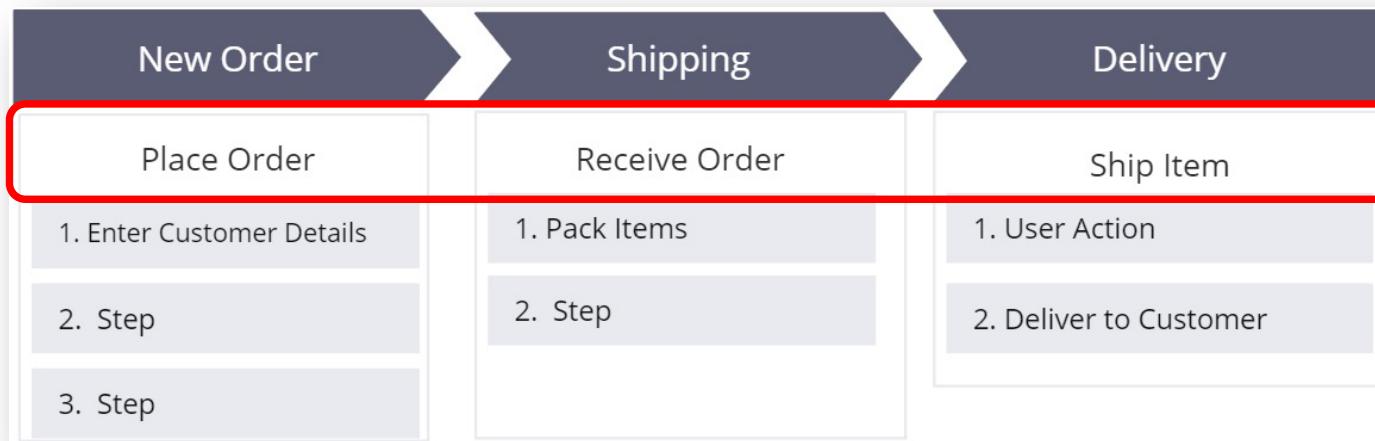
- Resolution is the act of closing a case to control the final status and processing that occurs when a case reaches the end of its life cycle.
- Resolution is visually annotated with a red line under the stage.
- Any stage can be marked as a resolution stage; primary or alternate.
- Each case type has a minimum of one resolution stage per case design.



Processes

Definition

- A series of tasks, or steps, within the stage.
- A collection of tasks to be completed within the stage.
- A process groups related steps
- One or more processes can exist for a stage.
- Multiple processes can execute sequentially or in parallel.

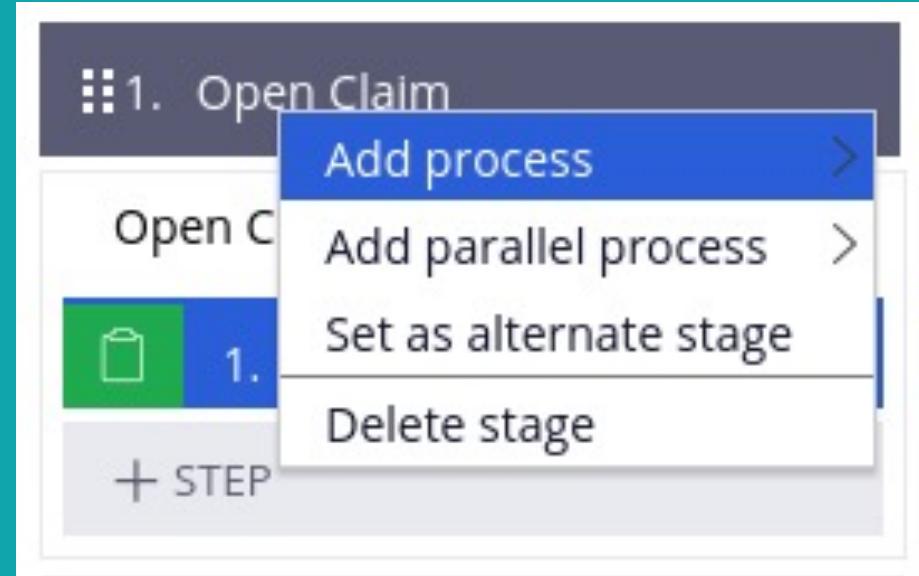


Creating processes

Navigation

From App or Dev studio:

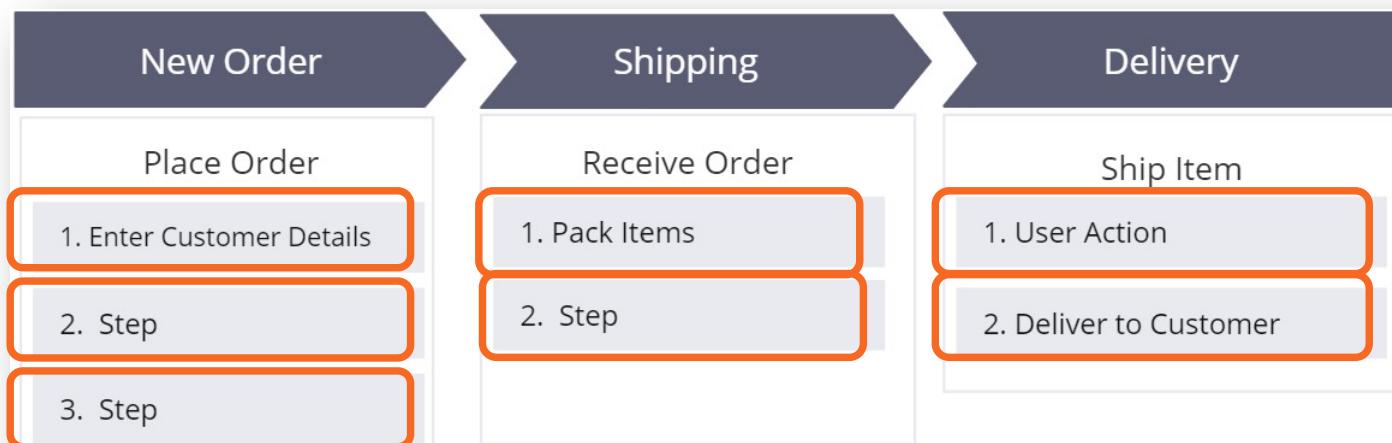
1. Hover over a stage
2. Select *Add process* to add a sequential process or *Add parallel process*
3. In the text field, replace the default process label with a descriptive name



Steps

Definition

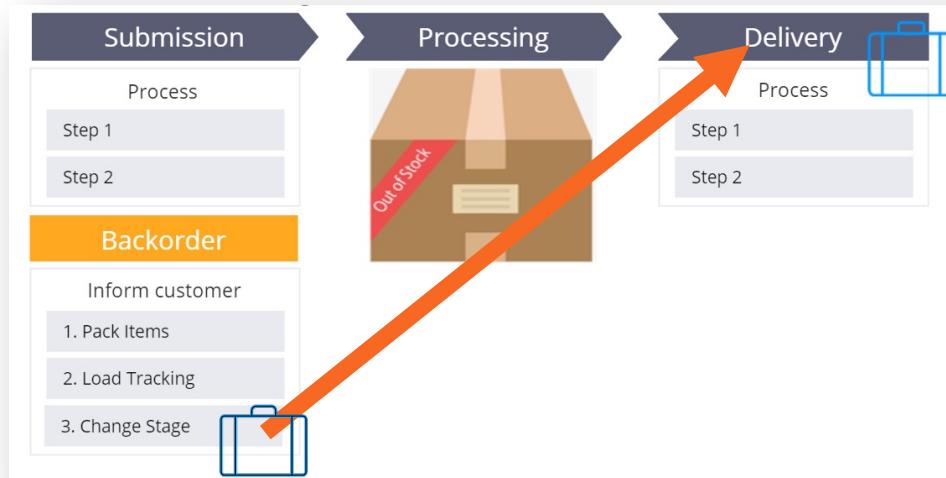
- A step is the smallest element of your business process and represents a single task.
- A step can be a user action or an automation that an application performs.
- Users' complete steps to case moves closer to its resolution and to achieving the business goal.



Change stage step

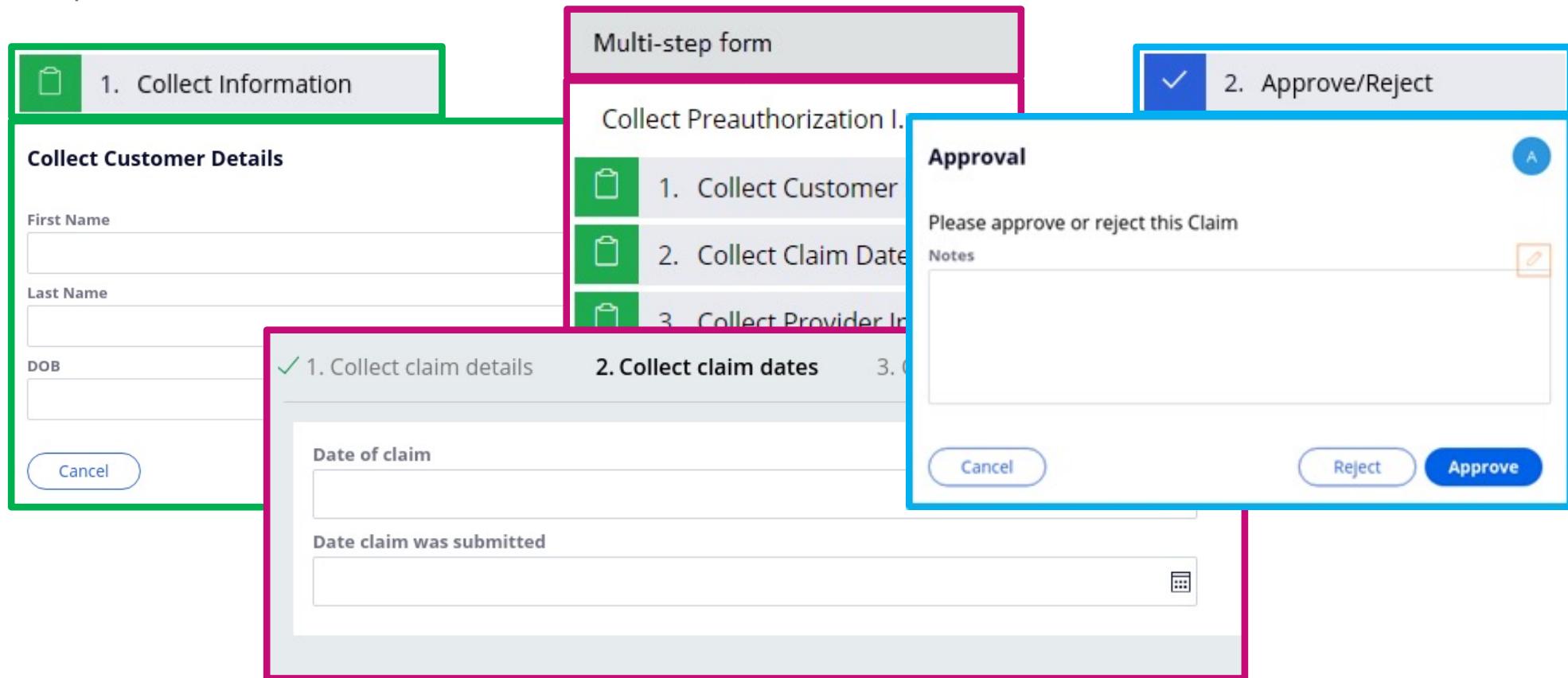
Description

- A **Change Stage** step is helpful to automate case flow to and from alternate stages.
- This type of configuration is most useful for automating transitions to and from alternate stages.
- Use a change stage step to automatically transition the case flow to a specified stage.



End user step types

Implementation



System (automations) step types

Implementation

 1. Send email

Step

Send to *

Email address

+ Add recipient

Send a common email to all recipients

Subject *

Message content

[Compose](#)

Message preview will appear here

Include attachments

Audit note

 2. Create case

Step

Create the following case *

Select...

Audit note

 3. Change stage

Step

Stage

Candidate Rejection

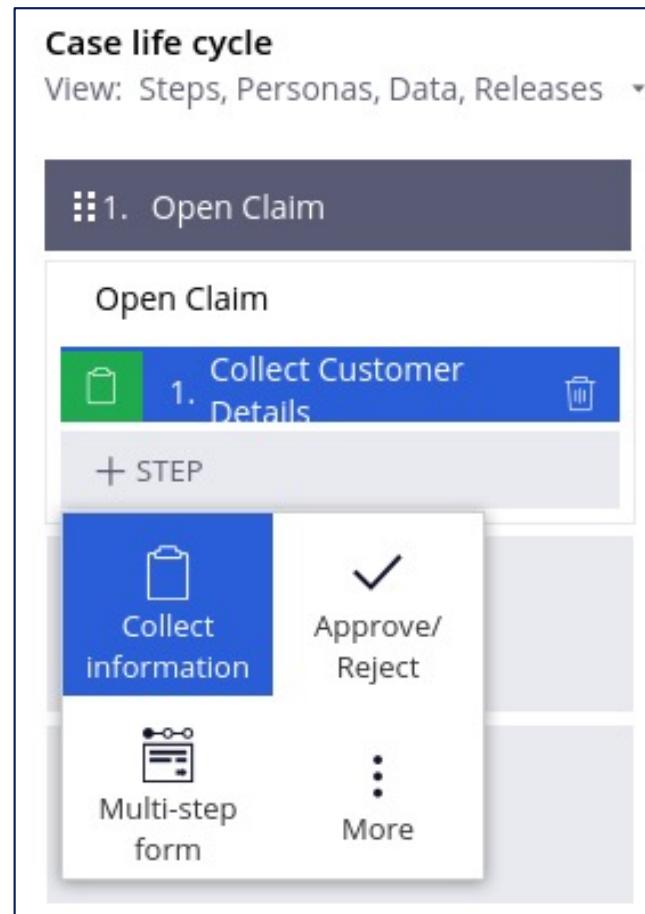
Audit note

Adding steps

Navigation

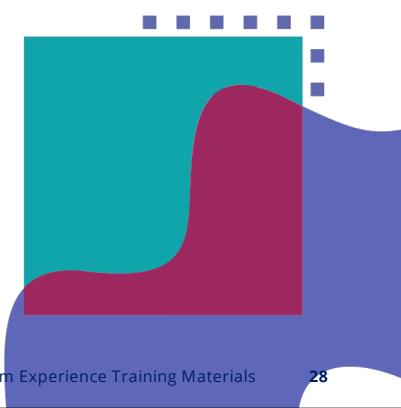
From App or Dev Studio:

1. Click Case types
2. Click the name of the case type that you want to open
3. In a process, click Step
4. Select a step type that you want to add
5. In the text field, replace the default step label with a descriptive name



Best Practices

- A case life cycle design should be easy to interpret.
- Name stages by using a noun or noun phrase to describe context.
- Name processes and steps by using the verb + noun phrase.
- As much as possible, try to use no more than two words.
- Use names that are meaningful and relevant to business users
- Consider organizing your case type so that it can be visualized on a single screen or page.
- Consider breaking case types down into parent and child cases if it becomes too complex with too many stages, processes or steps.
- Consider limiting number of stages to seven, plus or minus two to support good UX design.



Skill Mastery

Understand:

- case life cycle design
- how stages, processes and steps relate
- the difference between primary and alternate stages
- the resolution stages
- how to use the case designer
- the Pega Platform model-driven application development approach

Business Requirements



SKILL
LESSON

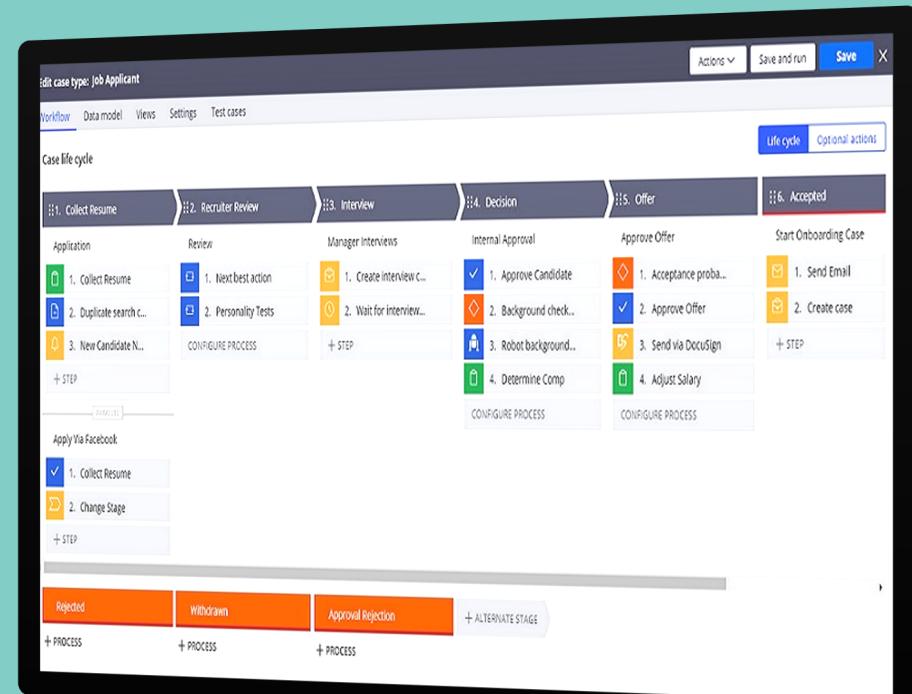
Gathering Requirements



Overview

Business application requirements gathering is famously hard. The ability to directly and visually capture business requirements change the way business and IT collaborate.

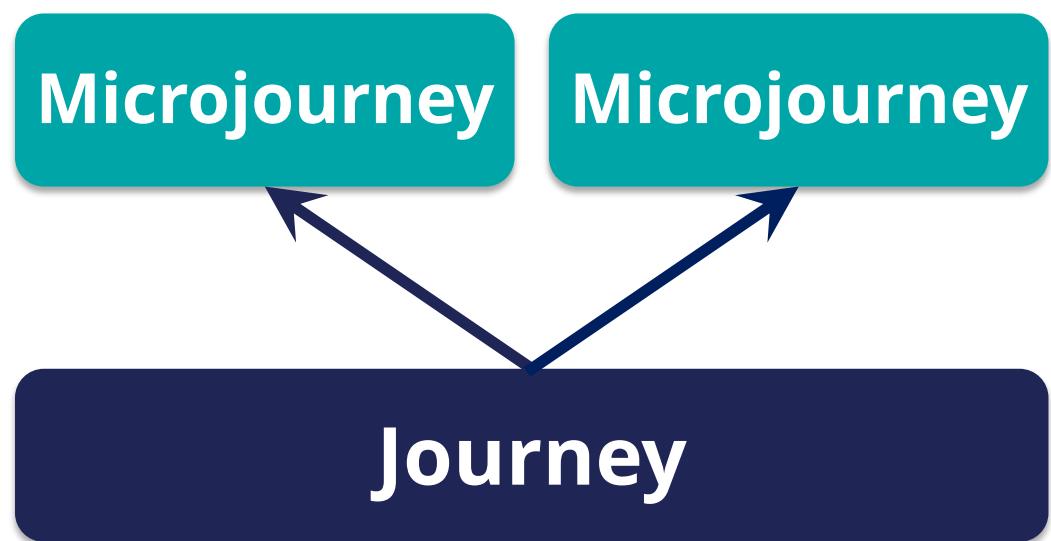
With Pega, journey/microjourney relationships, team member roles, and App Studio prototyping provides the efficiency, agility, reuse, standardization and accuracy needed to build a successful solution.



Journey and Microjourney

Definition

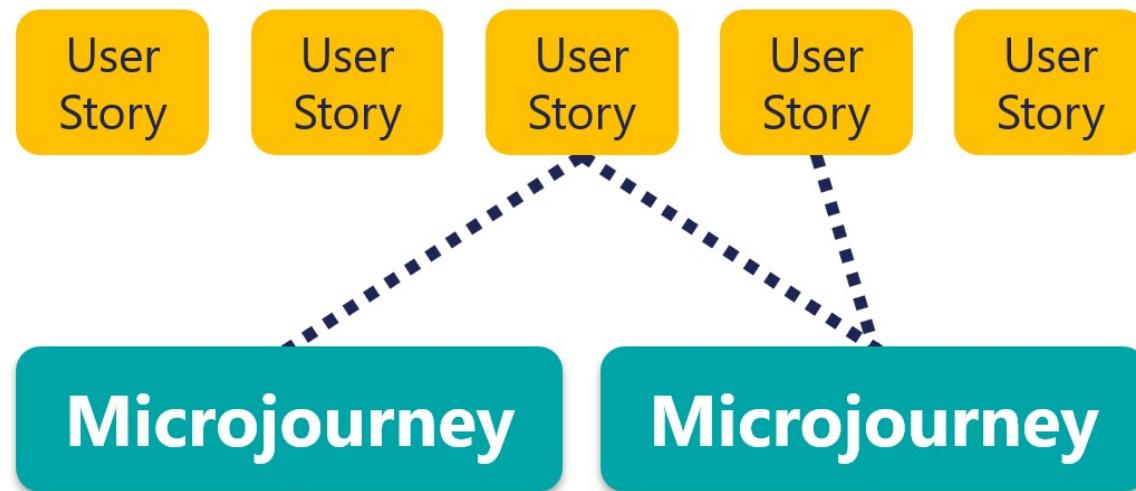
- A **Journey** is the series of interactions between a customer and an organization that occur as the customer pursues a specific goal. (Journeys are too big for an MLP.)
- **Microjourneys** are the series of interactions between **one type of customer** and an organization **through a specific delivery channel** that occur as the customer pursues a specific goal.



User Story

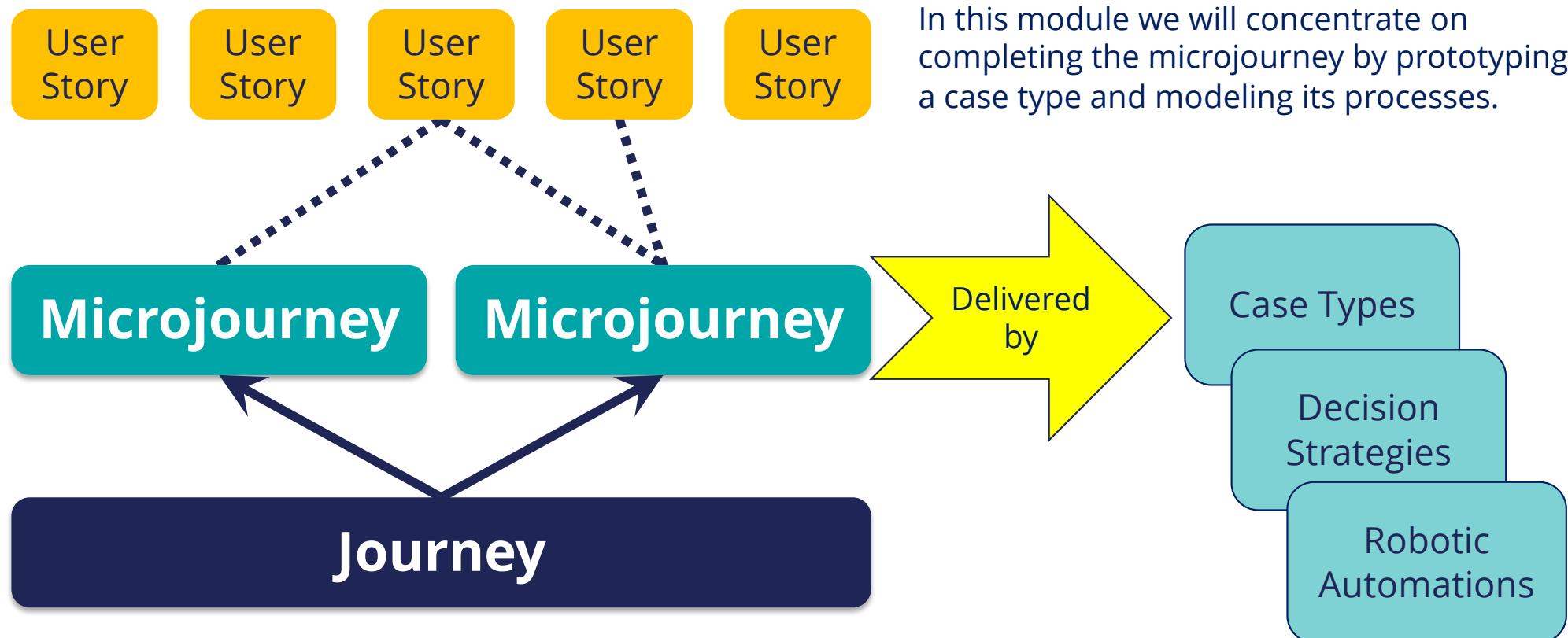
Definition

- A user story is a written description of an individual task to be performed.
- User stories can be unique to a microjourney or commonly used amongst multiple microjourneys.



Microjourney prototype

Microjourney relationships:

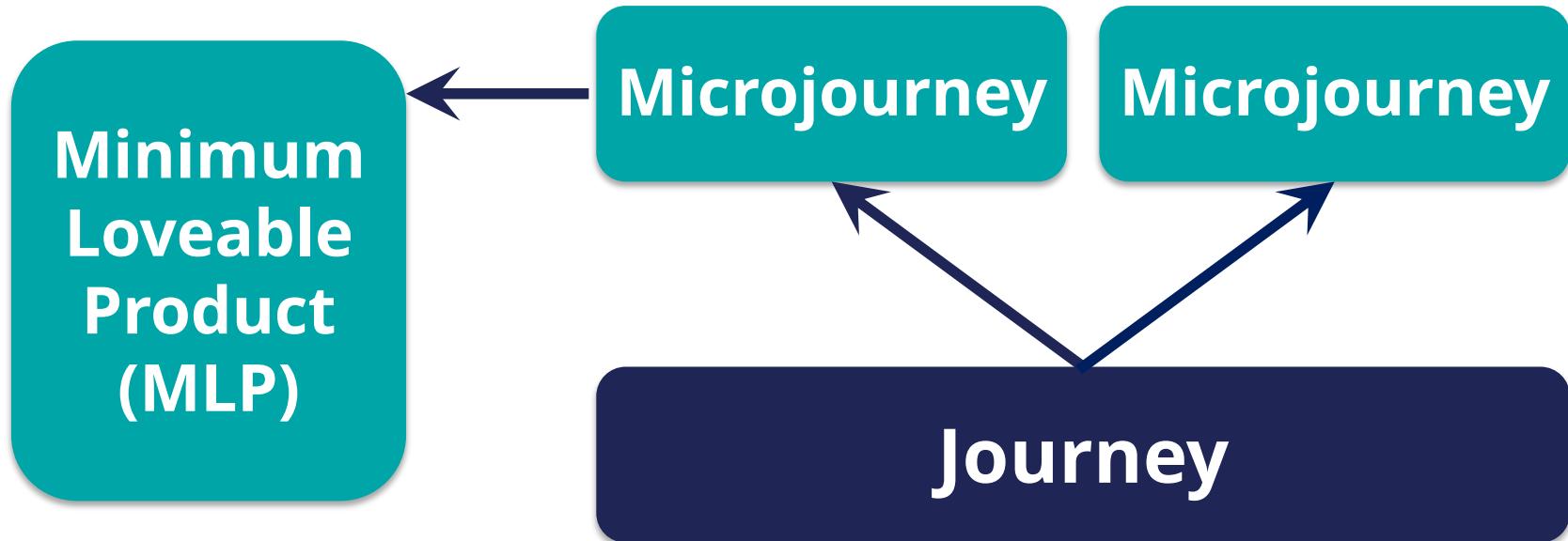


Implementation

Definition

Identify your MLP

- A program begins with a definition of its Minimum Loveable Product (MLP)
- An MLP is composed on one or more Microjourneys supported by in the MLP release



Business Solution

Use Case

- An initial conversation occurs with the business unit citizen developer and subject matter experts who determine the business solution will be automated in Pega.
- The business transaction will be implemented as a Case Type.
- During the conversation, a citizen developer can use the case designer to define the new features to be implemented in the current release:
 1. Define case type(s)
 2. Visually capture the “big picture” of the workflow “to-be” as seen by the stakeholders.
 3. Document pertinent business rules that will affect the case lifecycle.

REMINDER: Design thinking is a philosophy and approach to problem solve and innovate.

It typically consists of four to five days of intense workshops:

Day 1) Map; Day 2) Sketch; Day 3) Decide; Day 4) Prototype; Day 5) Test

Roles on a Pega Project

Pega projects involve different actors to achieve successful outcomes. In this lesson, you learn about the participants in a Pega project and how these participants work together to solve business problems.

- List common roles on a project.
- Match high-level tasks and responsibilities to the appropriate role.
- Identify the business purpose of assembling a project team.



Common Project Roles

Definition

The mix of team members and roles differs depending on the project goal. Some common roles found on a Pega project are shown below. Project roles should not be confused with sprint team members, although the roles may overlap.



Flexibility of Project Role Names

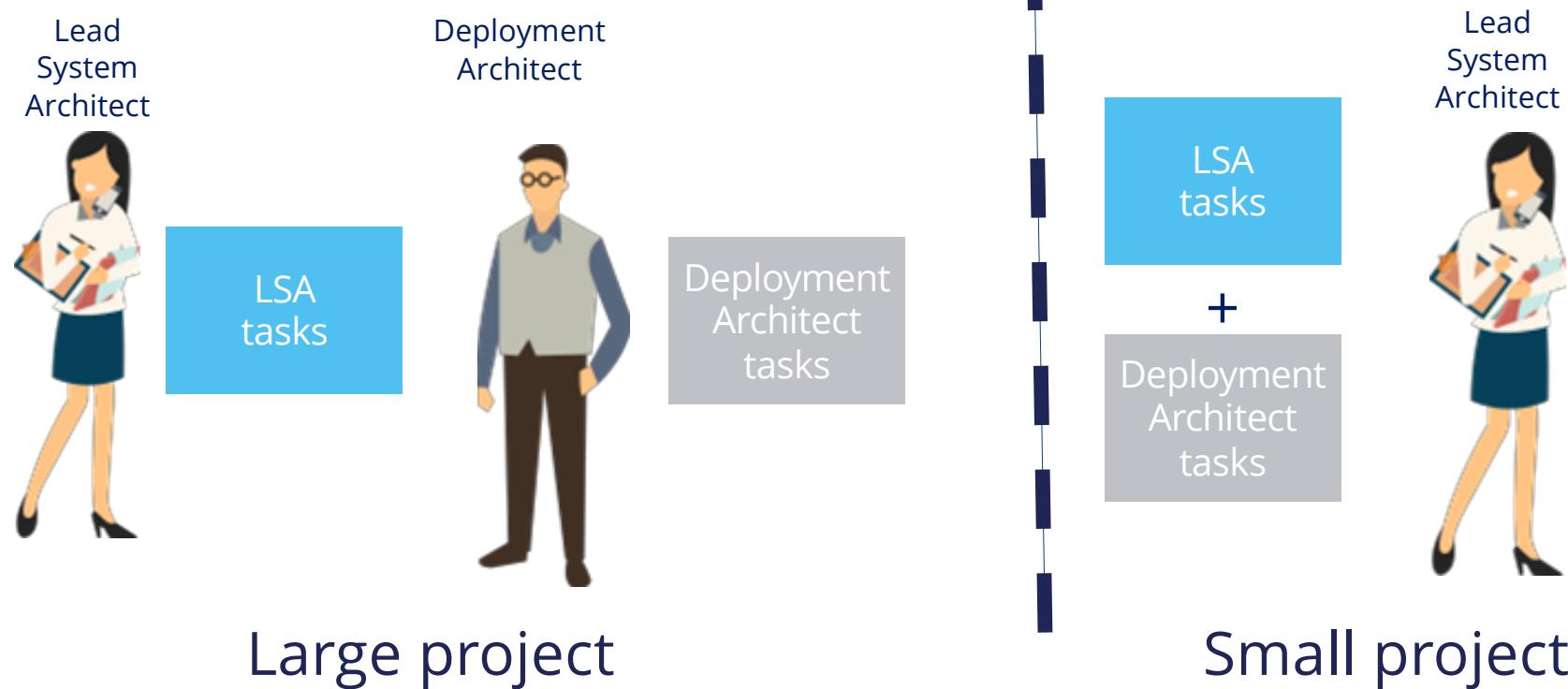
Pega roles may be referred to by different names depending on the organization implementing the project.



Overlap of Roles Tasks

Description

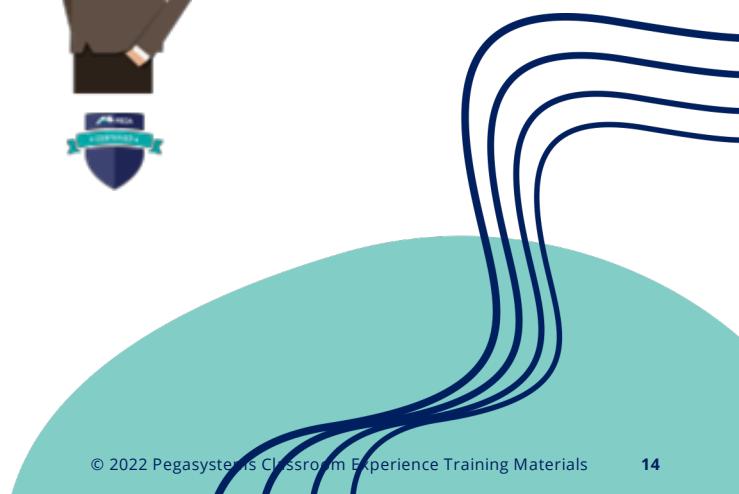
The size of a project affects how roles are distributed throughout the team.



Co-Production

Description

Pega strongly recommends co-production to help customers gain proficiency, enabling them to support and expand their application moving forward.



Design and Development Team Members

- Citizen Developer
- Business Architect
- System Architect

Citizen Developer(s)

They are non-technical business users who participate in application development.

These developers bring valuable insight and knowledge about the business' needs.

They are Pega enabled to use the App Studio to assist the Business Architect with prototyping case lifecycle design.

Citizen developers do not require any previous software development experience.

<https://community.pega.com/blog/what-citizen-developer>



Business Architect(s)

They subject matter experts and stakeholders to understand directly with business needs.

They are Pega enabled team members who use the App Studio primarily and occasionally the Dev Studio to prototype application functionality, define business rules, service level agreements, and processes.

Business Architects also assist in requirements gathering and documentation.

Business Architects have good analytical skills and often have a background in business analysis and/or systems analysis.

They may or may not possess previous software development skills.

[What's the difference between a Business Analyst and a Pega Business Architect? Click this link to find out!](#)

Stakeholder System Architect(s)

They are Pega enabled developers who configure and “fine tune” the application technically.

A lead or principal architect (LSA) designs the overall architecture while senior (SSA) and system architects (SA) configure assets such as user interface forms and automated decisions.

They contribute object-oriented design and technical implementation skills.

System Architects typically have software development or coding backgrounds.

[Click this link to read a blog entry about a Pega Instructor's journey from SA, to SSA, to LSA.](#)



Other Relevant Design Sprint Team Members

Description

- **Lead System Architect** - helps the team build the prototype
- **Senior Business Architect** - plays a crucial role in creating the final solution
- **Senior System Architect** - not necessarily involved in the Design Sprint, but does support prototype completion



Lead System Architect



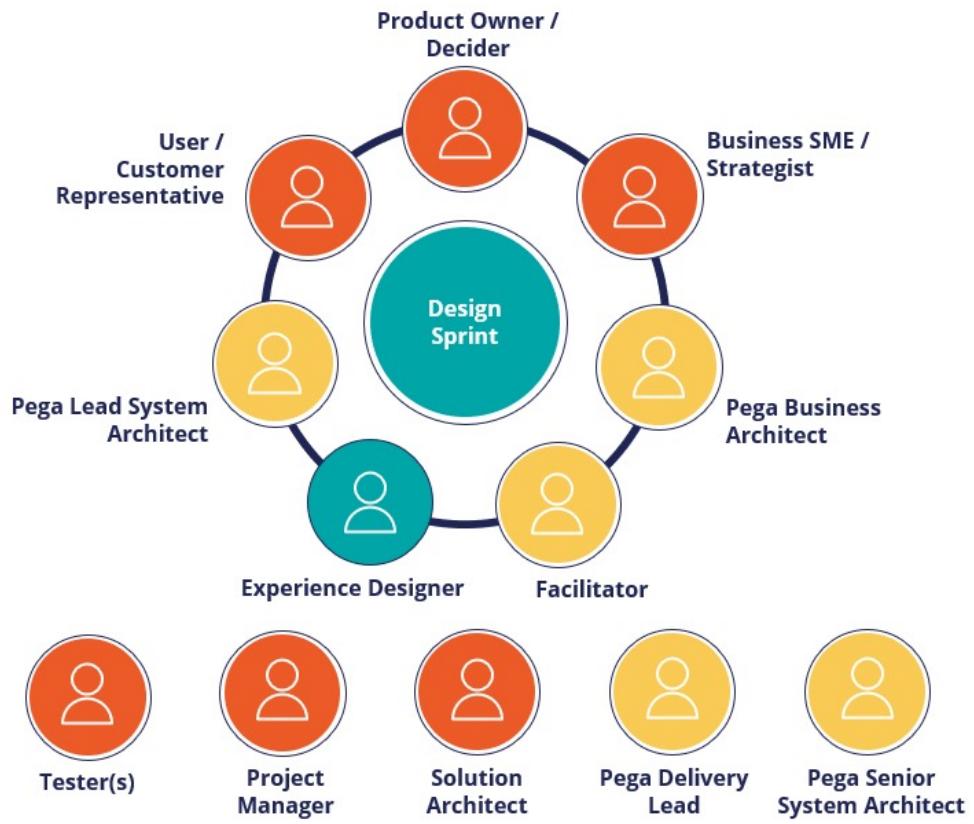
Senior Business Architect

Other Relevant Design Sprint Team Members (cont.)

Description

Best practice:

There should be no more than seven team members in a Design Sprint.

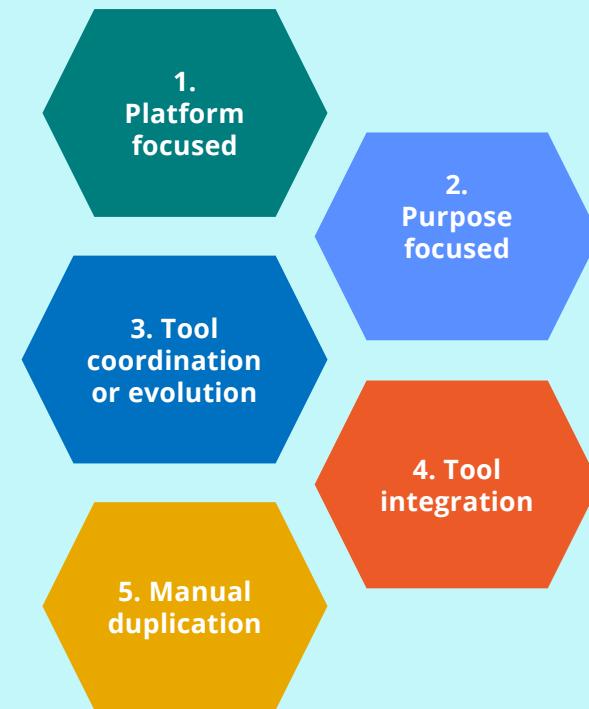


Description: DCO session

Description

- A time-boxed session (ideally 2 hours or less) where the SA, BA, and CD work directly with the product owner and SMEs to capture specific information so the design team can:
 - Contextually design the business vision for the “to-be” processes using visual models.
 - Document the epic feature(s) and/or associated user stories.
- Document the epic feature descriptions and associated user stories.

Common approaches to applying DCO





SKILL
LESSON

Branch Development



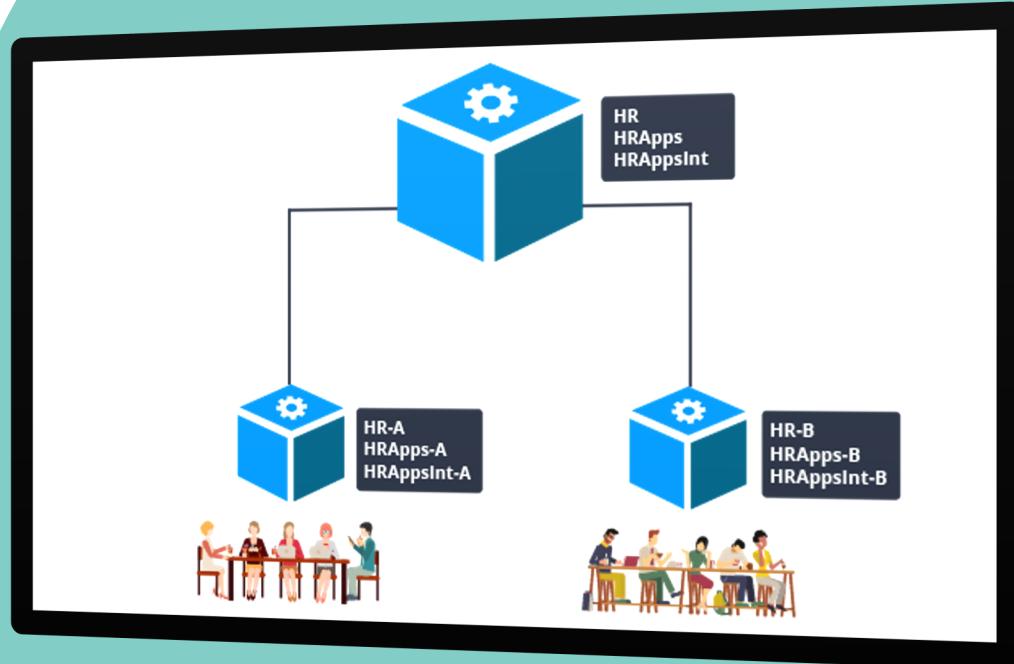


Overview

Implementing Pega applications using branches to develop software in parallel as a version-controlled environment is the standard.

Teams can develop and contribute to a single application in a branch while another team develops in a different branch. Both share the same base rulesets.

When required, all updates are merged into the base ruleset.



Use case

- The branching approach offers the most granularity and requires developers to select the appropriate branch for every single rule creation or update.
- Branches are based on three approaches:
 - To implement a feature
 - To implement a user story
 - To implement a sprint
- A branch should contain all the necessary rules for the implementation of the feature, story or sprint.

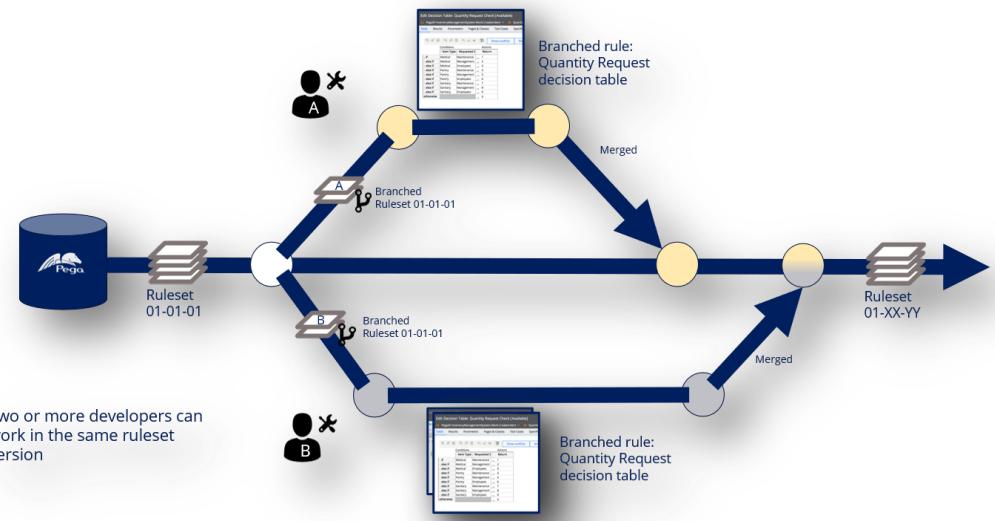




Branch

Definition

- A branch is a container for rulesets with rules that are undergoing rapid change and development.
- Create a branch for each team which will allow each team to work within an isolated space (the branch) without affecting other teams.
- Each team works on their changes independently.
- All team members can see each other's work while isolating the development changes from other teams.
- Changes do not affect other teams until the changes are stable, conflicts are resolved, and approval is granted to make the changes available to all development teams.





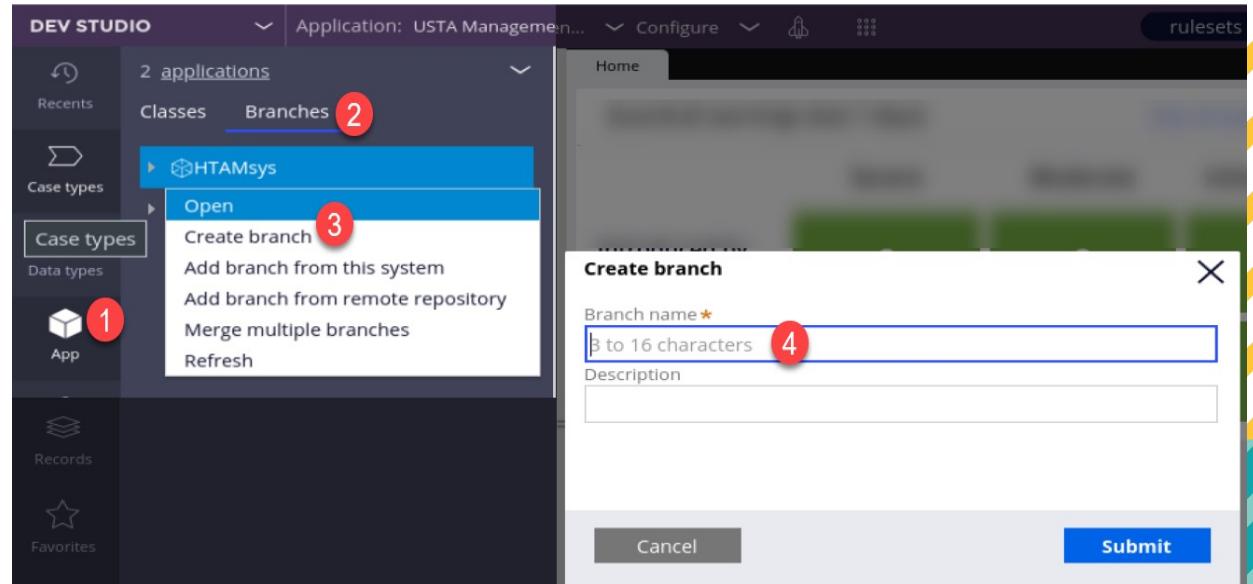
Create a new branch

Navigation

Create branches in an application to develop rules in them. The system automatically creates branch rulesets when saving a rule into a branch.

From Dev Studio:

1. Click the App explorer
2. Select Branches in the top left panel
3. Right-click on the application and select Create branch
4. Enter a name and description



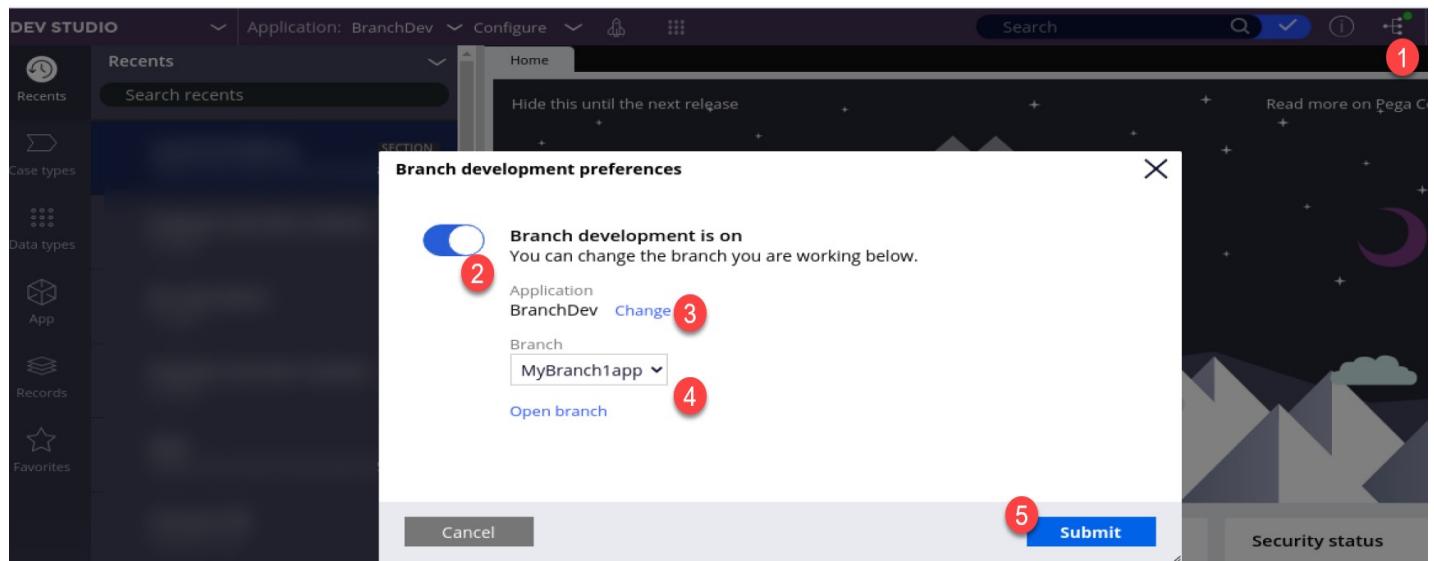


Enable branch development

Navigation

From App or Dev studio:

1. In the top right corner click on the branch icon
2. Toggle the slider so it turns blue to indicate "enabled"
3. Select the application by clicking on the *change* link
4. Select the branch from the drop-down list
5. Click submit





Add or delete a branch

Navigation

Add or delete branches from the application using the **application rule form**.

From Dev studio:

1. Select App explorer
2. Click on Branches
3. Select the application to display the application rule form
4. Click Add branch to add an existing branch or click the trash icon to delete an existing branch

The screenshot shows the Pega Dev Studio interface with the following details:

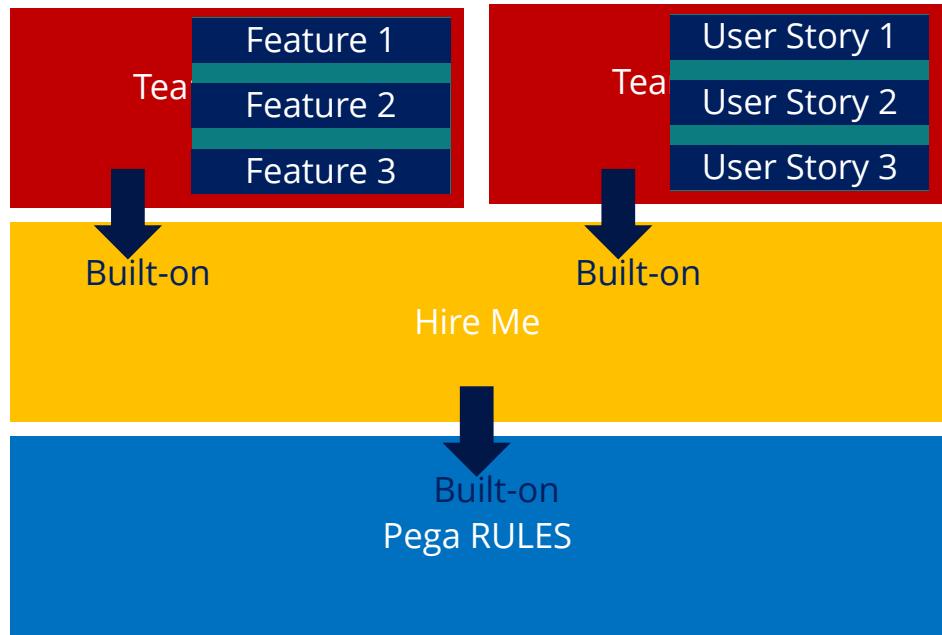
- Left Sidebar:** Shows '3 applications' with 'HireMe' selected. The 'Branches' tab is highlighted with a red box.
- Central Area:** Displays the 'Edit Application: HireMe' screen. The application ID is 'HireMe • 01.01.01' and the rule set is 'HireMe'. A message at the top states: 'This record has 1 info warning (including 1 unjustified) [Review/Edit]'. The tabs include 'Definition', 'Cases & data', 'Application wizard', 'Documentation', 'Integration & security', and 'History'.
- Right Panel:** Contains sections for 'Development branches' (with a red box around it), 'Enabled components' (empty), 'Application rulesets' (listing four items: 1. HireMe:01-01, 2. HireMeInt:01-01, 3. PA:01-01, 4. PAInt:01-01), and 'Presentation' (skin set to 'HireMe').



Branch ruleset

Description

- Create branches and develop rules in branch rulesets
- The rulesets associated with a branch are called branch rulesets.
- Modifications using rulesets in different branches, allows team members to work on isolated rulesets without impacting development of other features and avoid overwriting results and generating errors.
- A branch ruleset:
 - Is based on (branched from) another ruleset
 - Contains rules that are in active development in the associated branch
- After creating and developing rules in branch rulesets, branches can be manipulated in several ways such as creating branch reviews with other users, deleting branches from the system, and locking branches before merging.





Branch quality and content

Description

- View information about your branch, such as the rules that it contains and whether branches have been reviewed.
- Detect and address potential problems by monitoring the condition of a branch.
- Analyze the quality of your branch in detail by viewing branch quality metrics:
 - Guardrails
 - Test coverage
 - Unit testing
 - Failed unit tests
 - Rules without unit tests
 - Merge conflicts
 - Uncovered rules

The screenshot displays two main windows from the Pega DEV STUDIO:

Top Window (Branch: User-Story-1):

Content Branch quality

Content

Name	Type	Class	Ruleset	Updated by
StatusBasedDicount	Decision Table	PA-HireMe-...	HireMe	admin@hiremedev
TC_StatusBasedDiscount	Test Case	PA-HireMe-...	HireMeTest	admin@hiremedev

Bottom Window (Branch: Test):

Content Branch quality

Guardrails
Weighted score 75
Warnings: 3 Severe: 1

Test coverage
Rules covered: 0%
0 of 2 executable rules
Metrics derived from latest application coverage report.

Unit testing
Rules with unit tests: 0%
Test pass rate: 0%
0 of 1 supported rules
0 of 1 test cases

Merge conflicts Warnings Uncovered rules Failed unit tests Rules without unit tests

Total	Severe	Moderate
3	1	0

Rule list:

Rule name	Rule type	Ruleset	Severity	Warning type	Justified	Warning message	Introduced by
pzDeleteMobileApp Activity	Pega-ProcessEngine_Branch_test	Severe	Maintainability	Yes	Pega Unit Warning	Kamil Dudek	
pzDeleteMobileApp Activity	Pega-ProcessEngine_Branch_test	Informational	Maintainability	No	Obj-Delete-By-Handle Unnecessary Step Page KiranYNG		
pzDeleteMobileApp Activity	Pega-ProcessEngine_Branch_test	Informational	Security	No	pzPreventSensitiveDataInLogs	Kamil Dudek	



Branch operations

Description

- After you create branches and develop rules in branch rulesets, you can work with branches in several ways.
 - Reorder branches
 - Lock branches
 - Merge branches
 - Delete branches

Merge Branches

User-Story-1

Source ruleset	Target ruleset	
HireMeTest	HireMeTest	<input type="button" value="Create new version"/> Version 01-01-02
Total candidates 1	No conflicts or warnings found.	<input type="checkbox"/> 1 Password
Source checked out 0	Target checked out 0	<input type="checkbox"/> Lock target after merge <input type="text" value="Enter password (recommended)"/>
Source ruleset	Target ruleset	
HireMe	HireMe	<input type="button" value="Create new version"/> Version 01-01-02
Total candidates 1	1 Conflict, 0 Warnings	<input type="checkbox"/> 1 Password
Source checked out 0	Target checked out 0	<input type="checkbox"/> Lock target after merge <input type="text" value="Enter password (recommended)"/>

Keep all source rules and rulesets after merge



Skill Mastery

Understand:

- the purpose of a branch
- why a branch is used
- how to create a branch
- how to enable a branch
- how to review branch content and quality
- branch operations



Schematic

Design Layer

Branch

Rules Layer

Ruleset(s)

Java HTML CSS XML Javascript JSP

1 1 0 0 1 0 1 0 1 0 1 1 1 0 0 0 1



SKILL
LESSON

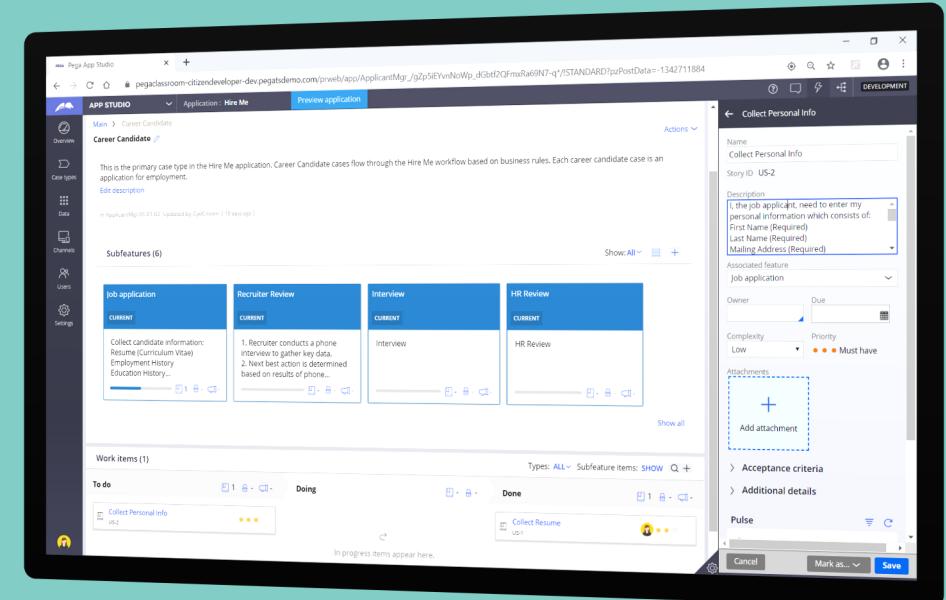
Documenting Requirements



Overview

The Pega development environment provides various tools for documenting requirements.

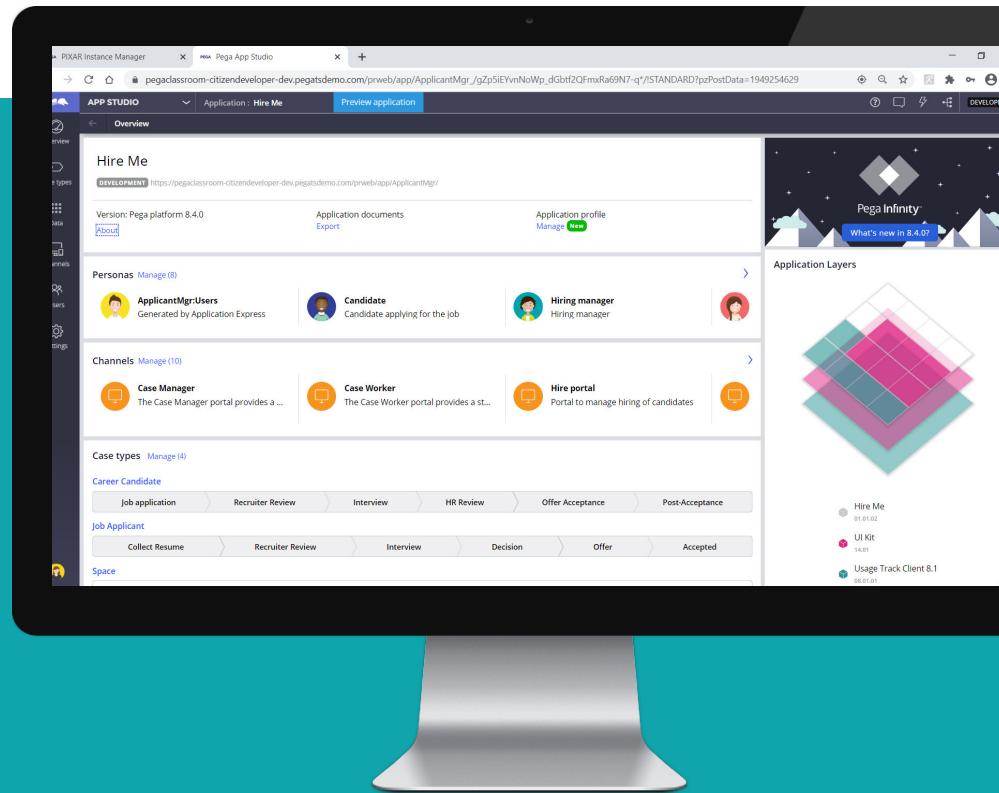
The Application Profile features map to document feature requirements and Agile Workbench to create and assign work items for the application configuration.



Application Overview Landing Page

Description

- The Application Overview landing page gives you easy access to key business assets.
 - Manage Application profile
 - Inventory
 - Feature map
 - Work items



Application Profile

Description

The Application Profile landing page tabs:

- Inventory - Persona and Data relationships
- Feature Map – Feature/subfeature, and work item relationships
- Estimator - greater accuracy and efficiency to estimate projects, more intuitive and automated

Improves communication with interested parties and provides a universal understanding of requirements.

The image displays three separate screenshots of the Pega APP STUDIO Application profile interface, each highlighting a different tab: Inventory, Feature map, and Estimator. The Inventory tab shows a Persona section with 'No Data'. The Feature map tab shows a 'Features (2)' section with 'Career Candidate' listed. The Estimator tab shows a 'Release' dropdown set to 'Unassigned'.

Inventory

Description

- The Inventory landing page is a tool for implementing the Pega Express methodology in delivering your projects in a goal-oriented and no-code way
- With the Inventory page, you can view lists of all personas and data objects in your application.
- You can quickly access the information that you need by grouping items in the list by different criteria, such as by persona or case type.

Inventory						
Persona		Data				
		Channel	Case type	Stage	Release	Status
Channel : Connect on the move App						Total 5
Approver	Connect on the move App	Hire	Decision	MLP 1	TO DO	⋮
Hiring Manager	Connect on the move App	Hire	Decision	MLP 1	TO DO	⋮
Hiring Manager	Connect on the move App	Hire	Decision	MLP 2	TO DO	⋮
Staffing consultant	Connect on the move App	Hire	Interview	MLP 2	TO DO	⋮
Staffing consultant	Connect on the move App	Hire	Offer	MLP 1	TO DO	⋮
Channel : Email for Job						Total 3
Applicant	Email for job	Hire	Collect resume	MLP 1	TO DO	⋮
Employees	Email for job	Hire	Collect resume	MLP 1	TO DO	⋮
Staffing consultant	Email for job	Hire	Offer		TO DO	⋮

Agile Workbench Access

- In the **App Studio**, click the Agile Workbench lightning bolt in the upper right corner.
- In the **Dev Studio**, the same icon can be found in the toolbar in the bottom-left corner.
-

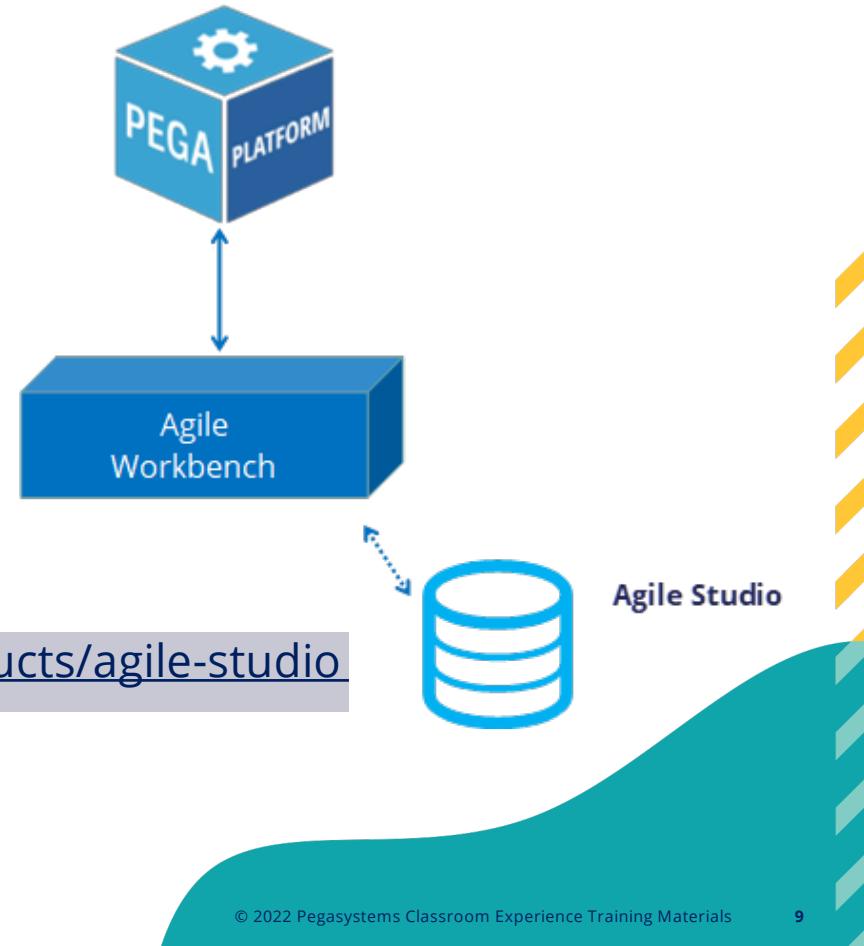


Agile Workbench vs. Agile Studio

Description

- Agile Workbench is part of the Pega Platform.
 - By default, Agile Workbench stores and tracks work items (user stories, bugs, feedback items).
- Agile Studio is an agile project management system that enables your application development teams and your stakeholders to execute your projects using the industry best practice Scrum methodology.
 - Agile Studio is a separate Pega product used for issue tracking.
 - For more information, go to:

<https://community.pega.com/knowledgebase/products/agile-studio>

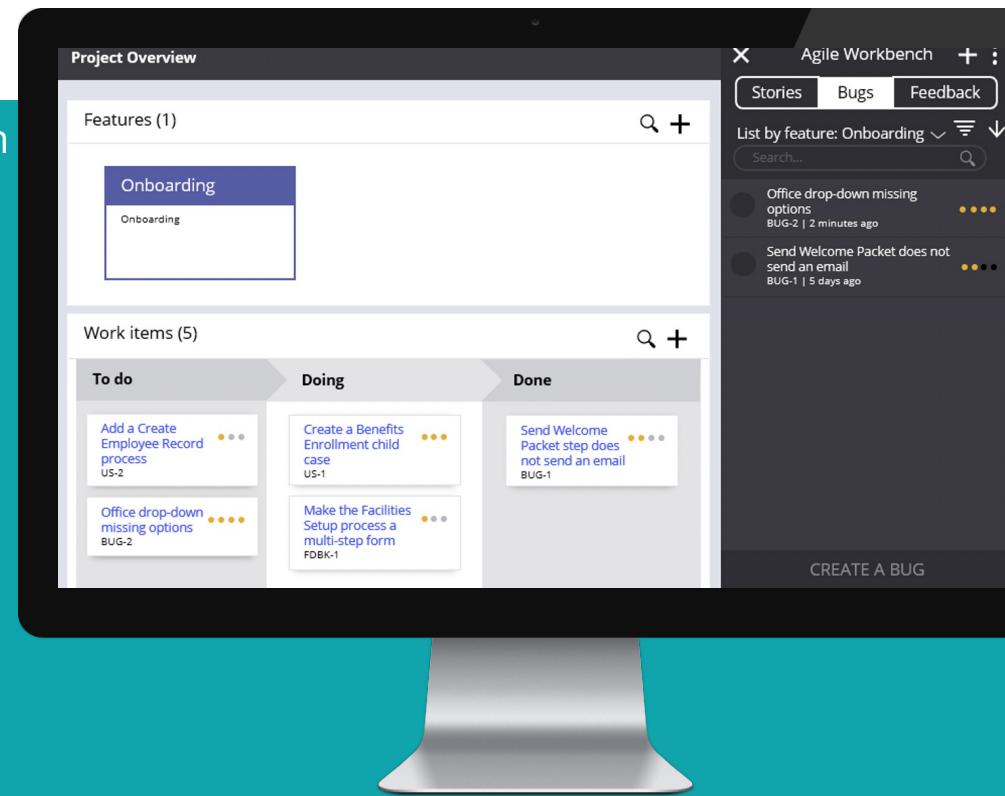


Agile Workbench

Description

Capture real-time information about your application to increase application development efficiency.

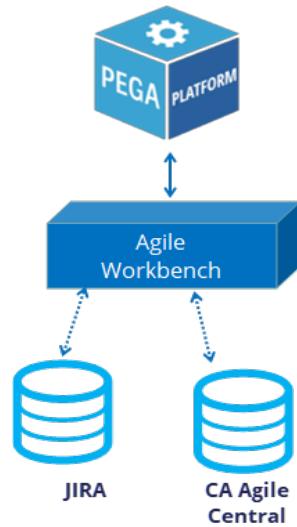
- Supports Direct Capture of Objectives.
- Supports agile development.
- Track feature development.
- Create or import, organize and assign user story backlog.
- Register and assign bug items including video and screenshot capture.
- Manage feedback and development status.



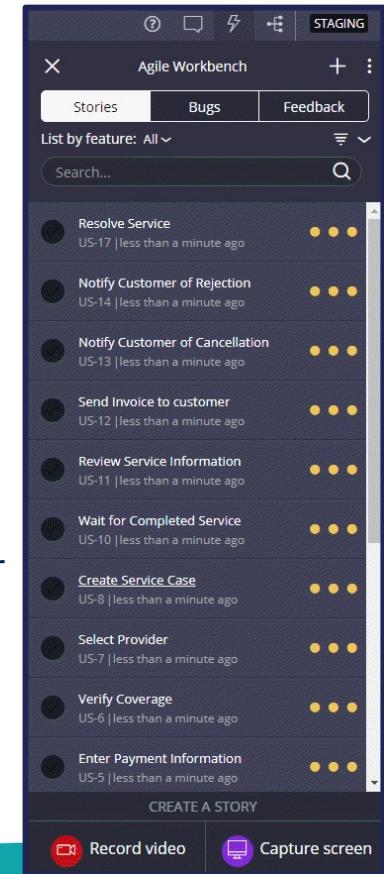
Integration and Implementation

Description

- **Agile Workbench** supports integration with JIRA and CA Agile Central for issue tracking.
- **Integration packages** are available from the Pega Marketplace.



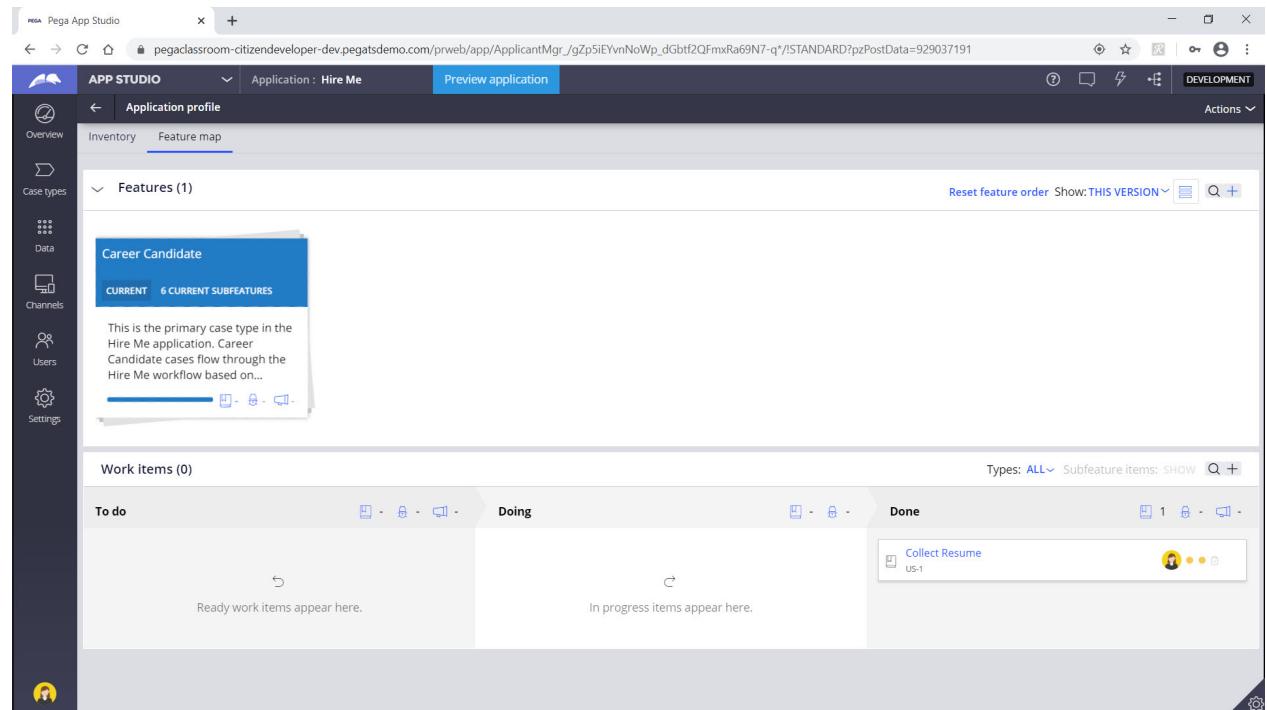
- **Using Agile Workbench in App Studio**, you can map development tasks to a feature by creating stories, bugs, and feedback.
- **App Studio** allows you to track development changes in the context of a feature.
- **Features** in an agile-development model help you plan and implement the capabilities for your application.



Feature

Definition

- A feature is a capability that you implement in your application.
- By defining features that are linked to rules and work items, you can improve the traceability of capabilities to their respective implementations.



Feature

Description

- A feature is a capability that you implement in your application.
- Features in an agile-development model help you plan and implement the capabilities for your application
- Features may be simple or complex. Complexity is visually represented by “nesting” subfeatures.

Case Type

Represented as a Main Feature

▼ Features (1)

Career Candidate

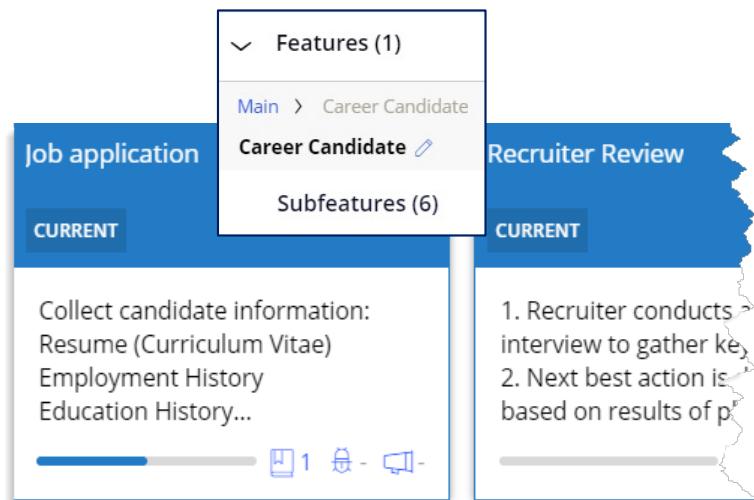
CURRENT 7 CURRENT SUBFEATURES

This is the primary case type in the Hire Me application. Career Candidate cases flow through the Hire Me workflow based on...



Stage

Typically as a subfeature



Process

May be nested as subfeature

▼ Features (1)

Main > Career Candidate > HR Review

HR Review

Subfeatures (1)

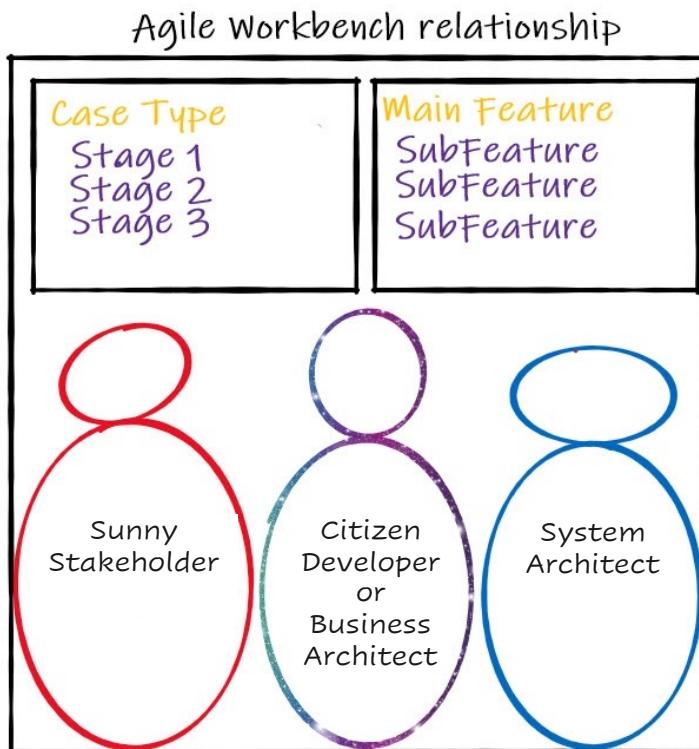
Background Check

CURRENT

Background Check

Feature-driven Development Enabled

Description



Ensure the microjourney meets all the customer user's needs

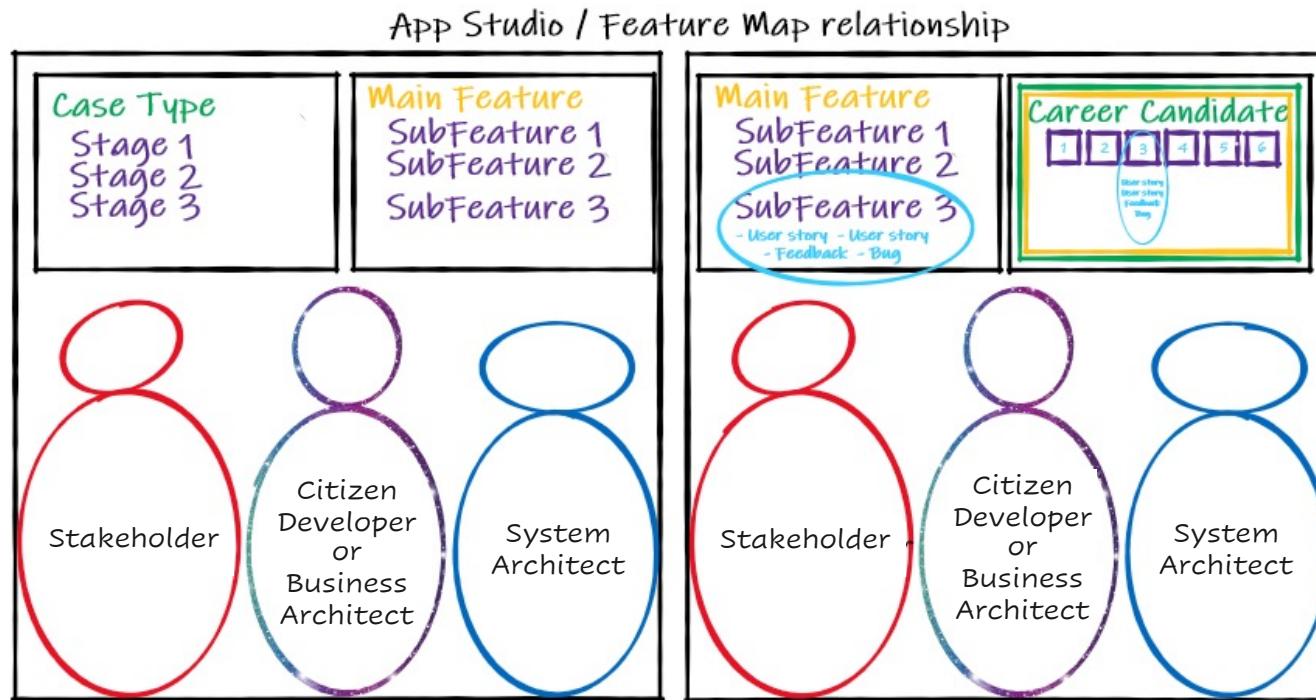
- Engage stakeholders using a feature-driven application development process:
 - Adopt a feature-driven development
 - Creating, aligning, and assigning feature-encapsulated work-items
 - User Story
 - Bug
 - Feedback

https://community.pega.com/sites/default/files/help_v84/express/procomhelpmain.htm#project-delivery/application-features/proj-implementing-a-feature-tsk.htm

Adopting feature-driven development

Description

Develop capabilities in the context of a feature to maintain functional requirements and project status directly in your application.

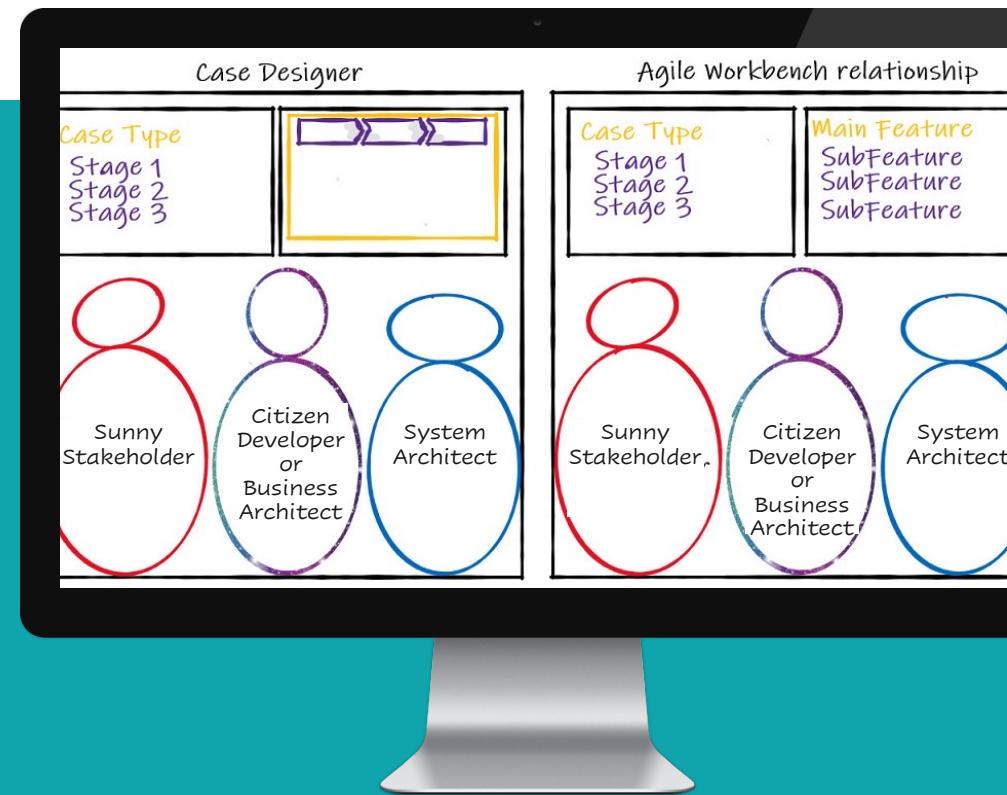


Capture and document business requirements

Navigation

Managing Application Profile:

- Feature Map
 - Main Feature
 - Subfeature(s)
 - Nesting Subfeature(s)
- Feature
 - Description
 - Associated Work Items
 - User Story(ies)
 - Bug(s)
 - Feedback item(s)



Estimator

Definition

Use the Estimator feature to create a more accurate project plan and to enhance communication with stakeholders.

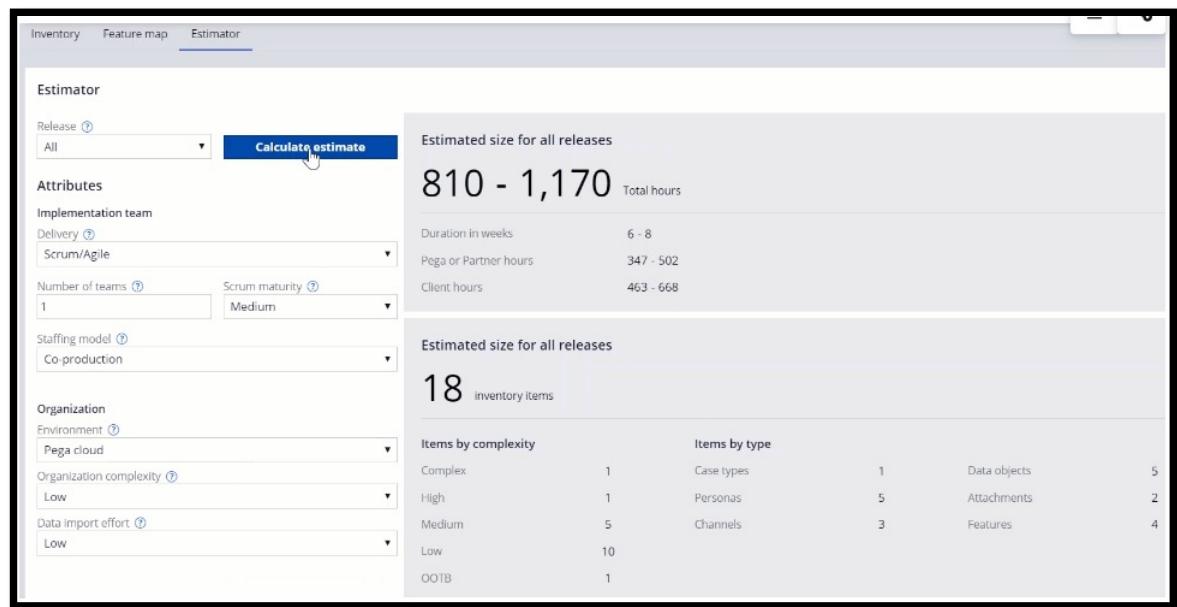
Use the Estimator to clearly state how much time and effort is required to deliver an application that reflects a specific design and required elements.

The screenshot shows the 'Application profile' screen with the 'Estimator' tab selected. On the left, there's a sidebar with 'Attributes' and dropdown menus for 'Delivery' (Scrum/Agile), 'Number of teams' (1), 'Scrum maturity' (High), 'Staffing model' (Co-production), 'Organization' (Existing on prem), 'Organization complexity' (Low), and 'Data import effort' (Low). In the center, under 'Estimated size for', it shows '0 - 0 Total hours'. Below that, there are two tables: one for 'Duration' and one for 'Pega resources'. The 'Duration' table shows '0 - 0' for Duration in weeks, Pega or partner hours, and Client hours. The 'Pega resources' table lists roles and counts: Project Delivery Leader (1), Lead / Senior Business Analyst (1), Lead System Architect (1), Senior System Architect (2), and UX/UI designer (0.5). To the right, there's another table for 'Client resources' with counts: Certified Business Analyst (1), Certified System Architect (2), Test lead (1), and Tester (2). At the bottom, there are two more tables: 'Level of effort for' (0 Inventory items) and 'Items by complexity' and 'Items by type' (both showing 0 for Complex, High, Medium, Low, and OOTB categories).

Estimator

Description

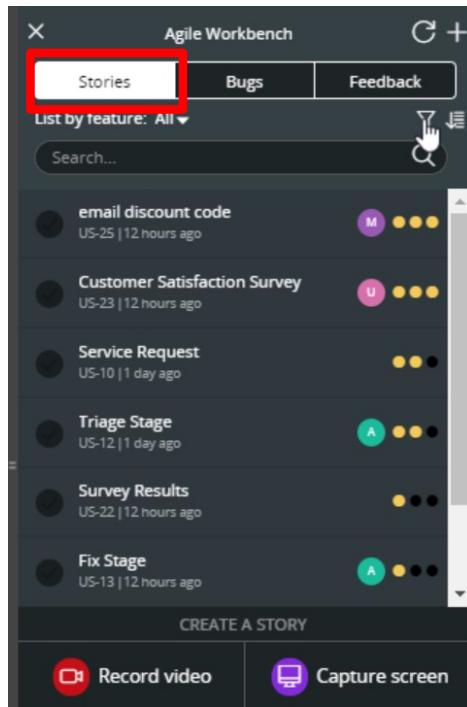
- A project estimate considers various factors:
 - complexity of the application elements
 - # of teams
 - development environment exists
- The creation of estimates in Pega Platform is automated.
- Estimation results maybe exported to .xlsx file.



Add Stories

Description

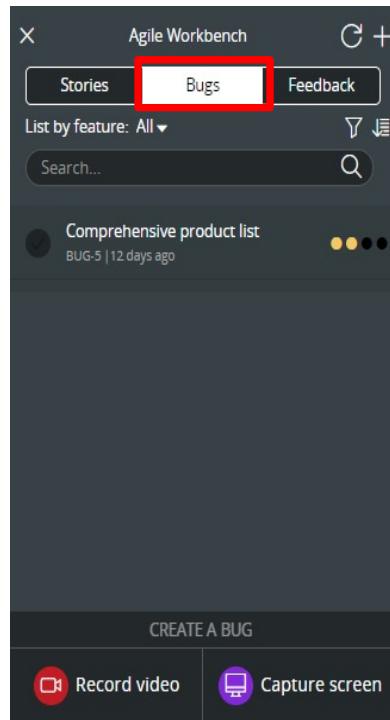
Team members may add user stories to the backlog and refine existing items for each application release.



The Pega App Studio interface shows the creation of a new story. On the left, the 'Workflow' tab is selected in the 'APP STUDIO' sidebar. A red box highlights the 'Case life cycle' section, which lists steps: '1. Interview' (selected), '2. Feedback', '3. Approve Schedule', and '4. Conduct Interview'. A red arrow points from the 'Feedback' step to a 'New Story' dialog on the right. The dialog is titled 'New Story' and contains fields for 'Name' (Schedule Interview), 'Story ID' (US-1), 'Description' (I, the human resources associate (HRA) need to be able to assign an interview date/time in the scheduling system.), 'Associated feature' (Interview), 'Owner' (Admin@Careers), 'Due' (7/1/2021), 'Complexity' (High), 'Priority' (Must have), and an 'Attachments' section with a 'Add attachment' button. A red box highlights the 'Save' button at the bottom right of the dialog.

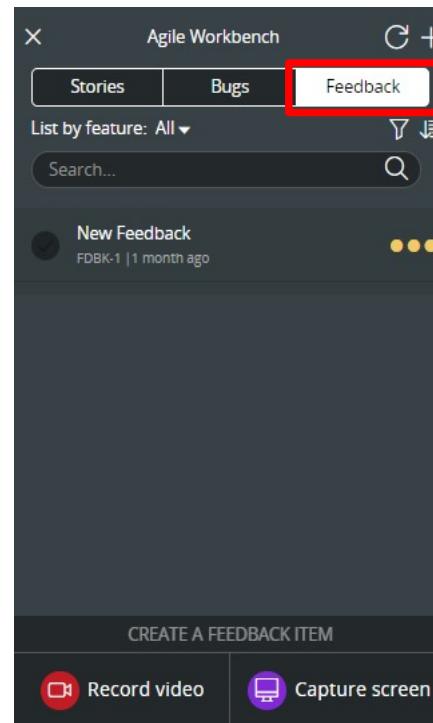
Record Bugs

- As you notice bugs and recognize changes to make, you record those bugs, feedback, and enhancements (user stories) in Agile Workbench.



Capture Feedback

- Capture feedback from a business stakeholder in real-time using Agile Workbench in Pega Platform.

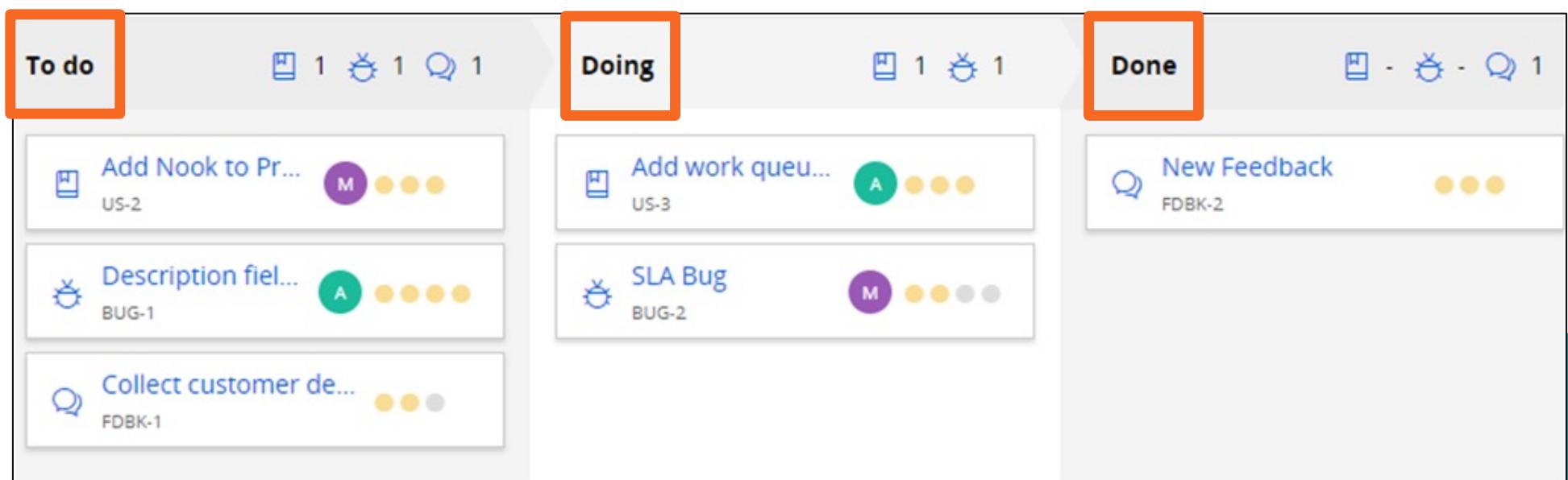


Track Status of Work Items

Implementation

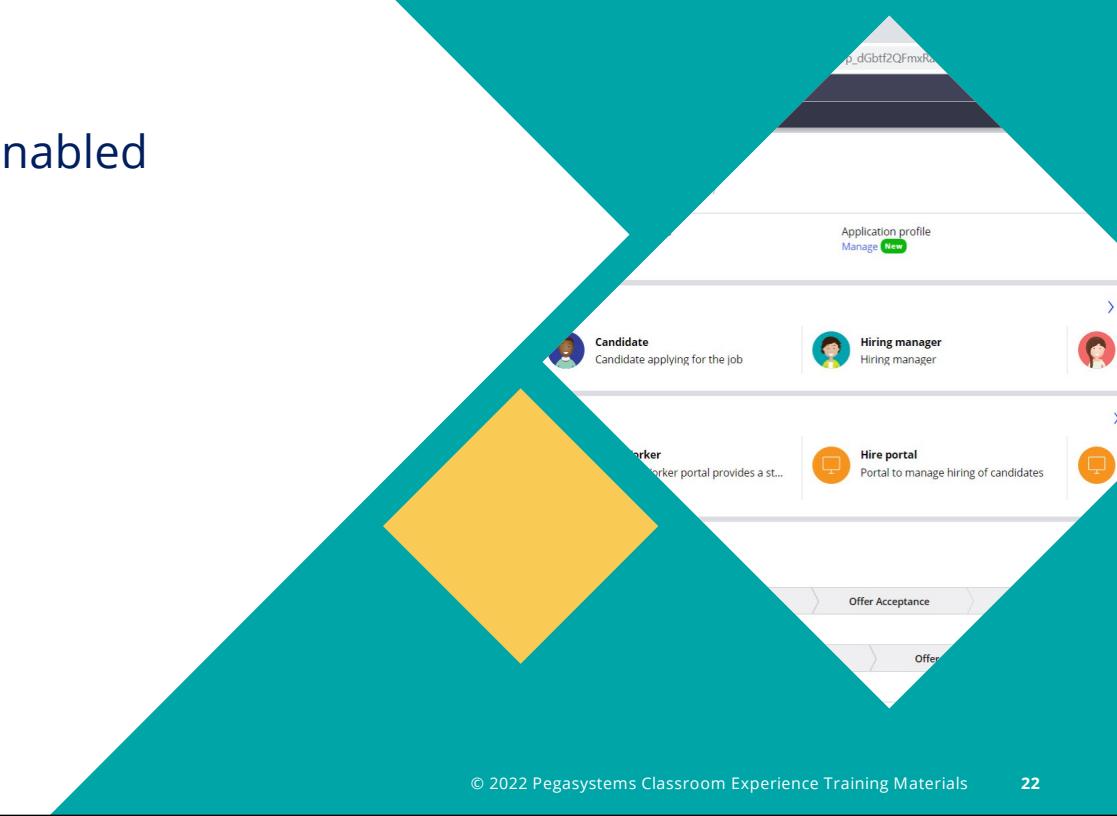
Track status in App Studio using the Feature Map landing page.

AppStudio > Overview > Application Profile [Manage] > Feature map



Skill Mastery

- Manage the Application Profile
- Manage Agile Workbench
- Become feature-driven development enabled



Case Management



SKILL
LESSON

Service Level Agreements

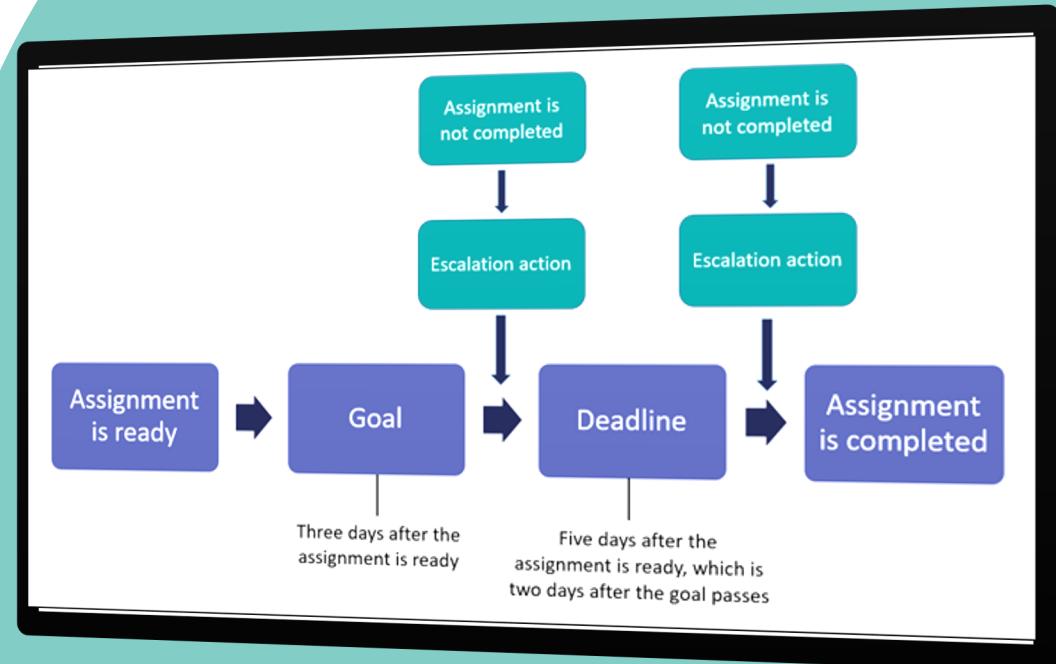
Getting work done timely



Overview

Organizations often establish service-level agreements to enforce on-time performance.

These obligations range from informal response-time promises to negotiated contracts.



Use Case

A commitment is made to customers and stakeholders that defines a timeline for delivering work or projects.

Encourage customer service representatives (CSRs) to resolve cases on time and enforce your service level agreements (SLAs) by setting goals and deadlines for a case type.

Standardize the way that case workers resolve cases in an application.

Bring visibility to unresolved assignments and cases in an application that need immediate action from case workers.



Service level agreement

Definition

- A timeline interval that can be applied to a case or elements in the life cycle that standardize resolution of a case.
- A **start** or initial defines whether the assignment is ready immediately or if there is a delay.
- A **goal** milestone defines how long the assignment *should* take.
- A **deadline** milestone defines the amount of time the assignment *may* take before it is late.
- A **passed deadline** milestone defines when to take further action because the assignment is past the deadline.



Passed deadline

Description

- Defines when to take further action because the assignment is past the deadline.
- The interval measures the **time that has passed since the deadline** for a still-open assignment.
- Configurable only in **Dev Studio**.
- Unlike the goal and deadline intervals, passed deadline can be configured to **repeat a fixed number of times**, or **repeat indefinitely** until the user completes the assignment.
- Each time the assignment passed deadline elapses, the **urgency** can continue to increase, and **escalation action** continues.



Service level agreements types

Description

- **Case-level service-level agreement**

- Timing begins when a case is created (or recently reopened) and stops when the case is resolved.

- **Stage-level service-level agreement**

- Timing begins when a case enters the stage and ends when the case leaves the stage.

- **Process-level service-level agreement**

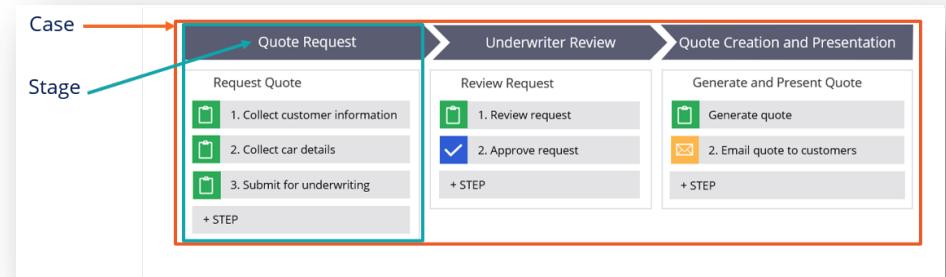
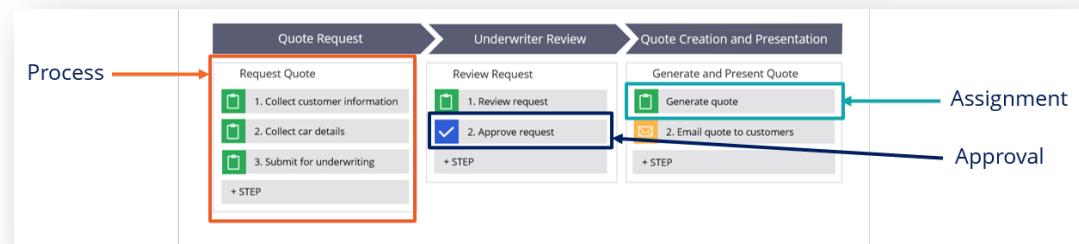
- Timing begins when the process is called and ends when the last step in the process completes.

- **Assignment-level service-level agreement**

- Timing begins when an assignment is created and ends after assignment completion.

- **Approval-level service-level agreement**

- Timing begins when the approval is called and ends after the approval completion.



Urgency

Definition

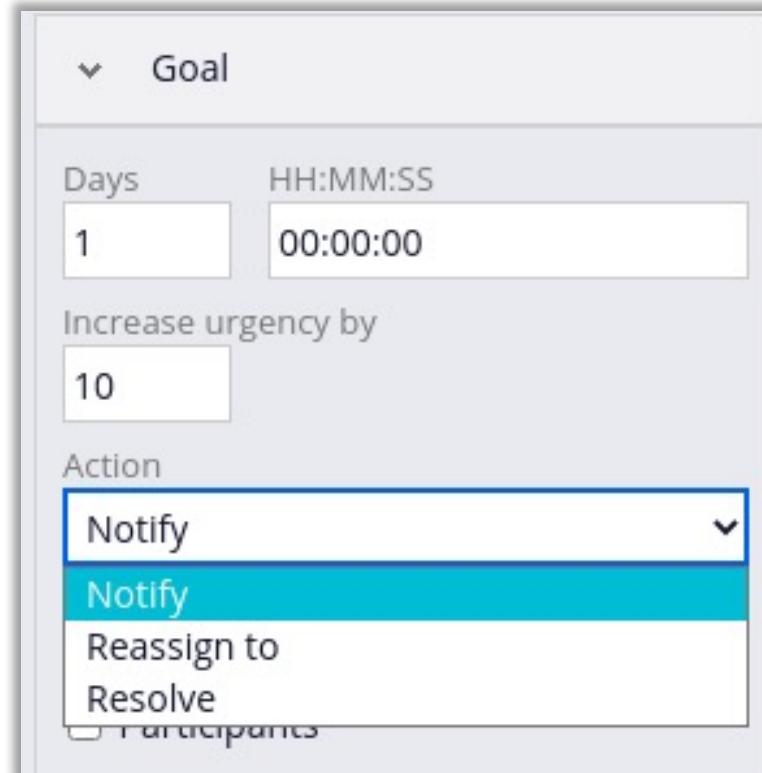
- Urgency is a numeric value, ranging from **0** to **100**, that is displayed in worklists and cases to indicate the importance of assignments.
- The initial urgency value is **10**. The maximum urgency is **100**.
- Use urgency to bring visibility to unresolved assignments and cases in your application that need immediate action from case workers.
- The higher the urgency, the more important it is to address the unresolved item.
- Typically, the urgency increases as an assignment advances to the next interval.



Escalation actions

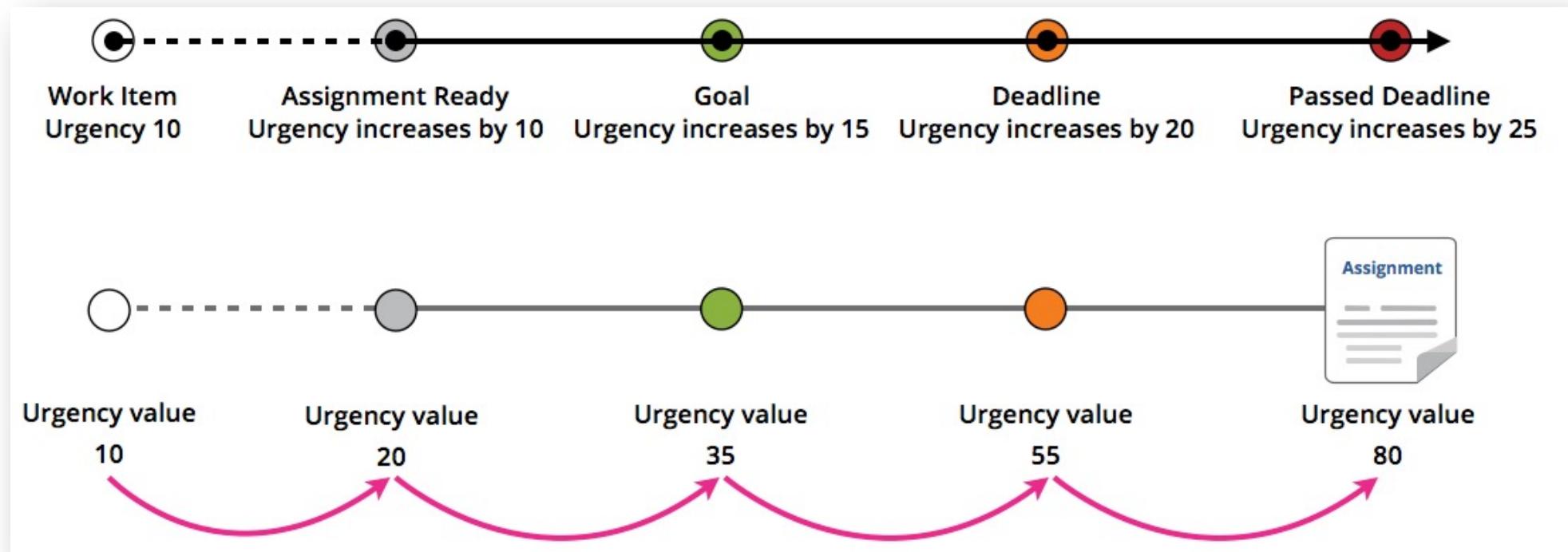
Definition

- Actions that your application takes to facilitate faster resolution times, based on a specified service-level agreement.
- You can configure escalation actions on service-level agreements to
 - Notify the assignee
 - Notify the manager
 - Notify the participants
 - Reassign the task
 - Resolve the case as the goals or deadlines occur.



Service level agreement

Implementation

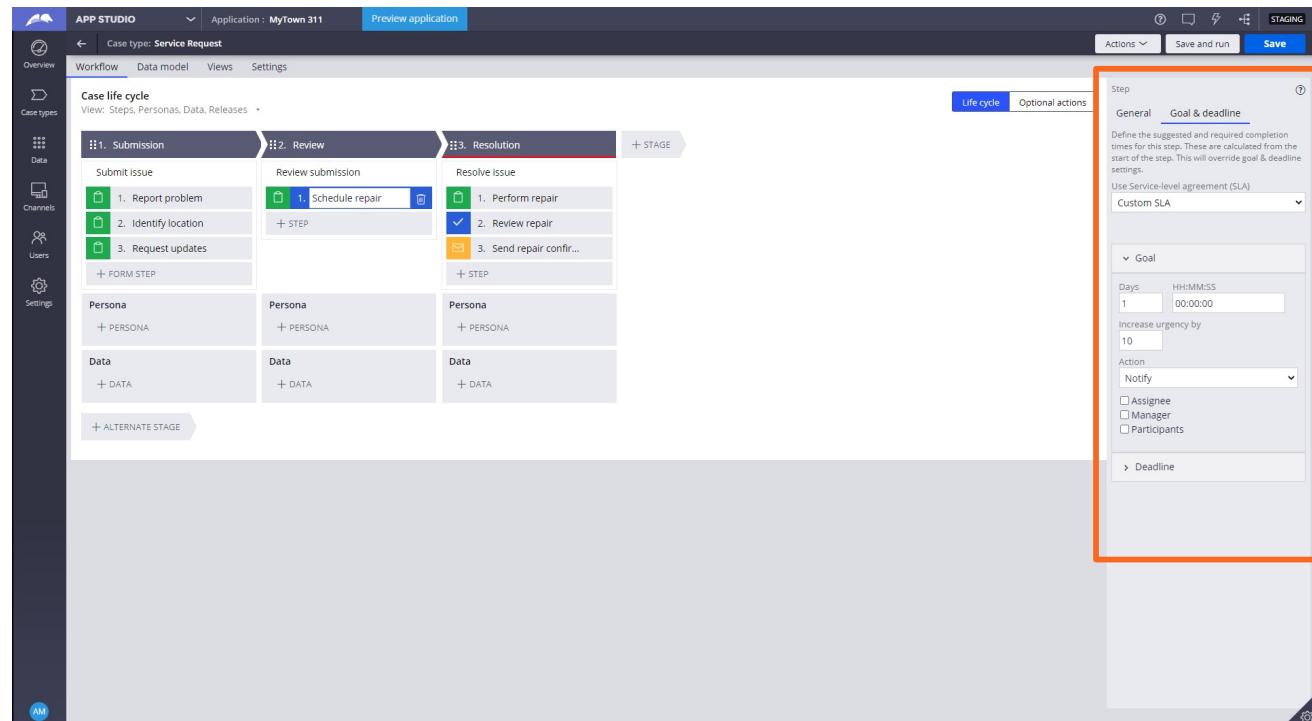


Configure SLA

Navigation

From App or Dev Studio:

1. Select the Case Type
2. Click on the Assignment step
3. In the configuration pane, set the values for Goal and deadline
 - Open the SLA rule form in Dev studio to configure the passed deadline.





Schematic

Design Layer

SLA configuration in right context options

SLA configuration in case settings

Rules Layer

Service level agreement (SLA)

Java HTML CSS XML Javascript JSP

1 1 0 0 1 0 1 0 1 0 1 1 1 0 0 0 1

SLA Unit Test and Debug

1. Configure a Service Level Agreement.
2. Run an instance of the case type.
3. Proceed to the case, step, process or stage for the configured SLA.
4. Check for the SLA time to be displayed with the correct time remaining in the header of the form.
 - The SLA intervals are displayed using only the largest time units configured and begin to count down as soon as the case is created.
 - As a result, the intervals are displayed as 14 minutes and 29 minutes, rather than the configured 15 minutes and 30 minutes.

The image displays two screenshots of a Pega case interface, specifically for a 'Service Request' case type. Both screenshots show the same case details: 'Service Request (SR-1)' status 'OPEN', a goal 'less than a minute from now', and a step 'Schedule repair' due 'DUE IN 4 MINUTES FROM NOW'. A red box highlights the goal and step labels. Below the step, a message states: 'Assign a due date for the issue' and 'The 'Schedule repair' step in the 'Service Request' case type does not yet have a view defined.' At the bottom right of each screenshot is a blue button labeled 'Advance this case'.

Service Request (SR-1) OPEN

less than a minute from now

Schedule repair
DUE IN 4 MINUTES FROM NOW

Assign a due date for the issue
The 'Schedule repair' step in the 'Service Request' case type does not yet have a view defined.

Cancel

AM

Service Request (SR-1) OPEN

The goal was less than a minute ago . The deadline is 3 minutes from now

Schedule repair
DUE IN 3 MINUTES FROM NOW

Assign a due date for the issue
The 'Schedule repair' step in the 'Service Request' case type does not yet have a view defined.

Cancel

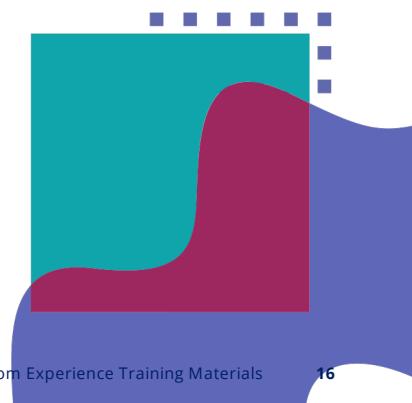
AM

Advance this case

Advance this case

Best Practices

If more than one escalation action is needed, configure *when* conditions to trigger some, but not all the escalation actions.



Skill Mastery

Understand:

- Service level agreements
- SLA types
- Goal and deadline
- Passed deadline
- Urgency
- Escalation actions



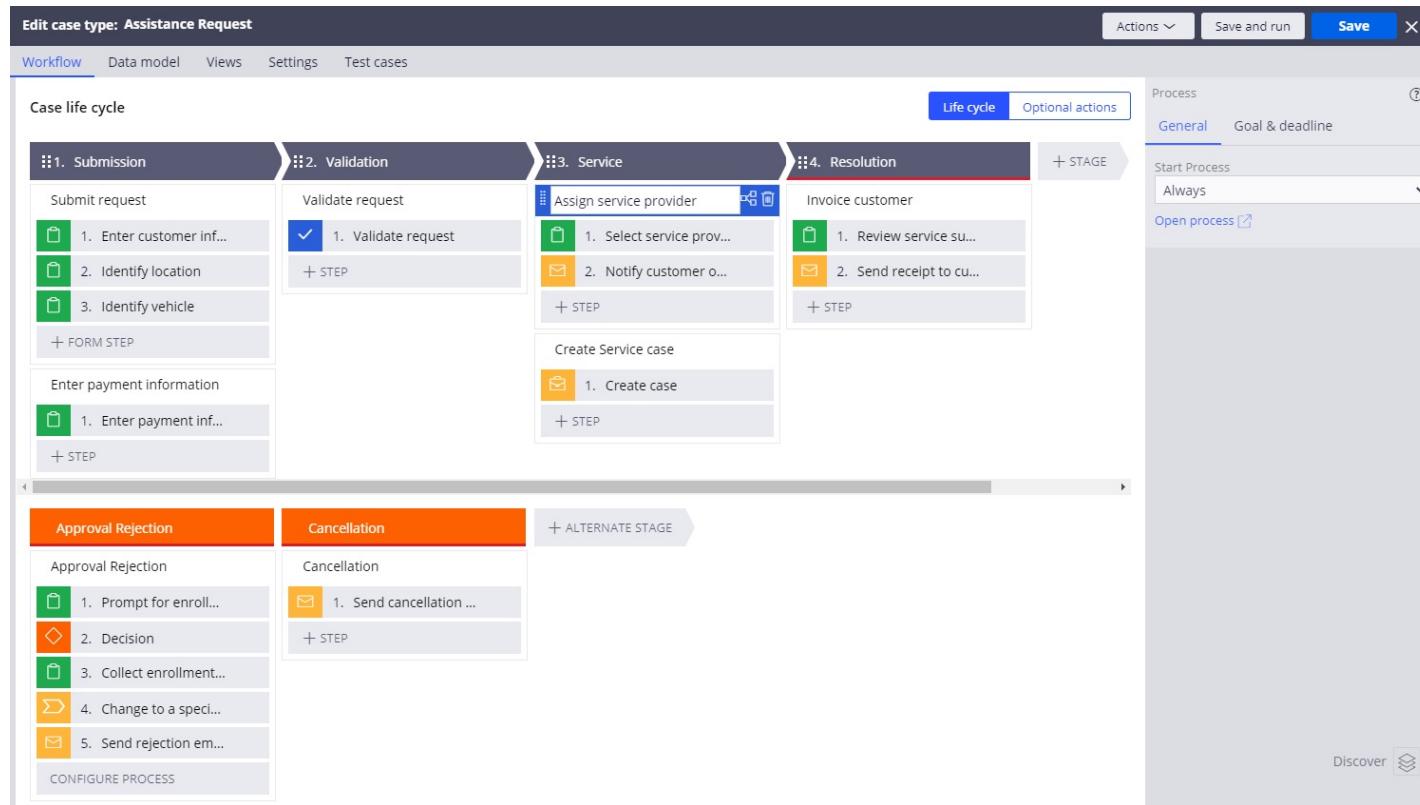
SKILL
LESSON

Child Cases and Wait Step



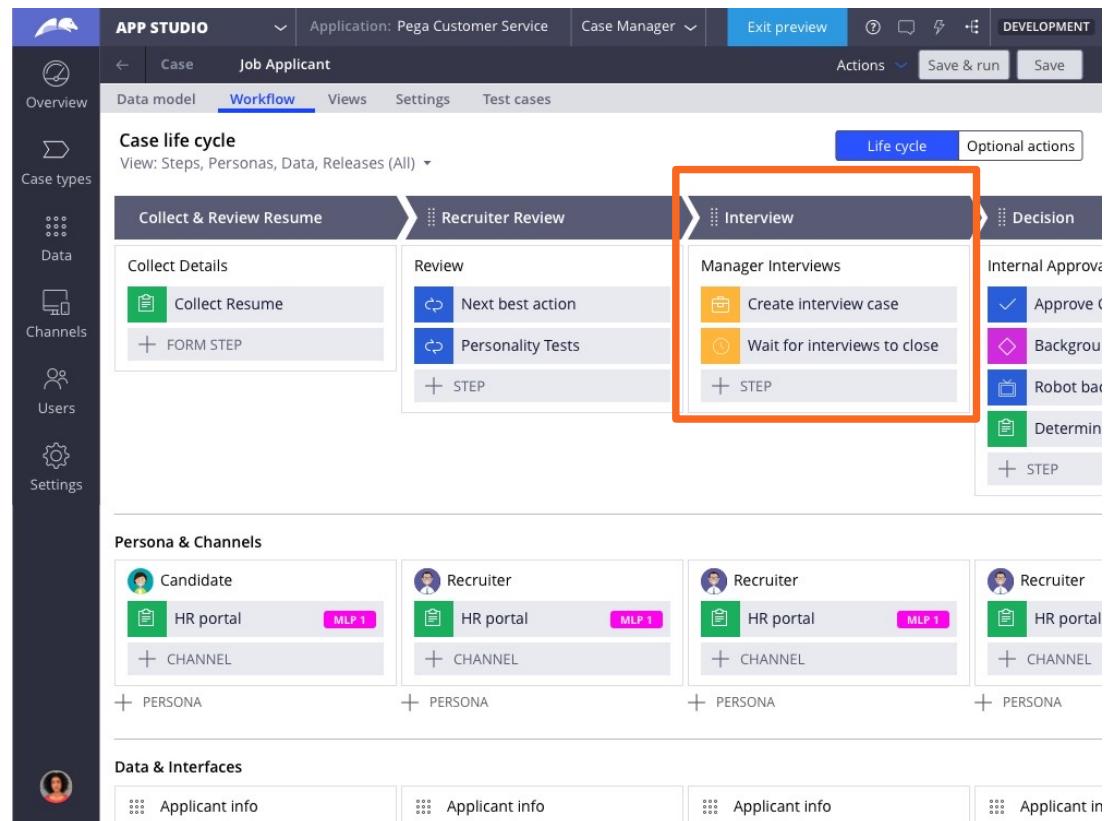
Overview

Complex business transactions may require the creation of multiple cases. Multiple cases allow for the organization of a business transaction and the ability to reuse case types in additional cases.



Use Case

- Implementing a business process in a separate child case type allows:
- Reuse of the case type as needed.
- Parallel processing between multiple cases.



Child case

Definition

Case relationships

- In the Pega Platform, case relationships are modeled with a case hierarchy that contains **top-level (parent) cases** and **child cases**.

Child case

- Represents work that must be completed to resolve the parent.
- A parent case that creates multiple child cases allows for work to be processed in parallel.
- When processing child cases in parallel, the parent case may need to wait until a child case is complete before the parent case can be resolved.
- Different parties with different expertise can handle each child case.

Top-level

- A case that does not have a parent case but can become a parent of other cases.

Child case benefits

Description

- **Child cases** are **beneficial** in situations for modeling work separately from the **Parent case** when several of the items in the list below are true:
 - Different data model is needed
 - Different life cycle is needed
 - Separate case ID and status for reporting is necessary
 - Separately assigning the case is needed
 - Must complete before the parent completes
- It would likely be appropriate to design a child case versus a subprocess when most of the items in the list exist.

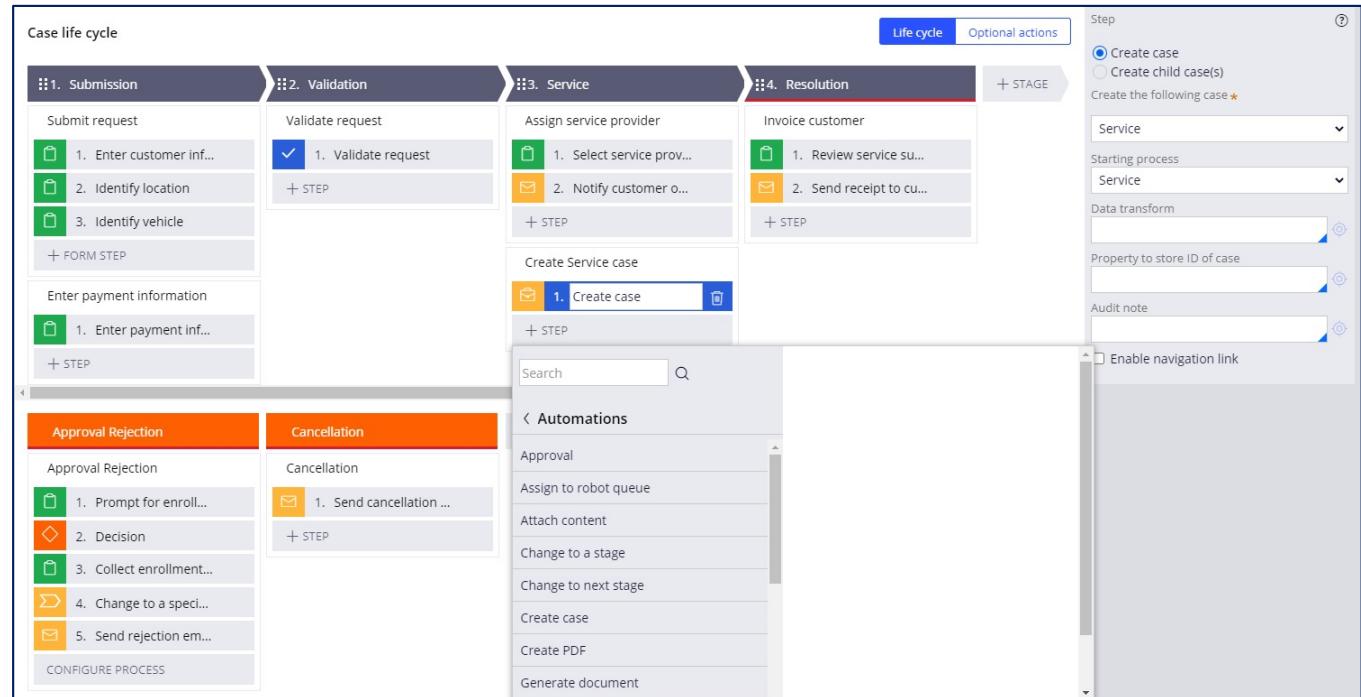


Add a create case step

Navigation

From App or Dev Studio:

1. Select the Case Type
2. Click on + Step
3. Then select More
4. Select Automations
5. Create Case



When encountering the Create Case step, the Pega Platform creates an instance of the specified case type.

Data propagation

Description

When creating a child case, you can also specify the information to copy from the parent case to the child case through a process known as *propagation*.

Data Propagation
in App Studio

- Identify the fields in the parent case **to transfer information to a new child case**.
- Option to “**transfer information to new case**”

Data Propagation
in Dev Studio

- Use a **Data Transform** to **copy the data values to fields identified** in a parent case to a child case.
- Option under the **Settings tab select Data Propagation**

Configure data propagation

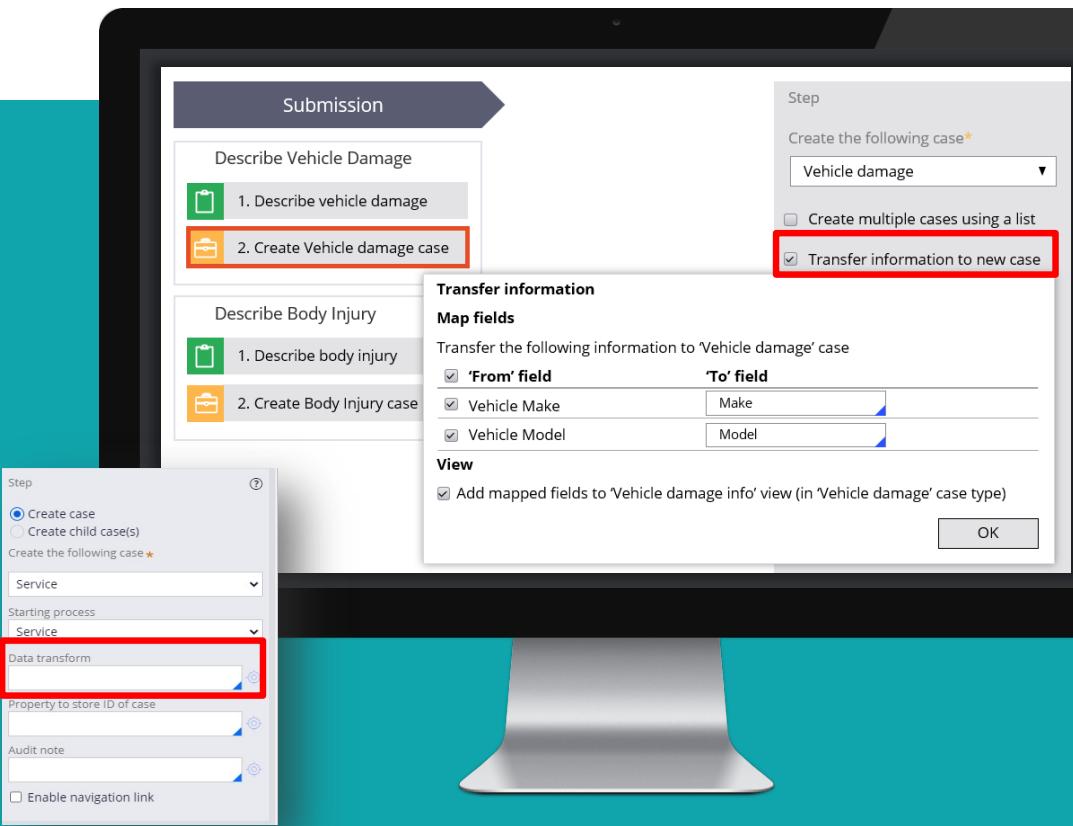
Navigation

From App Studio:

1. Select the Create case step
2. In the configuration pane, **enable the transfer information to new case** option

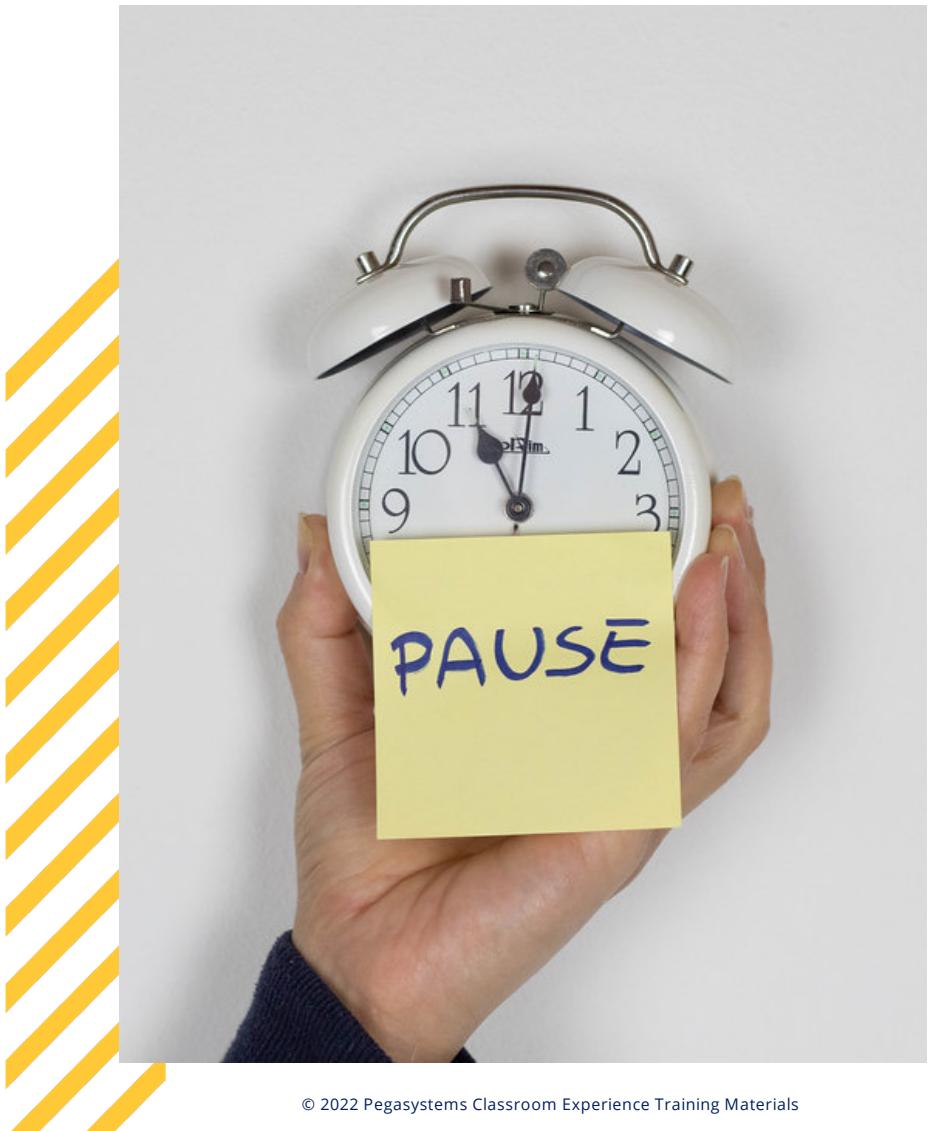
From Dev Studio:

1. Select the Create case step
2. In the configuration pane, enter the name of the Data transform



Use Case

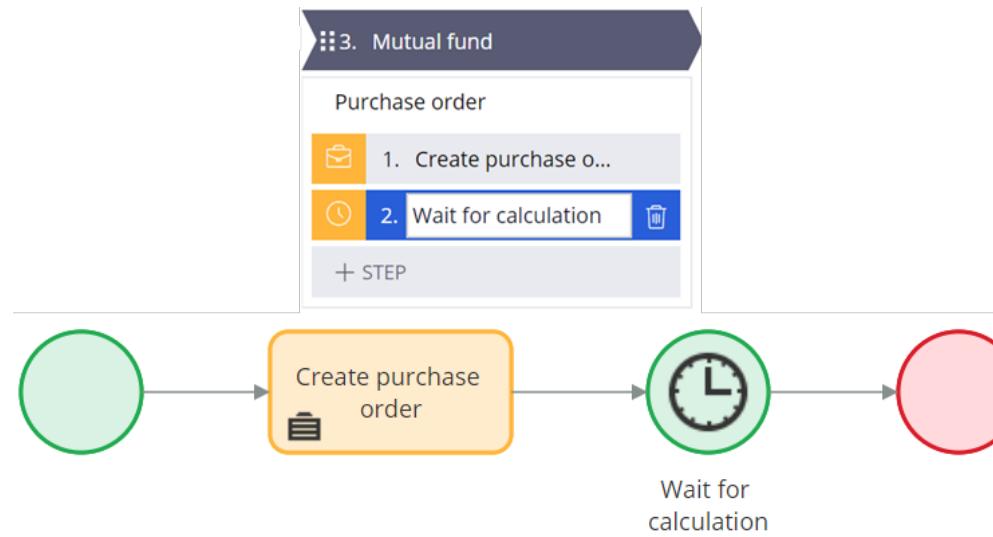
- A wait step can be used to:
- Pause a case during processing to provide time for offline processes.
- A child case dependency to be addressed before resuming processing.



Wait step

Definition

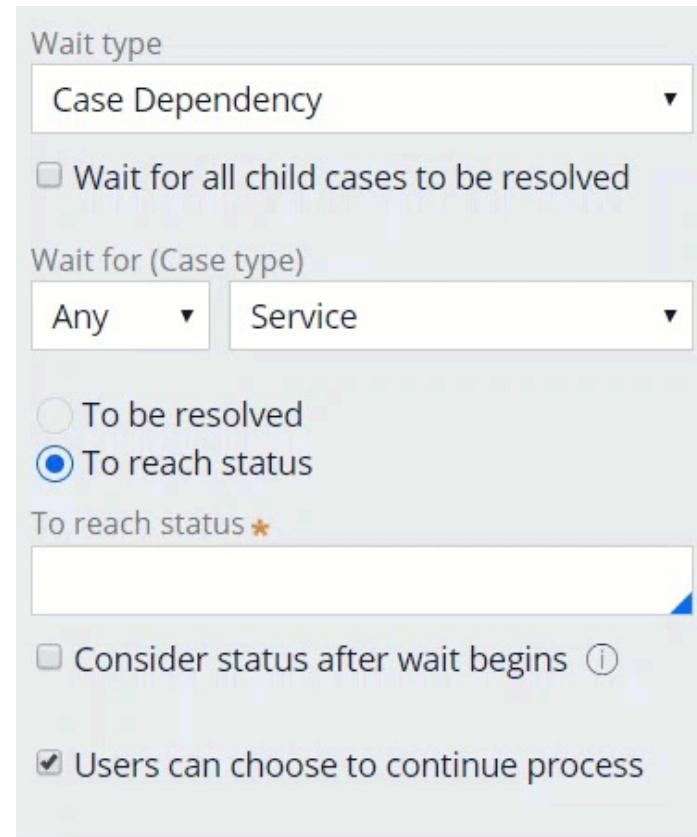
- Enforce dependencies using the Wait step.
- Allows pausing and resuming case processing when the case meets conditions that defined.
- The ability to pause processes can facilitate the resolution of complex cases that require other processes to end.



Case dependency

Definition and Description

- A relationship between a main business process (parent case) and supporting business processes (child case) that need to be resolved before a main process reaches resolution.
- Create case dependency by adding a Wait step to the parent case.
- The wait step pauses case processing of a parent case until the child case dependency resolves.



Wait type

Definition and Description

- The Wait step can be configured to pause case processing based on **Wait type**:
 - Case Dependency
 - When a parent case reaches the Wait step, the case pauses until all child cases, or any child case of a given type reach a designated status.
 - The status could be a standard status like Pending-approval or a custom status.
 - **To be resolved:** status is set to a value that starts with the word Resolved.
 - Timer
 - Pauses a case until the Set date/time interval expires or until a Reference date/time is reached

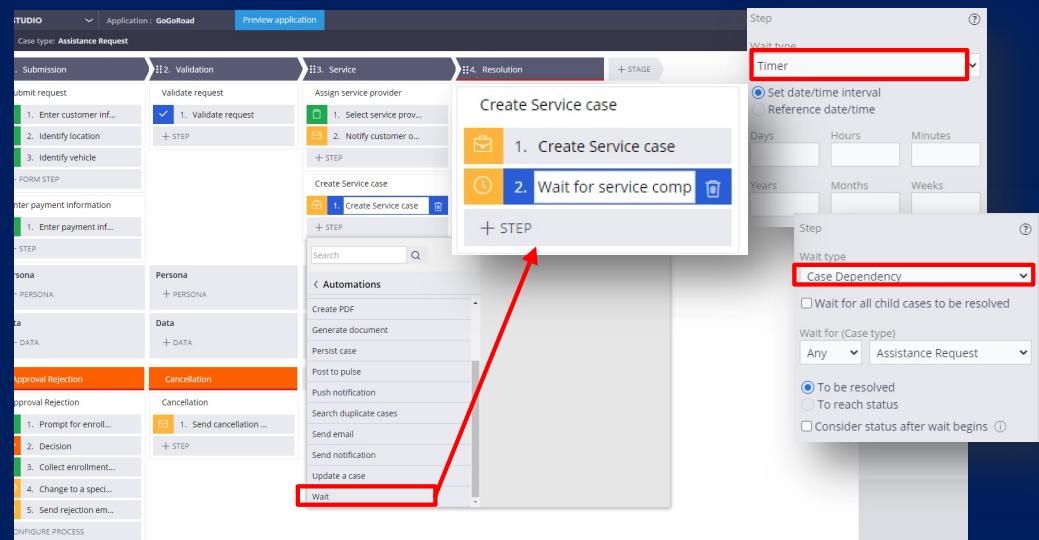


Add a Wait step

Navigation

From App or Dev Studio:

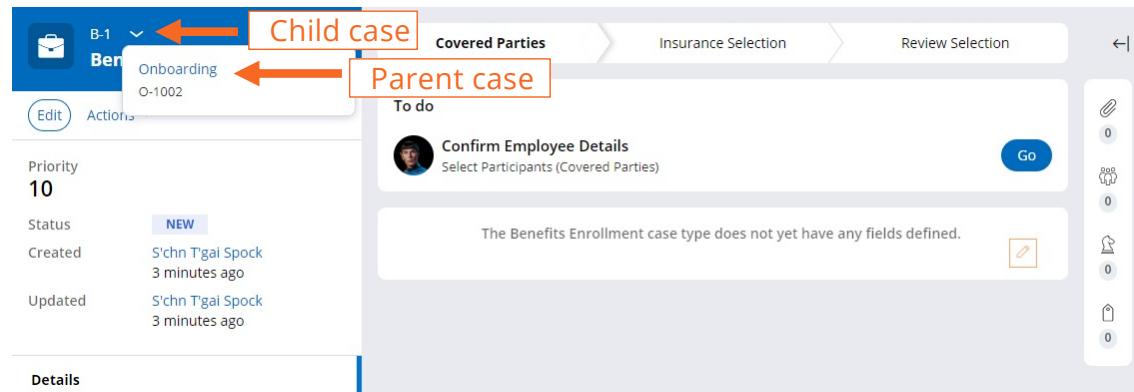
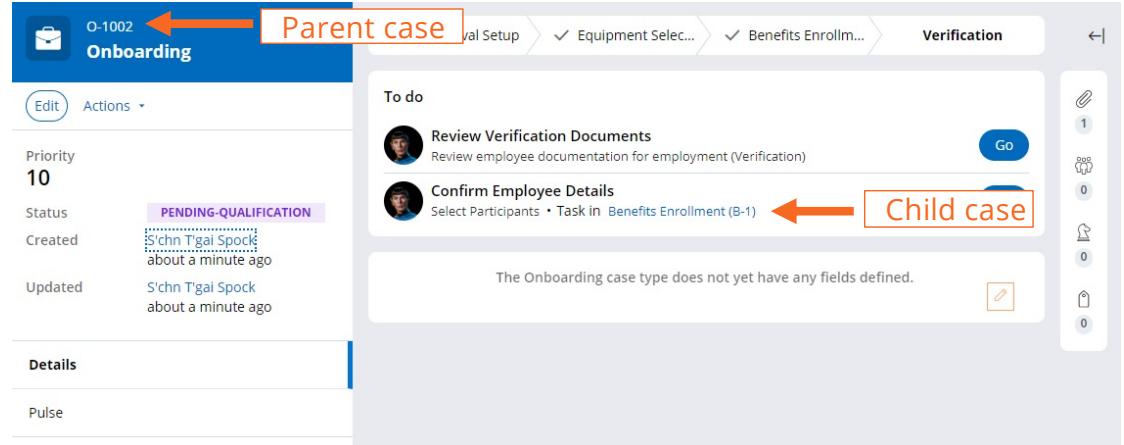
1. Select case type
2. Click + Step
3. More >Automations
4. Then select Wait
5. Select the Wait type in the configuration pane



Child case

Implementation

- Add a create case step or shape to a process.
- Configure Create case to create a separate case or a child case
- Run the Case Type to the point where the new case will be created
- Click on Actions > Refresh to view the open assignments which should display the child case
- Select the child case assignment which will display the parent case ID (link) above the child case ID



Wait type

Implementation

Dependency

- Add a wait step to the case life cycle
- Configure a case dependency between parent and child cases
- Run the case and check that the parent case waits for the child case to be Resolved or reaches a specified status

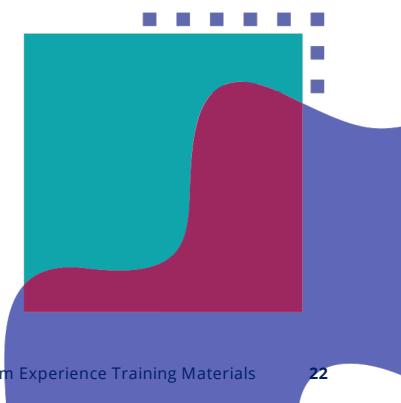
Timer

- Add a wait step to the case life cycle and configure a timer
- Run the case and check that wait is applied at the correct time in the process

The Wait shape puts the case in the deferred@pega work queue.

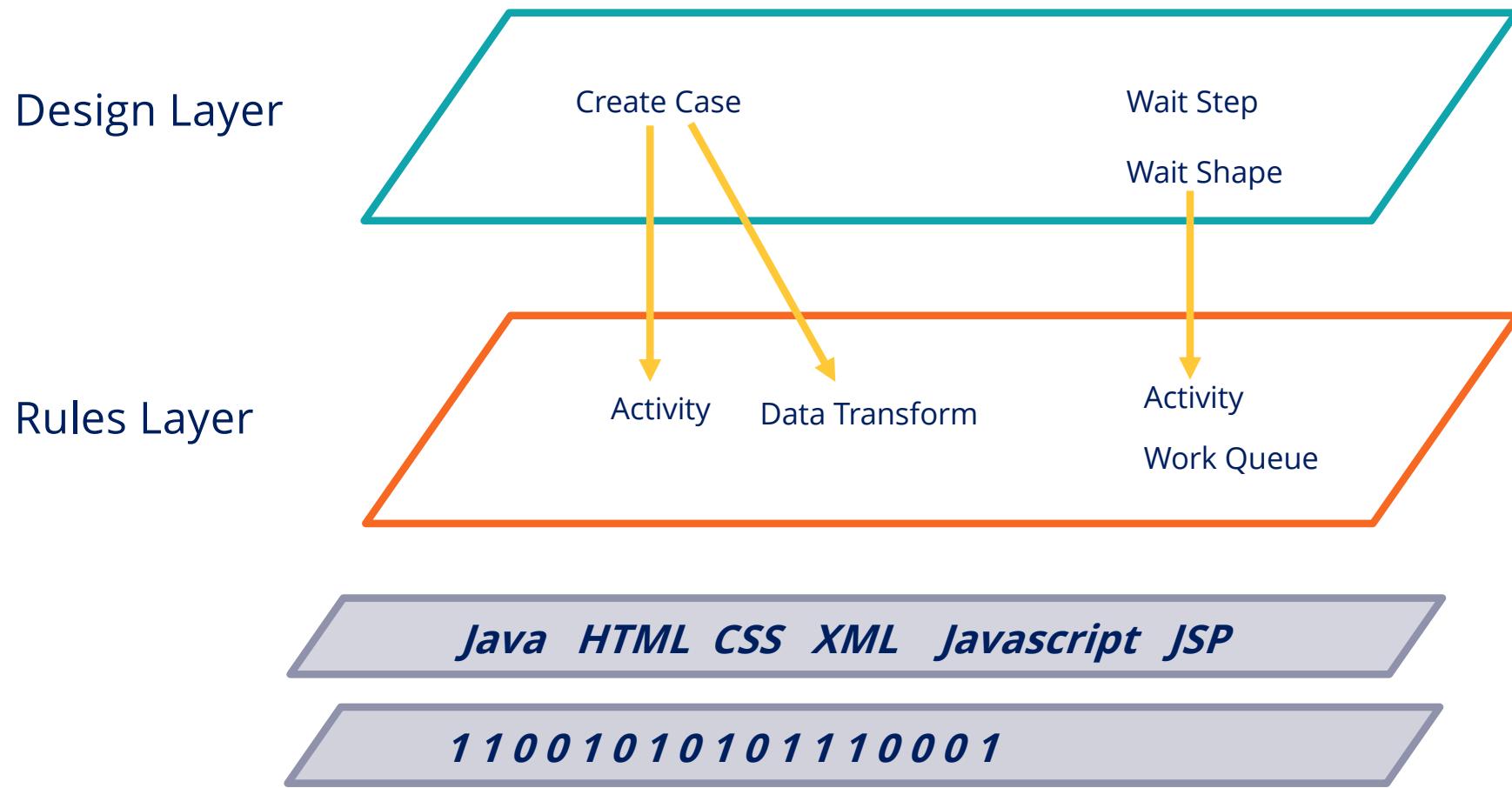
Best Practices

- Create case types that can be reused as child cases by multiple cases.
 - Aids in development speed and ease of maintenance.
- When creating child cases, use a wait step to enforce the parent-child dependency
- The child case cannot be a temporary case. The parent case refers to the child case by its case ID.
 - Temporary cases are not assigned a case ID, there is no unique identifier for the parent case to reference.
- The created case can either be a top-level case or child case. Dev Studio has additional configuration options for the Create Case step.
 - Multiple child cases can be created.
- A data transform can be used to propagate data from parent to child and back.





Schematic



Skill Mastery

Understand:

- top level cases
- child cases
- the Wait Step and Shape
- case dependency and timer



SKILL
LESSON

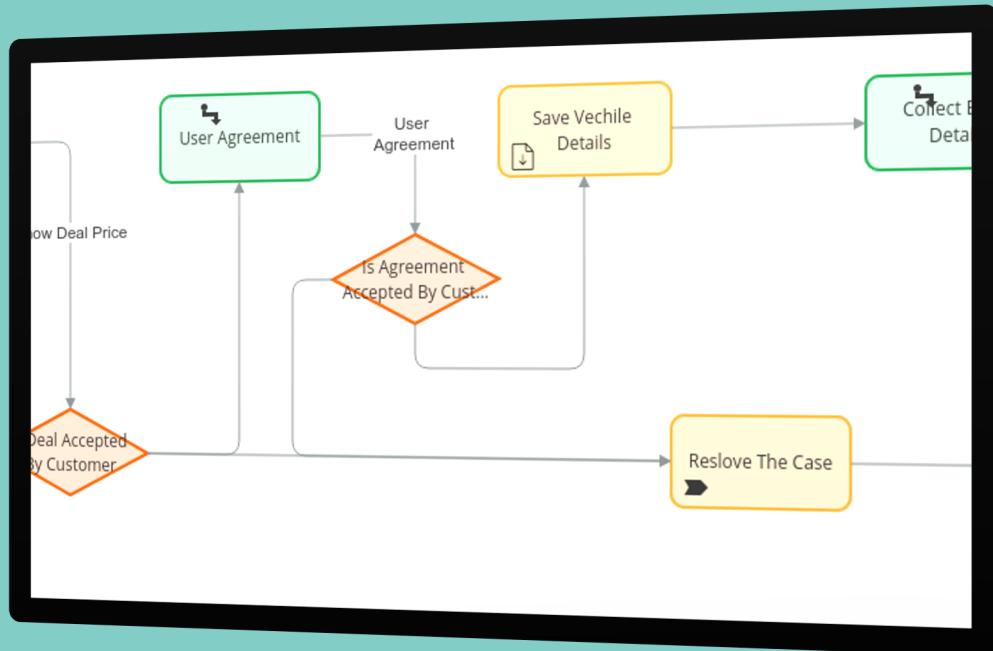
Process Design

The flow of your business activities



Overview

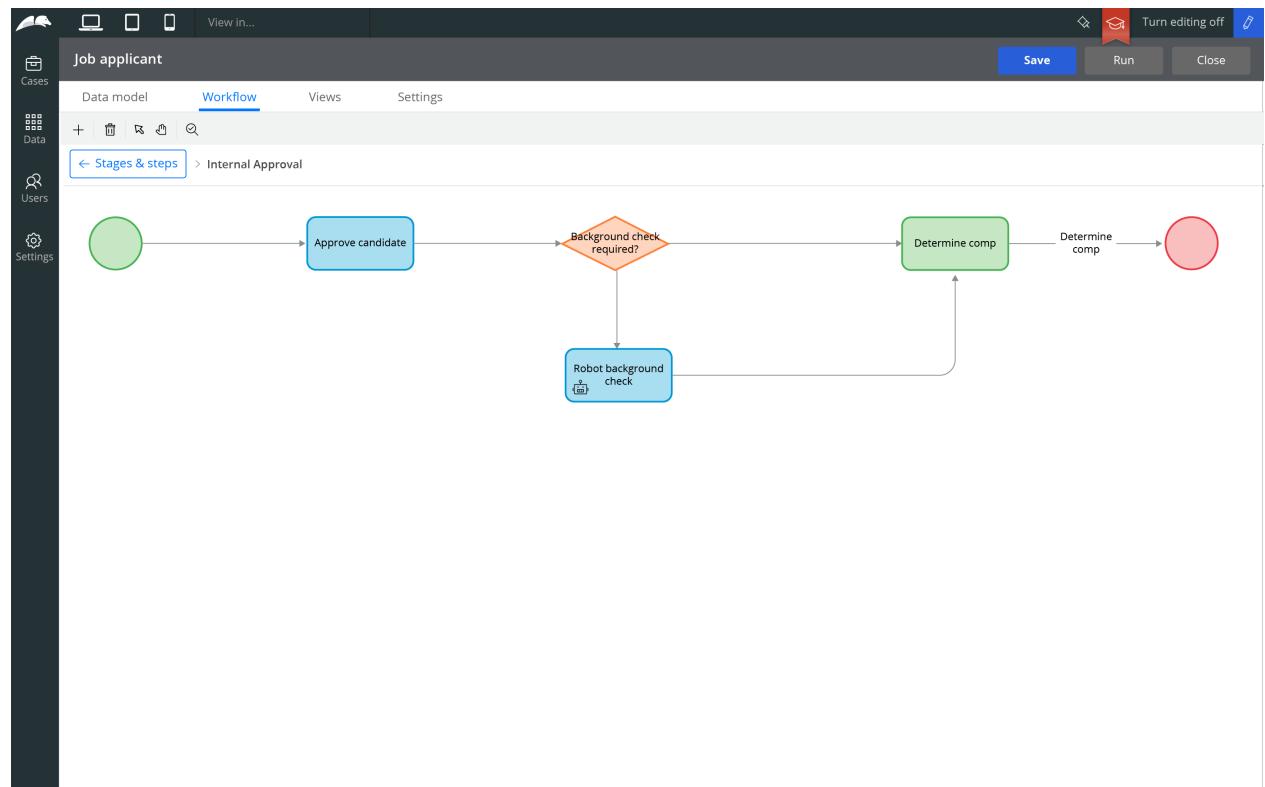
Case Design creates the model and helps with visualization and collaboration.



Conditional Processing

Use Case

- Create flows to support the different paths that users can take as they move a case through its lifecycle.
- By connecting different types of shapes in a flow, you can define the path that a case follows from creation to resolution.



Standard shapes available to the flow rule record

Definition



Represents the start in a flow. There is one Start per flow.

[Assignment]

Represents a pause in a flow. Signifies a person or system that a person or a system must act on a work object before the flow can progress.

[Subprocess]

Represents a reference to another flow rule. Also called as a *subflow*.

[Decision]

Represents a shape that references or calls a decision.

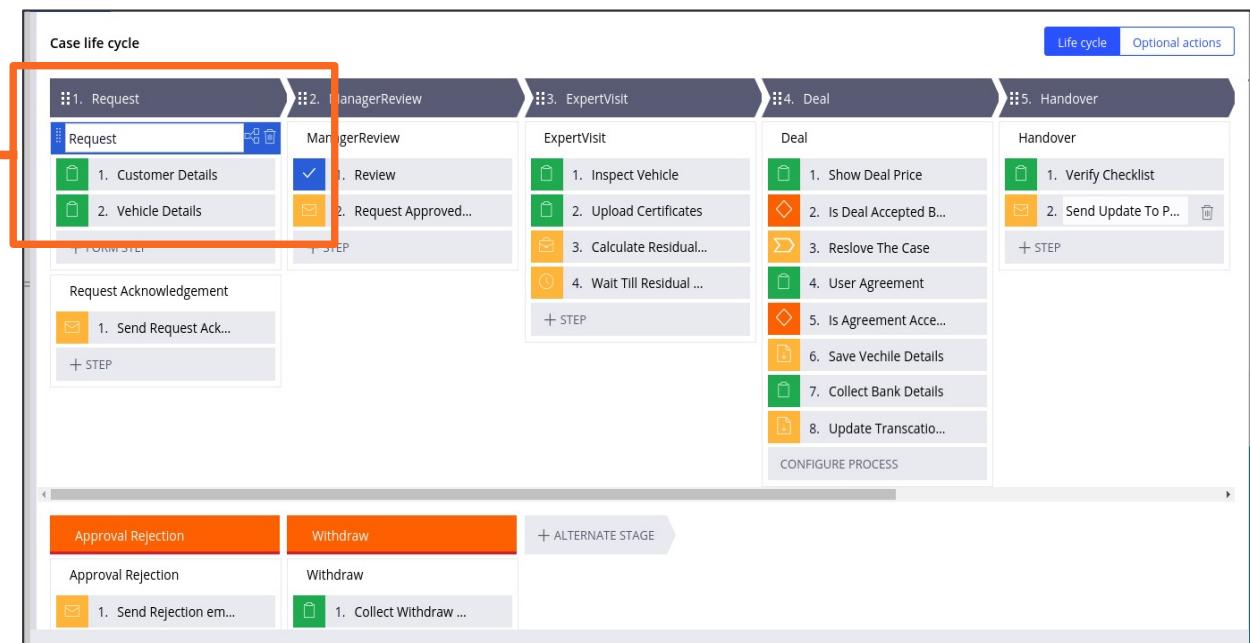
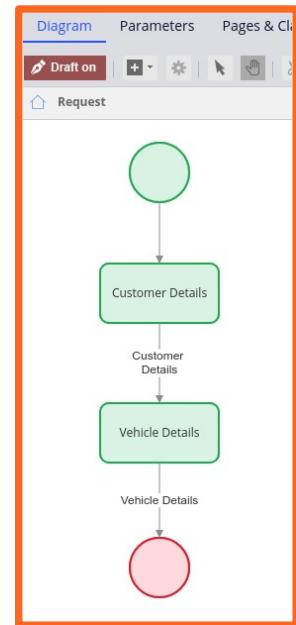


Represents an end point or termination in the flow. There can be more than one End Point per flow.

Process Design via Flow Rules

Description

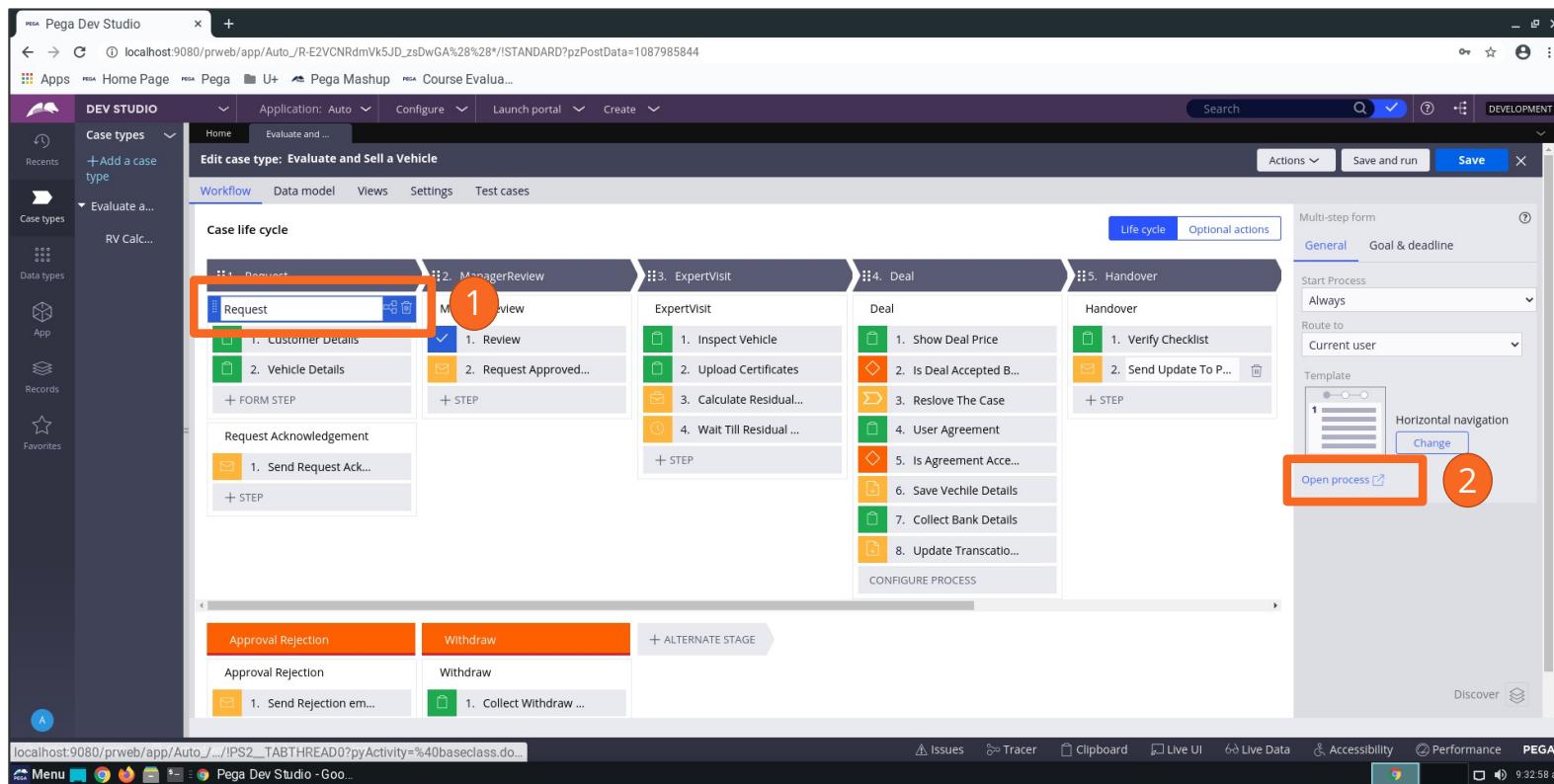
- Case designer processes contain steps.
- Each **process** in the case designer correlates to a **flow** record.
- When viewing the flow record, **process steps** are represented by **flow shapes**.
- The process flow steps/shapes represent the functions or tasks in the order they are to be performed.



App Studio: Accessing the Flow

Navigation

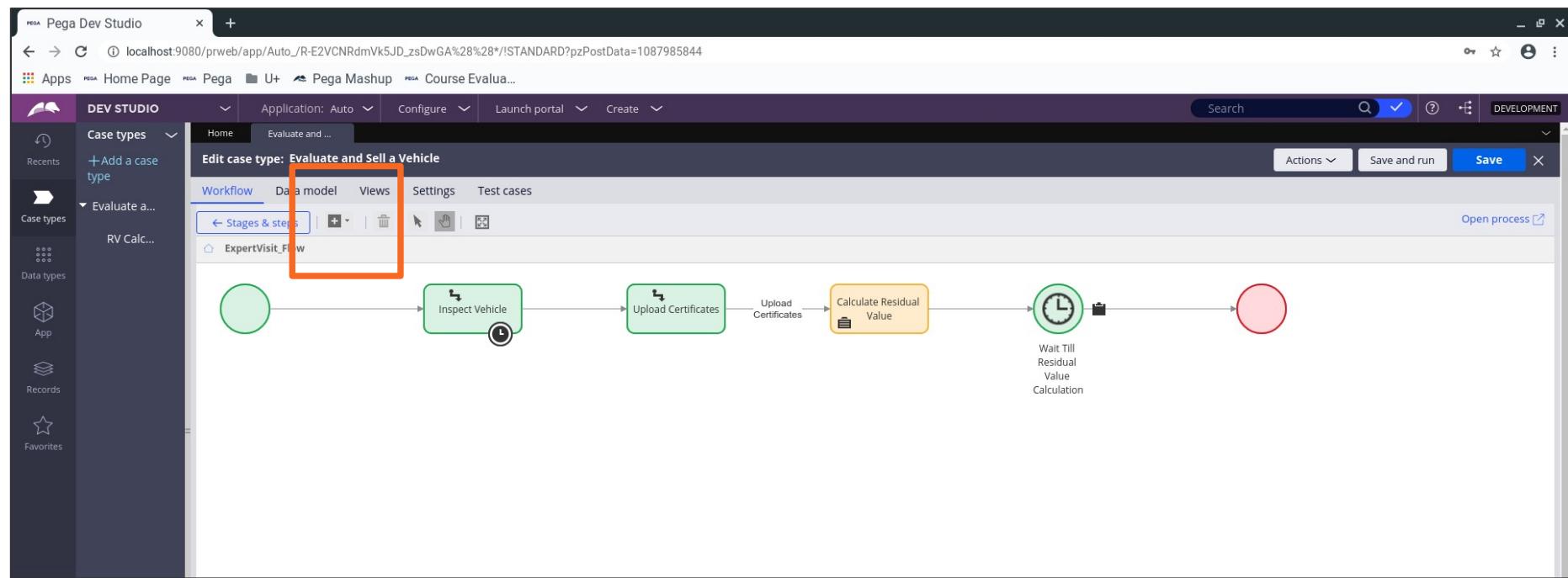
- Option 1 - Open Process modeler – high level process design with limited set of configurations
- Option 2 - Open Flow Rule – full configuration design capabilities available



Option 1 - Adding the Shapes to a Process Modeler

Implementation

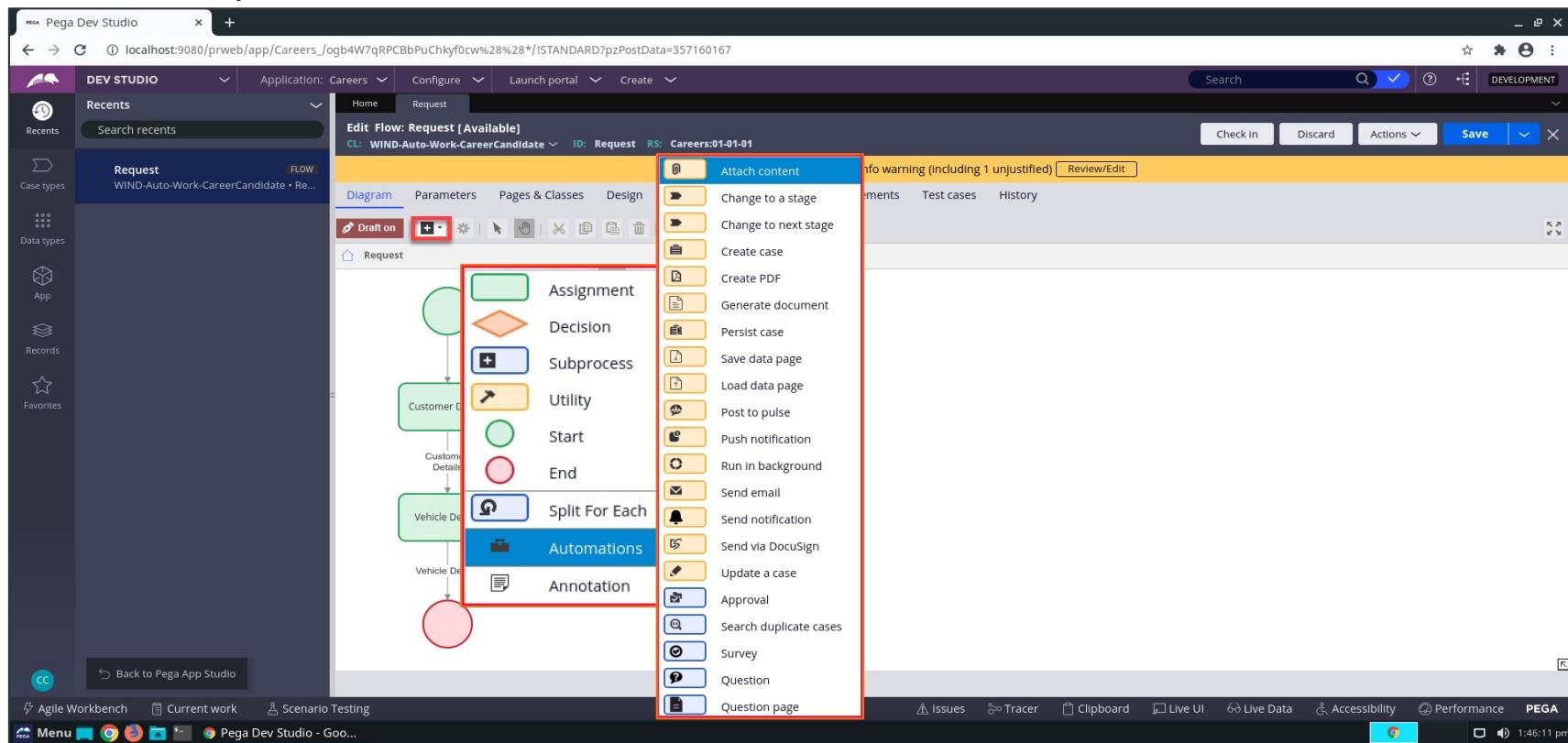
Click on the down arrow next to the plus sign  or right click in the Flow area to access the Select Shape menu. This is the Edit case Type wizard.



Option 2: Adding the Shapes to a Flow Rule

Implementation

- Click on the down arrow next to the plus sign  or right click in the Flow area to access the Select Shape menu.



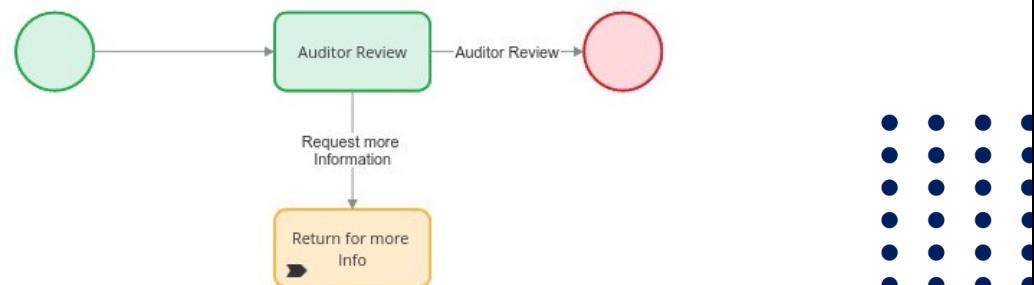
Assignment Shape-Decision or Human Based Decisions

Description

[Assignment]

Represents a pause in a flow. Signifies a person or system that a person or a system must act on a work object before the flow can progress.

- When a person has more than one choice on an assignment shape, it is a human based decision.
- Allows a user to make a choice and see different Views associated with that choice or path.



Add the assignment shape. Then add the connectors that represent the options needed. Name all the connectors.



Run Time

Implementation

- The user can select the different actions.
- Likelihood controls the default view and order of choices.
 - Highest likelihood is the default.
 - Higher likelihoods are in the order of highest to lowest.

The screenshot shows a Pega application interface. On the left, there is a card with the following details:

- F-1008
- Financial
- Priority: 10
- Status: Pending
- Created: [date]
- Updated: [date]
- Pulse: [status]

A context menu is open, with the "Actions" item expanded. The "Actions" menu contains the following items:

- View history
- Manage tags
- Manage notifications
- Request more information** (highlighted with a red box)
- Send Auditor Review** (highlighted with a red box)
- Refresh
- Pin to space
- Pin to recents

To the right of the card, a "Create" dialog is open. The dialog has the following fields:

Adjusted Amount	Cost Code	Amount
<input type="text"/>	<input type="text"/>	<input type="text"/>

Below these fields are two additional fields:

Expected Amount	Accepted
<input type="text"/>	<input checked="" type="checkbox"/> Accepted

On the right side of the dialog, there is a "Notes" area with a large text input field.

At the bottom of the dialog are "Cancel" and "Create" buttons.

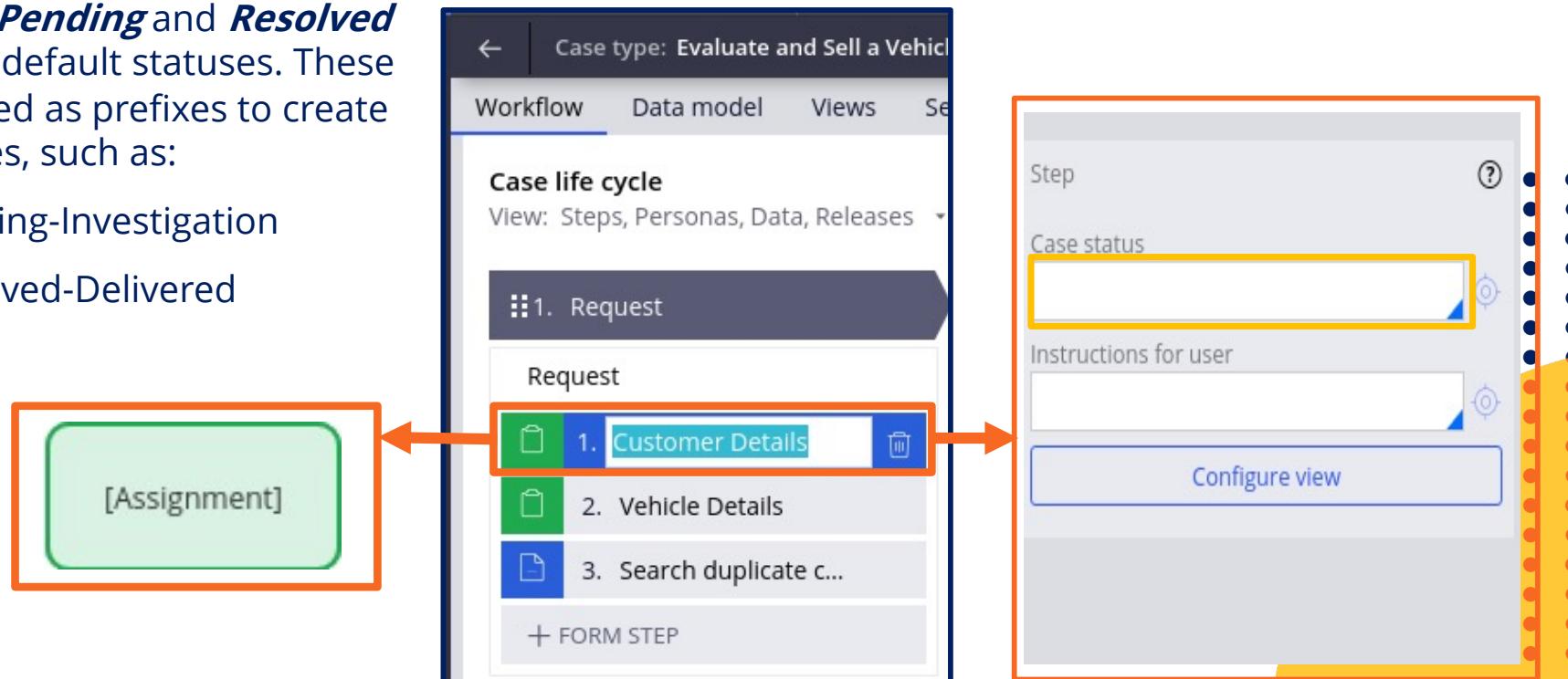
Case Status and Instructions

Definition

Status is a verbal representation of where the work is at in the life cycle. Instructions describe the task to be performed.

New, Open, Pending and **Resolved** are available default statuses. These can be used as prefixes to create custom values, such as:

- Pending-Investigation
- Resolved-Delivered



Decision shape to automate flow control

Definition

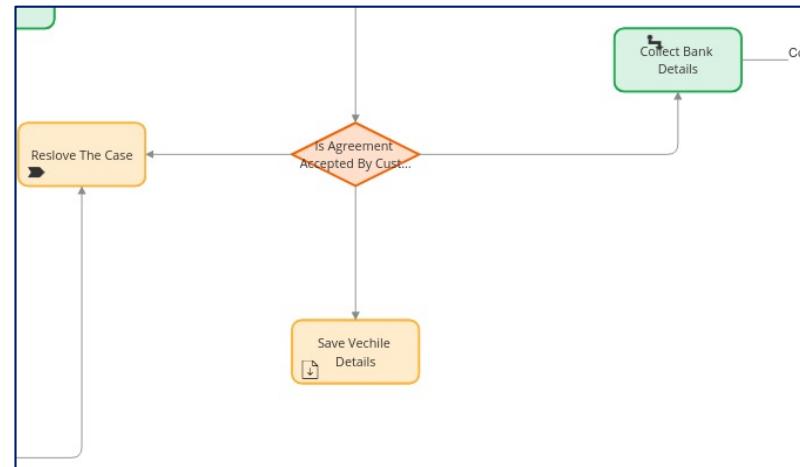


The decision shape represents a gateway branching the process down a different path depending on the incoming data or end user choices.

Process Modeler view:



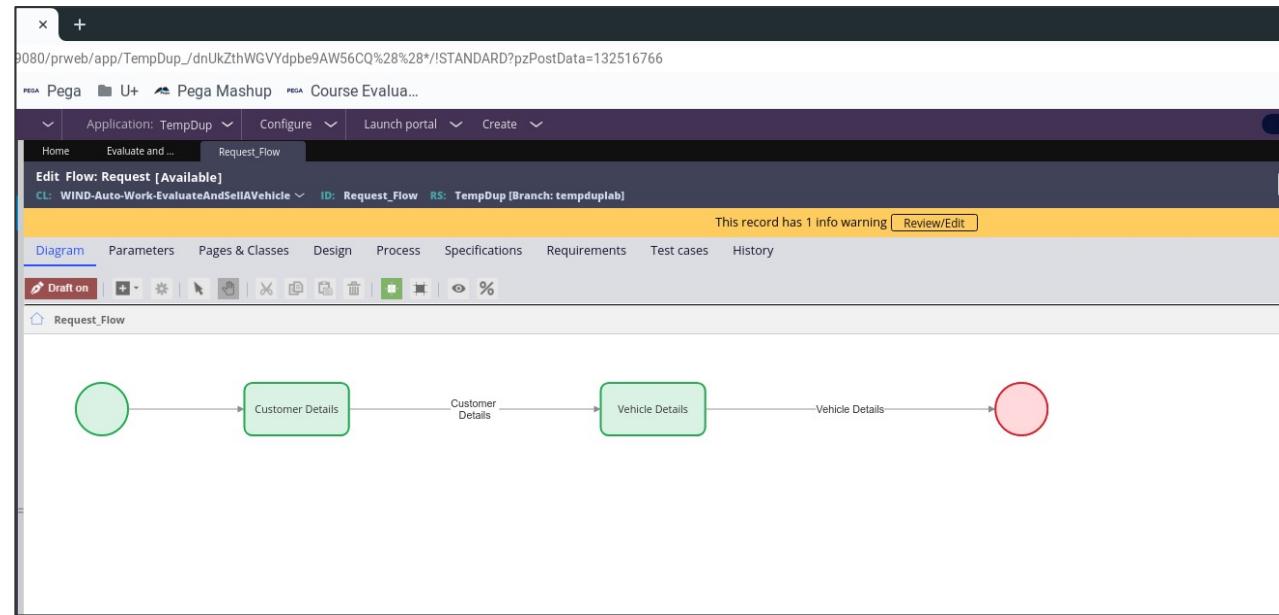
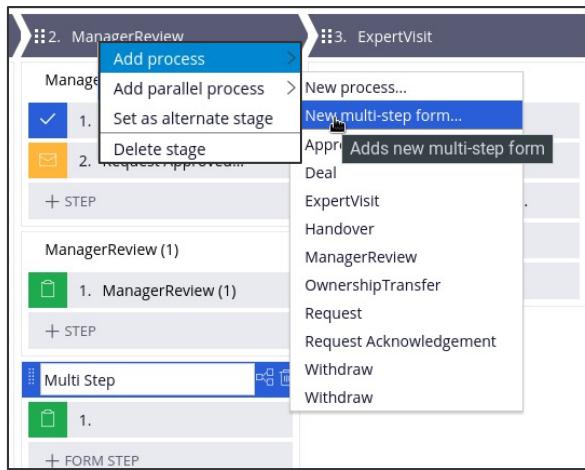
Flow rule record view:



Multi step Form

Definition

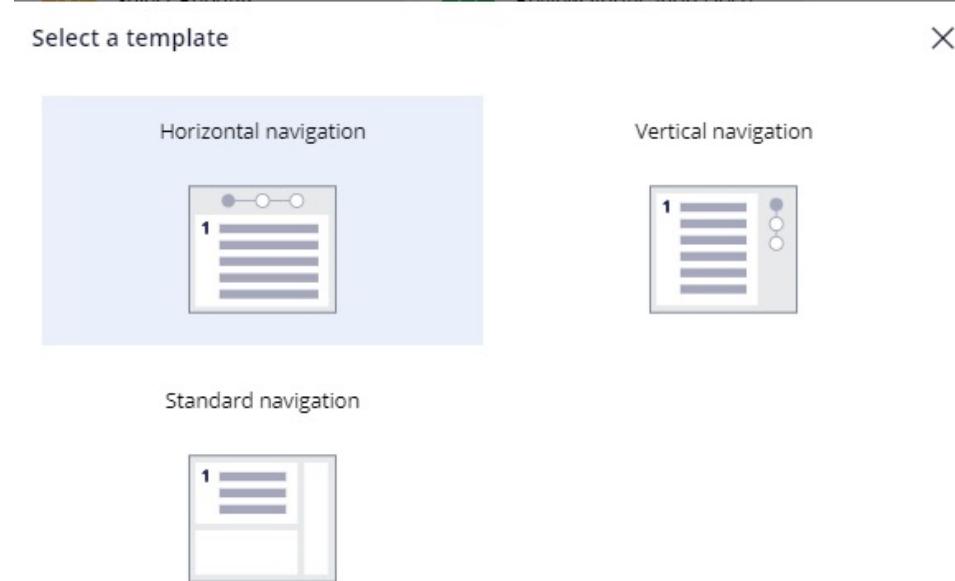
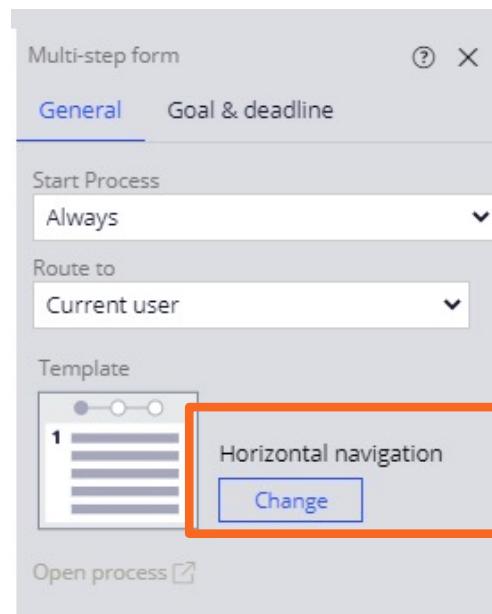
- Represents the concept of an interactive form.
- A user can navigate through several screens entering information. The user sees *Back* and *Forward* buttons as well as navigation links allowing the user to move from the current page back through its ancestors.



Multi Step Form: End User View

Description

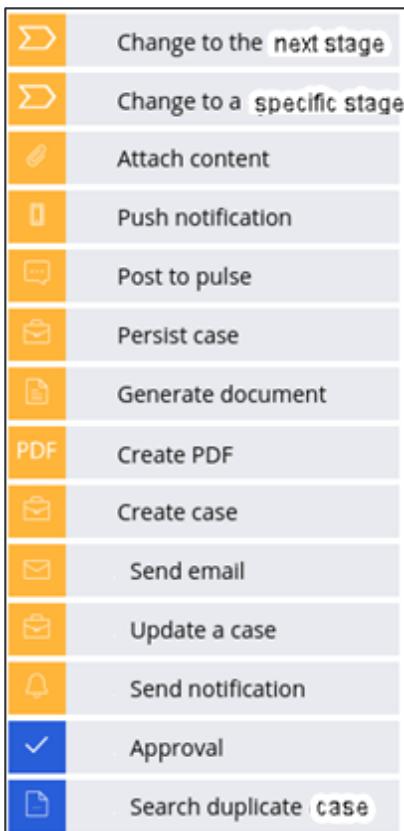
The Multi Step form allows navigation backward and forward with buttons. Also, the form can be presented as horizontal, vertical, or standard.



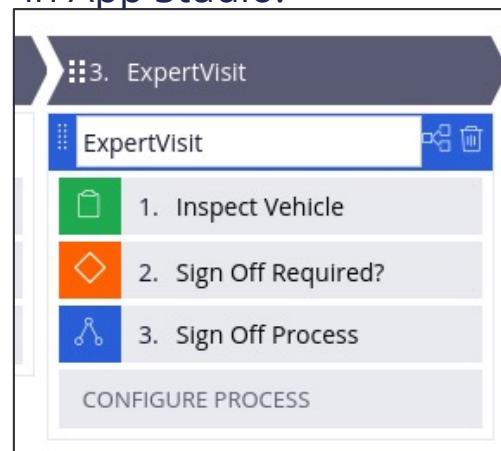
The screenflow navigation option is configured as a harness on the start shape in the flow.

Steps available in the Workflow Tab of Case Designer

Description

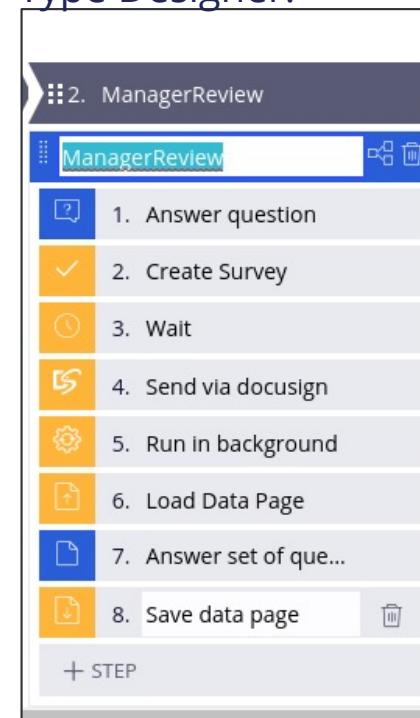


This is the view of the Steps available in Case Designer (Workflow Tab) in App Studio.



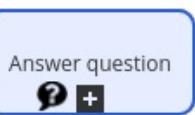
This is the view of some Case Designer steps after you have added *Assignment*, *Decision*, and *Subprocess* shapes via the Process Modeler.

This is the view of the Automation Shapes from the Workflow Tab in the Case Type Designer.



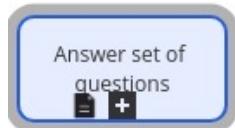
Automation Shapes available while using Dev Studio

Description



Answer question

Represents the ability of the system to collect a single response from a user who is processing a case. Isolating individual questions, one can selectively capture input without the formal structure of a survey.

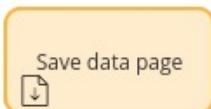


Answer set of
questions

Represents the ability of the system to collect information from a user who is processing a case. Provides a structured format for questions and answers incorporating user input into the case.

Automation Shapes available when using Dev Studio*

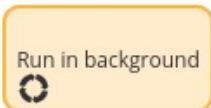
Description



Represents the ability of the system to save a data page. A data page is an object of information in memory.



Represents the ability of the system to preload load a data page into memory lessening the time needed for a screen to display.



Represents the ability of the system to start a background process, such as asynchronous processing or queuing of items.



Represents the ability of the system to collect digital signatures for case attachments.



Represents the ability of the system to create a survey.



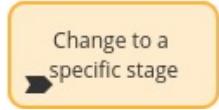
Represents the system waiting for a subcase to reach a specific status or time period to expire.

Automation Shapes available when in the flow

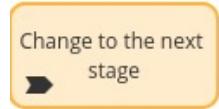
Description



Represents the ability of the system to attach content to the case.



Represents the ability of the system to navigate the case to a specific stage.



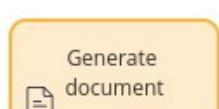
Represents the ability of the system to navigate the case to the next stage.



Represents the ability of the system to call and create another case.



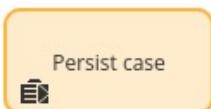
Represents the ability of the system to create a PDF file.



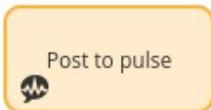
Represents the ability of the system to generate a Word document.

Automation Shapes available when in the flow

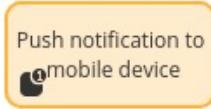
Description



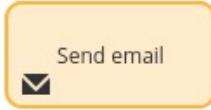
Represents the ability of the system to save the case.



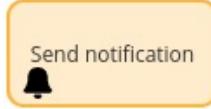
Represents the ability of the system to post a pulse message.



Represents the ability of the system to send a notification to a mobile device.



Represents the ability of the system to send an email.



Represents the ability of the system to send notification via a configured channel.



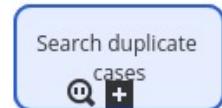
Represents the ability of the system to update another case/s running in parallel.

Other Shapes available when in the flow

Definition



Represents the ability of the system to call the out-of-box Approval Subprocess.



Represents the ability of the system to search for duplicate cases.



*Annotation is for information to be added for contextual purposes.

*Annotation is not considered a smart shape



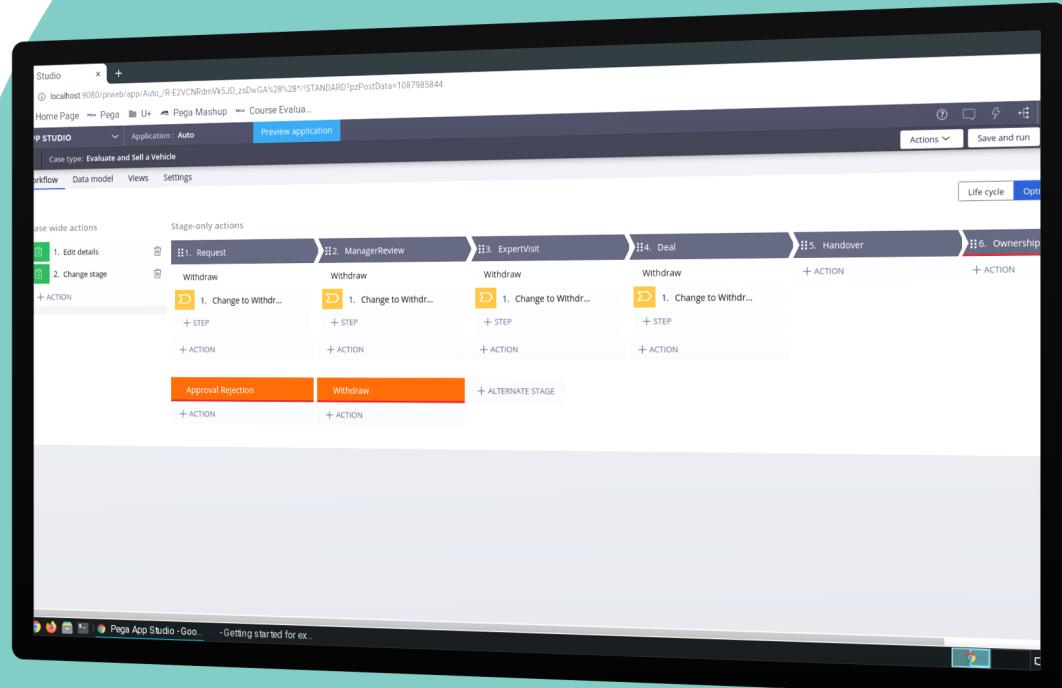
SKILL
LESSON

Optional Actions



Overview

Optional Actions allow users to make choices during the case life cycle and modify a predefined flow.



Use Case

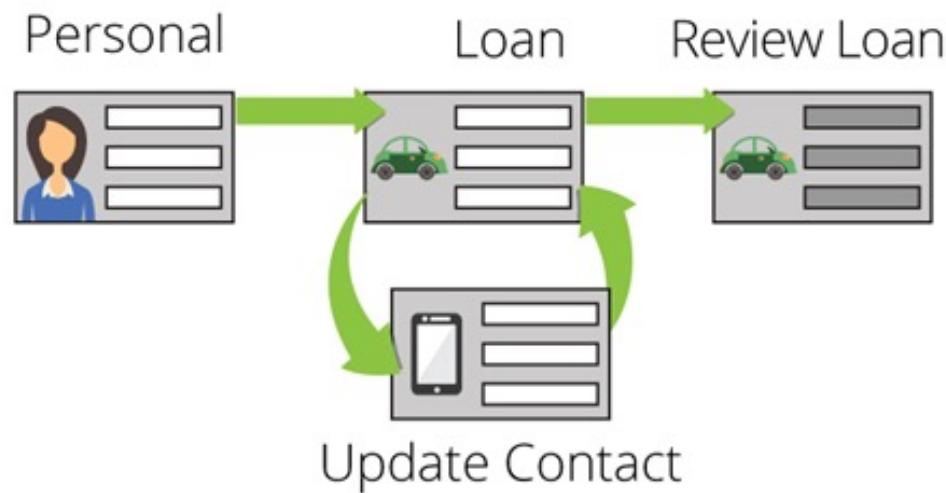
- Use Optional Actions to allow an end user to start a new process or call a screen to interact with that is outside of the predefined case life cycle sequence.
- For example, while reporting a car accident to an insurance company, the customer mentions having a new phone number. The customer representative uses an optional action to update the customer's contact information.



Optional Actions are Flow Actions or processes a User chooses out of sequence

Description

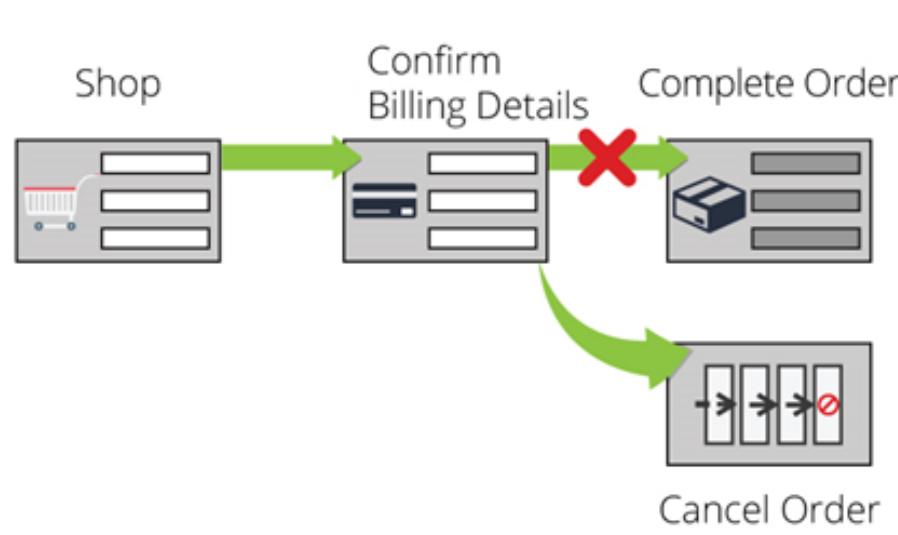
- Optional Actions allow a user to select either a Local Action or an entire process containing flow actions and local actions to be ran.
- Optional Actions as a Local Action do not move a case anywhere. It simply shows UI with the expectation the end user will interact with the form.



Optional Actions are Flow Actions or processes a User Chooses out of sequence

Description

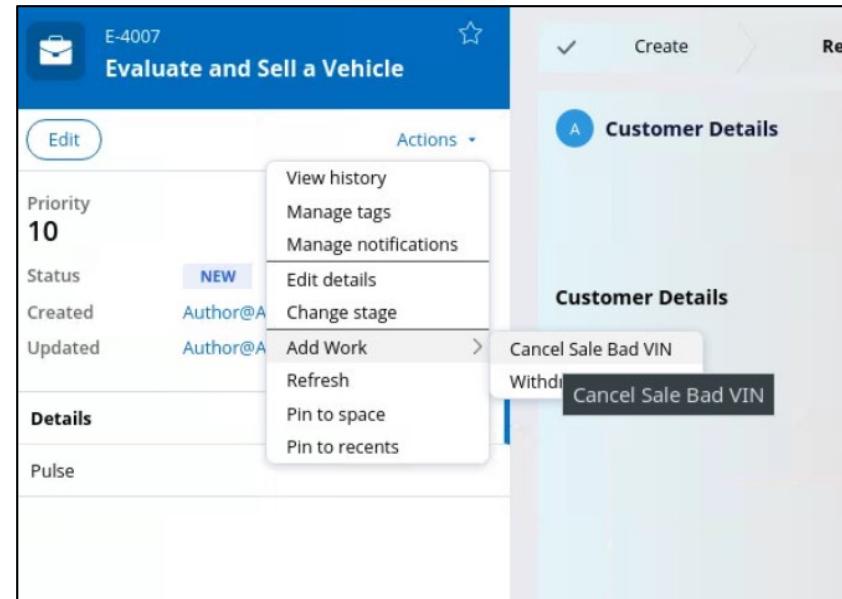
Although starting an optional process does not interrupt the current processes running in a case, it is often used to start a process that ultimately results in early termination of the case, such as a process to cancel an order.



What are Optional Actions?

Definition

- Users can perform Optional Actions while a case is in any stage or step of the life cycle.
- By allowing users to choose when additional processing is needed, you can support out-of-sequence events in a case.
- You can configure two types of optional actions:
 - As a user action
 - As a process



User Interaction with Optional Action

Description

- An End User interacts with an Optional Action via the Actions Menu found in the left-hand corner of each view.
- Actions are listed first, and Processes are listed under *add work*.
- Availability of Optional Actions and Processes will be seen by the end user for the entire case when defined as a Case wide action or process or only available for the Stage when defined under that stage.

The screenshot displays the Pega Case Management interface for the case type "Evaluate and Sell a Vehicle".

Left Panel (Workflow View):

- Case wide actions:** 1. Edit details, 2. Change stage, + ACTION.
- Stage-only actions:** 1. Request, 2. ManagerReview, 3. ExpertVisit, each with sub-options like Withdraw, Change to Withdraw, + STEP, + ACTION.
- Bottom:** Approval Rejection, Withdraw, + ACTION, + ALTERNATE STAGE.

Right Panel (Case Detail View):

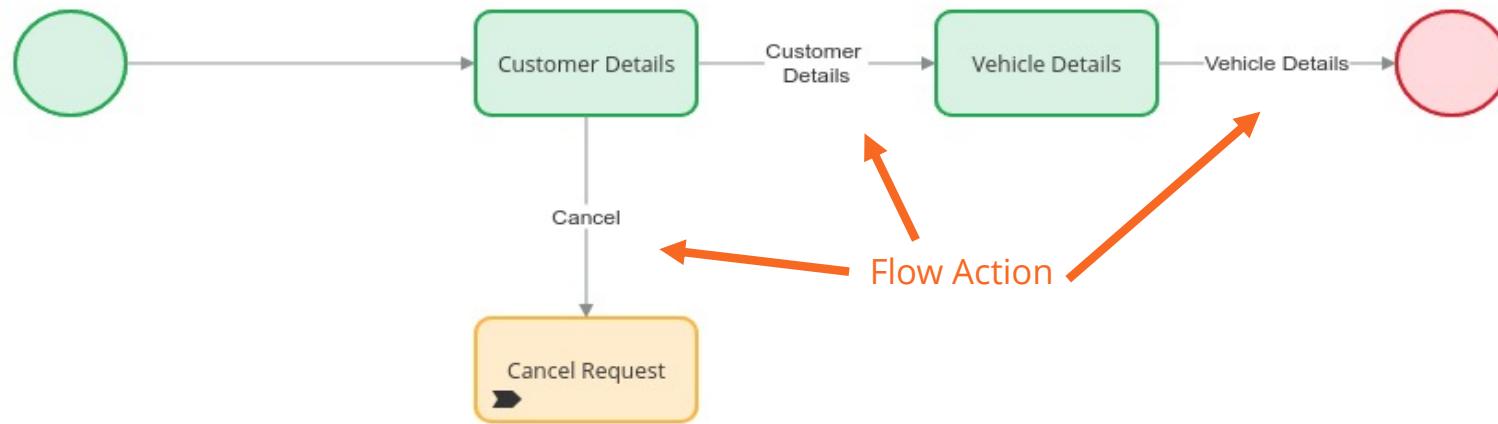
- Case Details:** E-4007, Evaluate and Sell a Vehicle, Priority 10, Status NEW, Author@A, Created, Updated.
- Actions Menu (opened from the top right):** View history, Manage tags, Manage notifications, Edit details, Change stage, Add Work (with options: Refresh, Pin to space, Pin to recents), Cancel Sale Bad VIN, Withdraw, Cancel Sale Bad VIN.
- Customer Details:** A section showing customer information with a "Customer Details" heading.



Flow Actions are Connectors

Description

- Flow Actions are the connectors between the shapes in the process flow.
 - They define the path the user takes when submit is clicked.
- End Users have a choice of paths when more than one connector emanates from a shape.
- Flow Action Connectors move a case to another shape acting as placeholders for the UI a user interacts with.
- Flow Actions can be localized to a specific place in the flow.
 - A local Flow Action does not move the case anywhere.
 - It displays the UI (view) for a user to interact with.





Setting a Flow Action to be a Connector, Local or Both

Implementation

1. Open the flow action rule.
2. Review the rule type.
3. Configure the Indicator.

The screenshot shows the Pega Dev Studio interface with the following details:

- Connector properties** dialog (left):
 - Connector: Show Deal Price
 - Connector section:
 - What is it doing? (dropdown menu, step 1)
 - Flow action*: ShowDealPrice
 - Likelihood*: 100
 - Audit note: [empty]
 - Set properties button
- DEV STUDIO** main window (center):
 - Application: Auto
 - Edit: Flow Action: Show Deal Price [Available] (step 2)
 - Flow tab selected in the sidebar.
 - Indicator section:
 - Used as: Local and Connector (selected, step 3)
 - Disqualify this action from bulk processing
- Bottom navigation and status bar**: Agile Workbench, Current work, Scenario Testing, Menu, Issues, Tracer, Clipboard, Live UI, Live Data, Accessibility, Performance, PEGA, © 2022 Pegasystems Classroom Experience Training Materials, 11:31:08 am, 10.

Optional Action and Processes

Description

- Optional Action can be defined for a Stage (e.g. Withdraw) or Case wide
- Availability of Optional Actions and Processes will be seen by the end user for the entire case when defined as a Case wide action or process or only available for the Stage when defined under that stage.

The screenshot displays the Pega Case Management interface for a case type named "Evaluate and Sell a Vehicle".

Workflow View: Shows the "Life cycle" and "Optional actions" tabs. The "Optional actions" tab is selected, displaying three stages: "1. Request", "2. ManagerReview", and "3. ExpertVisit". Each stage has an associated "Withdraw" action and an "Approval Rejection" action. The "ManagerReview" stage also has a "Change to Withdraw" step.

Case Details View: Shows the case details for E-4007. It includes fields for Priority (10), Status (NEW), Created (Author@A), Updated (Author@A), and Details. A context menu is open over the "Actions" button, listing options like "View history", "Manage tags", "Manage notifications", "Edit details", "Change stage", "Add Work", "Refresh", "Pin to space", and "Pin to recents".

Customer Details View: Shows a list of customer details. One item, "Customer Details", has a context menu open with options "Cancel Sale Bad VIN" and "Withdraw".

Pega Logo: Located at the bottom left of the interface.

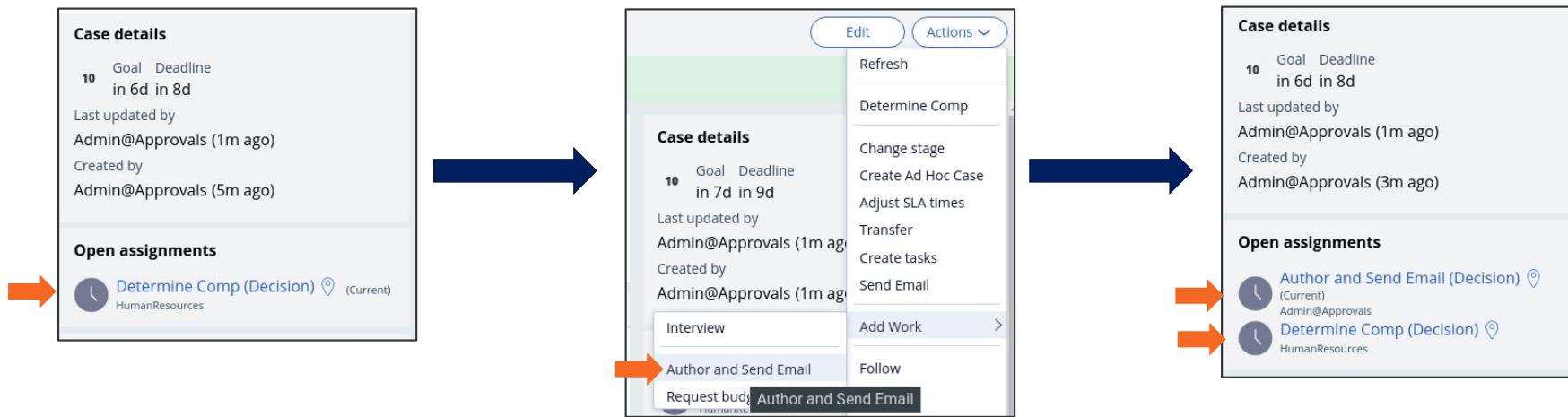
Page Footer: Includes the copyright notice "© 2022 Pegasystems Classroom Experience Training Materials" and the page number "11".



Optional Actions and Process Flows

Implementation

- Optional Actions as a Process calls a process (Flow).
- When a flow runs via the optional process feature, it is started in parallel with other flows running in the case.
- All Optional Process will appear under the Add Work submenu of the Actions menu.



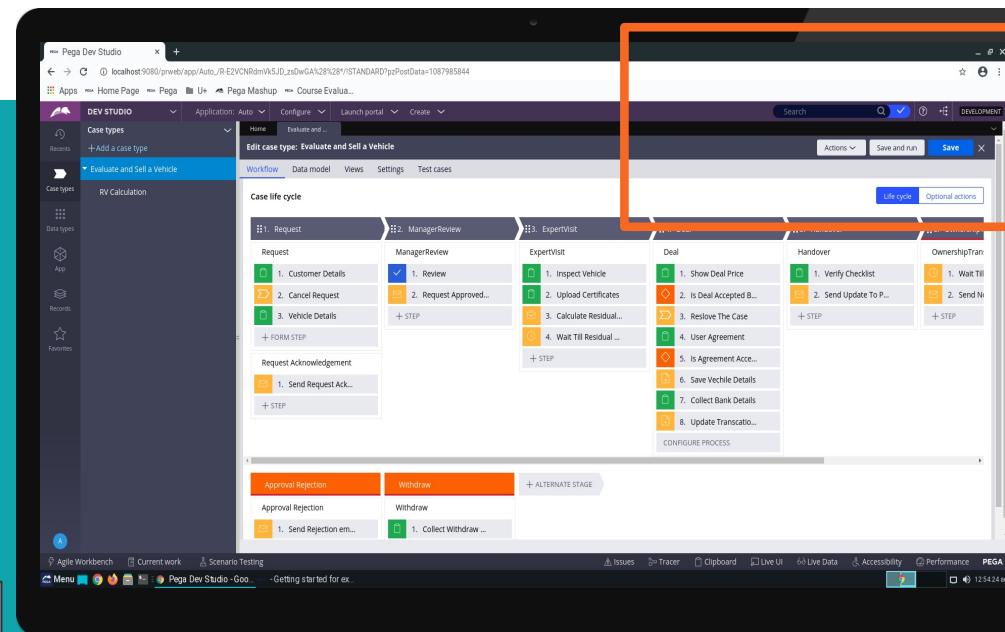
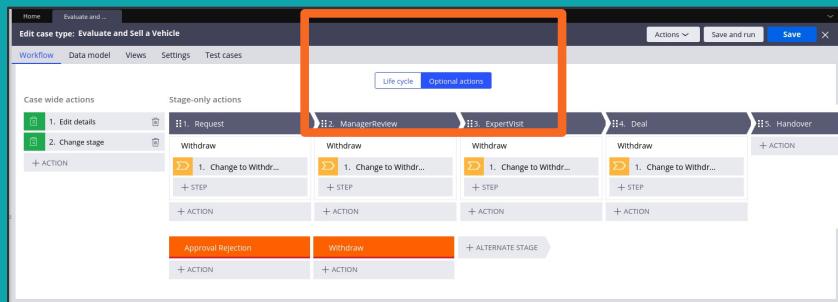
Optional Actions Tab

Navigation

The **Workflow Tab** in the Case Designer consists of two Tabs:

- Case Lifecycle (default)
- Optional Actions

Availability of Optional Actions and Processes is controlled by placing the action or process under Case wide actions or a specific stage.





Schematic

Design Layer

Case Designer > Optional Actions

Rules Layer

Case Type

Flow

Flow Action

Java HTML CSS XML Javascript JSP

11001010101110001

Skill Mastery

Understand:

- Flow Actions
- The difference between Local and Connector Flow Actions
- Optional Actions and Processes
- The relationship between the Case Type diagram and Flow Rules
- The relationship between the Flow Rules and the Flow Actions

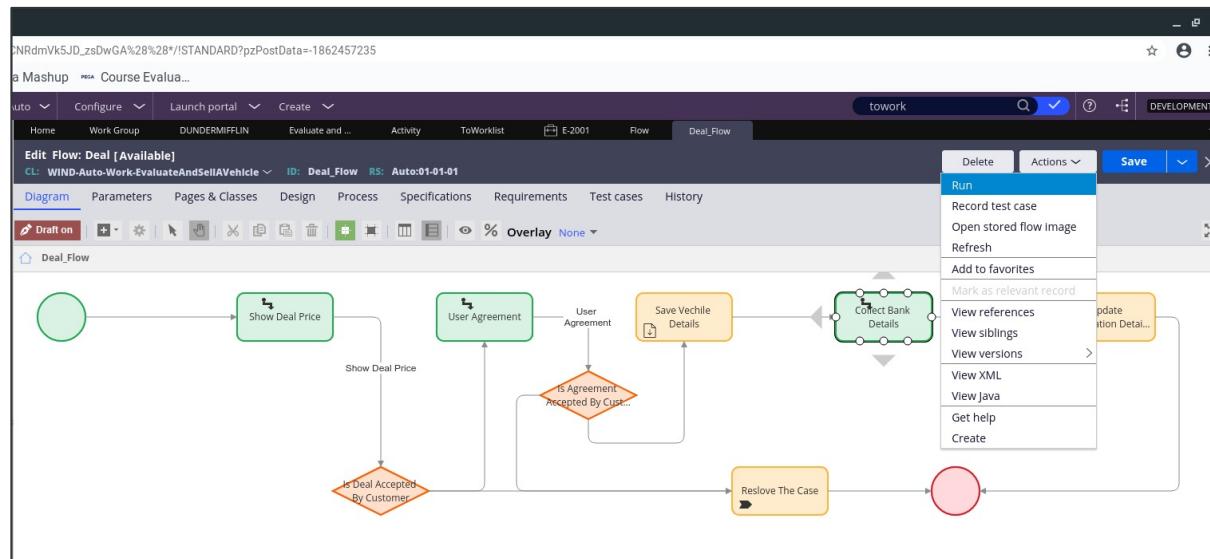


Best Practices

- Reuse existing Flow Actions and Processes.
- Flow Actions and Process are made more reusable by saving them higher in the class structure or in a Framework.

Unit Testing and Debugging Guidance

- Optional Actions and processes are best tested by running the process
- Optional Processes can be tested stand-alone by opening the Flow Rule directly and choosing the Actions menu--> Run.
 - The flow must be in *draft mode*.



Data Management



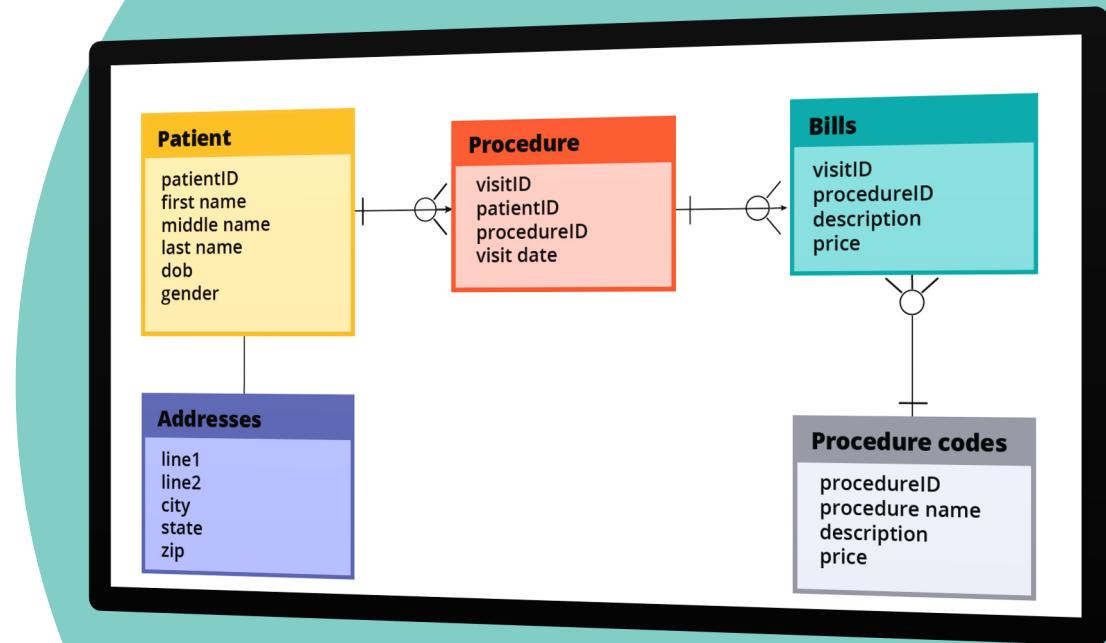
SKILL
LESSON

Property Rules

The use of data elements

Overview

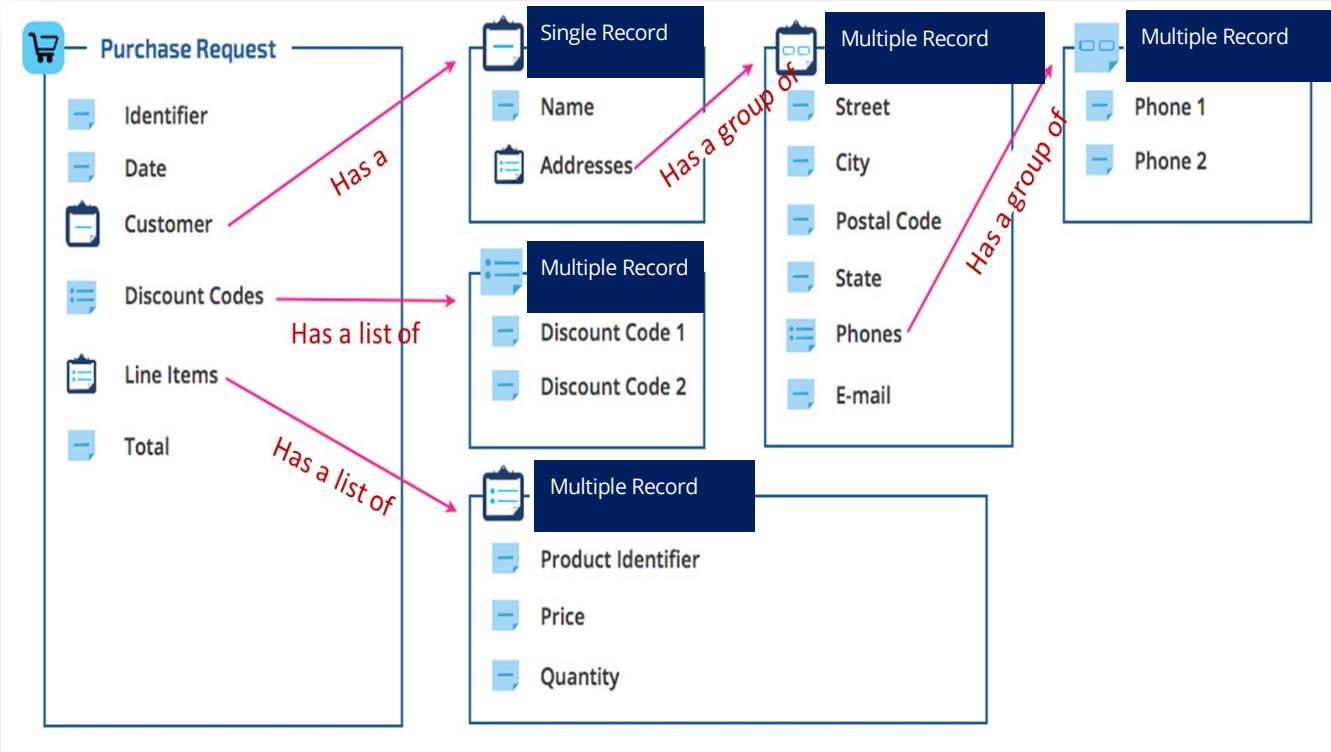
The Pega Platform uses property rules to define data characteristics for handling and processing cases.



Properties

Definition

- Model data in business processes and define the information necessary to provide and reach a business goal.
- Help collect and provide consistent data no matter the method in which users interact with an application.
- To save time and make an application more relevant for customers, reference properties in different elements of an application, and then populate them with relevant data at run time.
- Referencing properties, ensures a user receives the same information through a website, an email, or a mobile app.



Standard property names

Description

- Pega comes with a set of standard property rules.
- The prefix identifies how the standard property can be used.
- System property names start with *px*, *py* or *pz*.
 - **Px** - properties that users see on a form, but cannot directly enter or change values, i.e. *pxCreateDateTime*.
 - **Py** -properties that users can explicitly enter or change i.e. *pyDescription*.
 - **Pz** -properties that are reserved for internal use that end users cannot see, enter, or change i.e. *pzInsKey*.

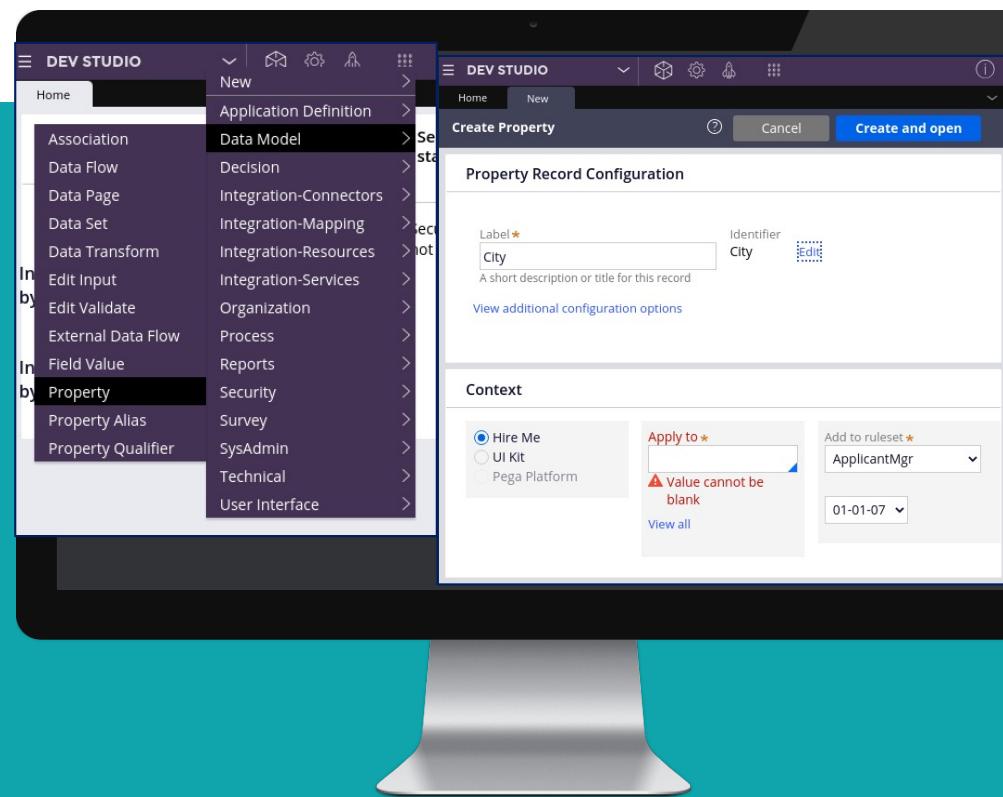
pxUpdateSystemID	pega
pxUrgencyPartyTotal	0
pxUrgencyWork	10
pxUrgencyWorkClass	10
pxUrgencyWorkSLA	
pxUrgencyWorkStageSLA	0
pxUrgencyWorkStepSLA	
pyAgeFromDate	20200708T164935.621 GMT
pyCancelLabel	Cancel
pyConfirmationNote	pyStepRoutedConfirmation
pyCustomerSatisfiedTimestamp	
pyDocumentTitle	
pyElapsedCustomerUnsatisfied	6.0
pyElapsedStatusNew	6.0

Configure property rules

Navigation

From Dev Studio:

1. Select the **Create menu > Data Model > Property**
2. Specify the following:
 - o Label –a short description
 - o Identifier –label name or edit
3. Under context, select the following:
 - o Application
 - o Apply To – the class or case type name
 - o Add to ruleset and version number
4. Click Create and open.



Three ways to create property rules

Implementation

Easy

- In App or Dev Studio
- Using configure view in Case Designer

Simple

- In App or Dev Studio
- Using Data Model in Case Designer

Manual

- In Dev Studio only
- Using Create Menu

Fields

Description

- In the user interface context, properties are referred to as fields.
- Fields are a visual approach to properties.
- At runtime, fields collect, process and present data while processing cases
- The characteristics of data is described using property rules so the application can handle the data properly.
- Each field in a view (screen, form, section), holds a single value that is described by a property rule.

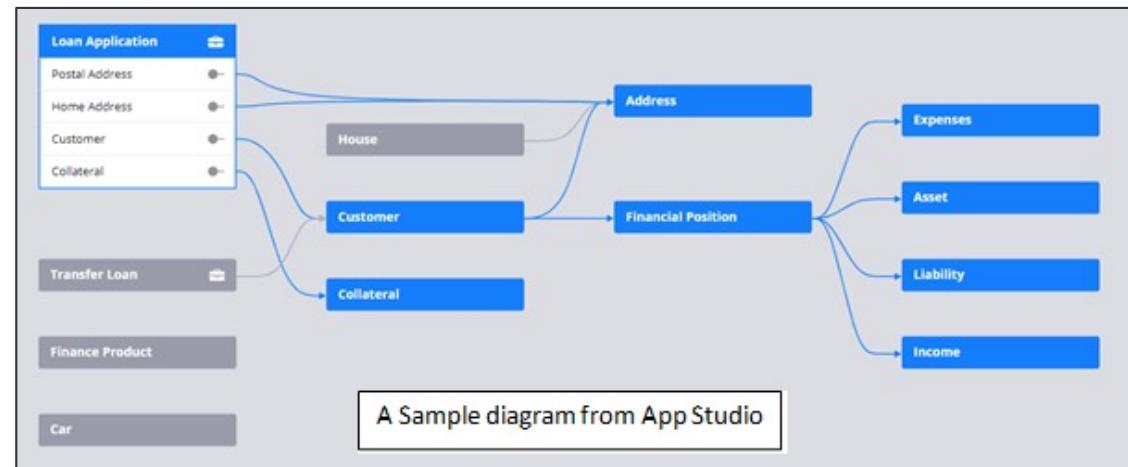
New: Booking

Metro area	Boston	Text
Arrival date	3/16/2018	Date
Departure date	3/30/2018	
Number of guests	2	Integer
Title	Cozy Room	Currency
	Downtown Condo	Price per night
		\$100.00
		\$200.00

Data model

Implementation

- Data elements or collections of related data elements in a case type comprise the data model and defines the **case type data structure**.
- A collection of related data elements is called a **data type** or **data object**.
- The basic steps of data modeling are:
 - Identify data objects.
 - Identify attributes for each data object.
 - Identify existing data objects that can be reused.
 - Relate data objects.
 - Plan for the persistence/integration of the data objects created by the application.



Field types (1 of 3)

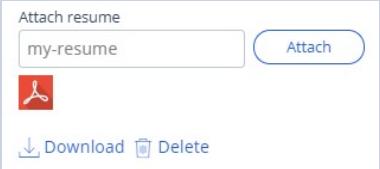
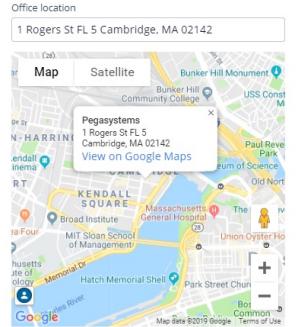
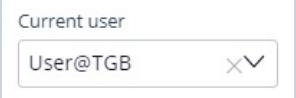
Description

Field type	Type of data
Text (single line)	Any single line of text.
Text (paragraph)	A large text box that accepts multiple lines of text.
Boolean	Allows users to select a check box to indicate one of two possible responses.
Currency	Currency code and value are stored based on the default currency type.
Date & time	UTC (Coordinated Universal Time) value normalized to Greenwich Mean Time (GMT). Date automatically displays in localized format.
Date only	Calendar date with a localized format.
Decimal	Numbers with a fractional component. Use this field type when fractions are needed.

Field type	Type of data
Email	Valid email format with a @ symbol. An email field is an action-oriented control, meaning the value stored in the field displays as a link.
Integer	Positive and negative whole numbers, including the value zero (0).
Phone	Digits display in a localized format. A phone field is an action-oriented control, meaning the value stored in the field displays as a link.
Picklist	A list of predefined values.
Time only	UTC (Coordinated Universal Time) value normalized to Greenwich Mean Time (GMT).
URL	Web address. A URL field is an action-oriented control, meaning the value stored in the field displays as a link.

Field types (2 of 3)

Description

Field type	Type of data	Example
Attachment	Document or file.	
Location	Address input or automatic geolocation.	
User reference	Enter or select a user ID that exists in the system.	

Field types (3 of 3)

Description

Field type	Data source	Example
Data Reference	Single or multiple records from a selected data page.	A user selects from a list of products to order.
Embedded Data	User-supplied data such as a name and address sourced from inside a case type.	A company needs to capture shipping addresses.
Query	A data page or view that is not sourced from inside the case type. The data page defines parameters that the Query data relationship is configured to use.	An application needs to update the current weather.
Case reference	Single or multiple records from a selected case type.	A user selects from a list of service cases from the Service Case type.

How to determine which field type is needed?

Description

How to determine which field type is needed:

- Simple vs. Complex
- Field vs. Data relationship
- Is the data element defined by other data elements?
 - Complex, data relationships provide context for a set of single-value properties

Customer

First Name	Andrea
Last Name	Harte
Email	andrea.harte@example.com

Current Customers			
	Email	First Name	Last Name
1	dimitri.lee@example.com	Dimitri	Lee
2	james.sheen@example.com	James	Sheen
3	haru.mayazaki@example.com	Haru	Miyazaki
4	sarah.jacobs@example.com	Sarah	Jacobs
5	clark.stone@example.com	Clark	Stone
6	frank.singh@example.com	Frank	Singh
7	amanda.flores@example.com	Amanda	Flores

Configure data relationship

Navigation

From App or Dev Studio:

1. Enter a **field name**.
2. Select **Embedded data** for type.
3. Select an existing data object or click Define new data object from the drop-down list.
4. In options, select:
 - *Single record* to create one entry for all the fields
 - *Multiple records* to create a list of separate entries for the fields
5. Click **Submit** to save.

The screenshot shows a configuration dialog box on a computer screen. The dialog has the following fields:

- Field name *****: testing
- Type: Embedded data
- Data object *****: Client
- Options:
 - Single record
 - List of records
- Advanced (button)
- Cancel button
- Submit & add another button
- Submit button (highlighted in blue)

Data Relationships

Implementation

- `.CreditCards` is a multiple record data relationship
- It is being used by `.Customer` which is a single record data relationship
- `.Customer`
 - `.FirstName`
 - `.LastName`
 - `.CreditCards()`

Customer

First Name
Sandra

Last Name
Washington

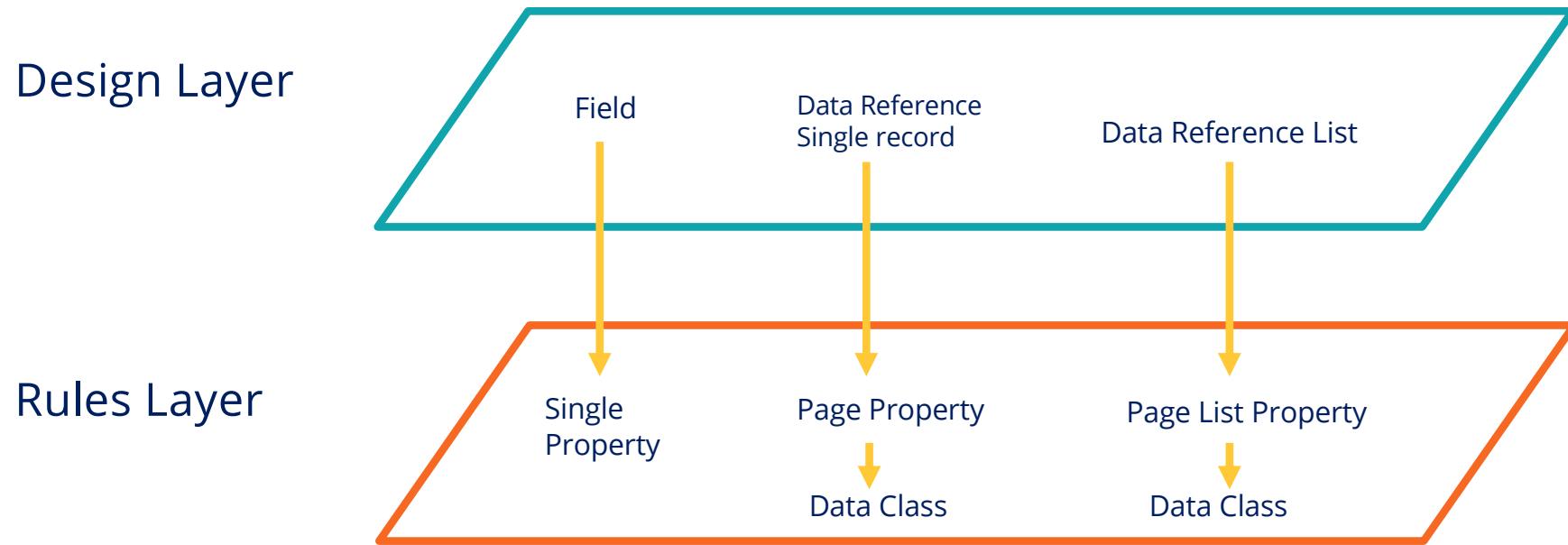
Credit Cards

+ Add item X Delete

	Card Number	Expiration Date	Verification Code
1	0000 0000 0000 0000	01/2025	321
2	1111 1111 1111 1111	02/2024	456



Schematic



Java HTML CSS XML Javascript JSP

1 1 0 0 1 0 1 0 1 0 1 1 1 0 0 1

Best Practices

- Property names are case-sensitive, use only letters and digits.
- Do not start property names with an underscore (_) or dollar sign (\$).
- Reuse data objects by promoting a built-on data object or copying the assets from a standard Data- class.
- If your data source is unavailable, connect your data objects to simulated data in local storage.
- Re-purpose data views instead of creating new ones when your system of record changes.
- Update data sources created in App Studio in Dev Studio only when necessary because some changes made in Dev Studio make the rules un-editable in App Studio.

Skill Mastery

Understand:

- Properties
- Fields
- The data model
- Field types: simple, complex and fancy
- Data relationships
- Property typology
- Configuring property rules
- Standard properties
- Property modes
- Referencing properties



SKILL
LESSON

Data Validation



Overview

End users input valuable information into applications to automate decisions and processes.

This data may need correction and validation. Validation data integrity must be considered for information entered by users.

This module covers:

- Data validation in App Studio
- Validating a user's field input
- Validating character patterns
- Adding a validation rule to a form/flow action

The screenshot shows a mobile application interface titled "Enter payment information" with a subtitle "DUE IN 4 DAYS AGO". The main instruction reads "Enter payment information for customers with standard coverage". A section titled "Payment information" contains fields for "Card type" (set to "Visa"), "Card number" (containing the value "1111222233334444"), and "Expiration date" (set to "5/12/2020"). A red error message "⚠ 'Expiration date is not valid.'" is displayed next to the expiration date field. At the bottom is a blue "Cancel" button.

Use Case

- Validate case data
- When input is required
- When you need to restrict the choices a user can select
- When you want to ensure the input conforms to a pattern
- When you want to ensure the correctness of discrete or dependent information



Validation

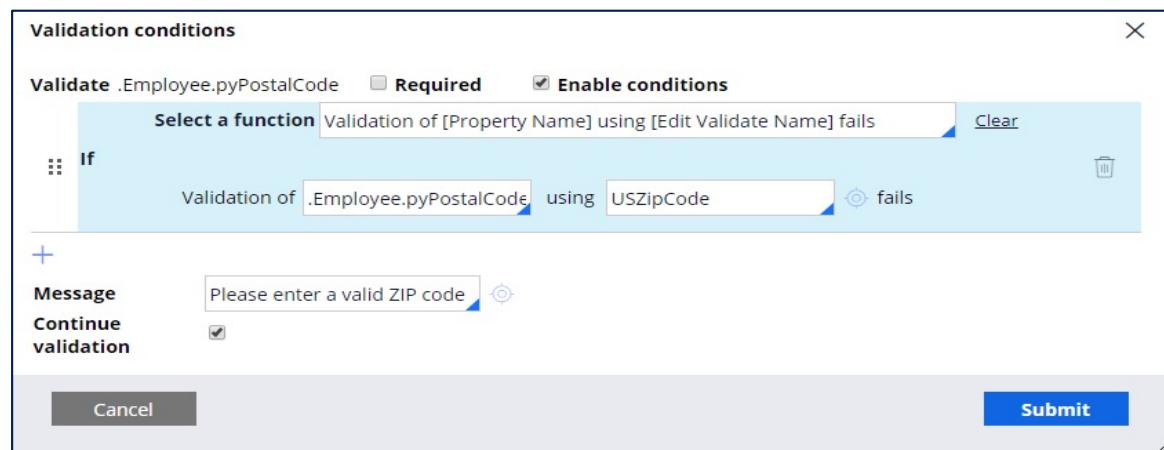
Definition

Validation is used to ensure that data is correctly formatted and valid before being input into an application. This allows the system to process the information without errors.

Use validation rules when you cannot predict or control the value users enter in a form.

The data must:

- Be the right type
- Fit the business logic
- Be restricted to possible values



Pega also provides property types, controls, and rules to support validation requirements.

Different Types of Validation

Description

- **Property Type** - User can only enter the type of value as defined by the property
- **Control** - restricts users from entering or selecting invalid values on a form
- **Validate** – to compare a property against a condition
- **Edit Validate** – to test single value, value list, and value group properties for patterns
- **Required** – User must enter a value

Enter payment information
DUE IN 3 HOURS FROM NOW

Enter payment information for customers with standard coverage

Payment information

Card type*

Select...

⚠ Value cannot be blank

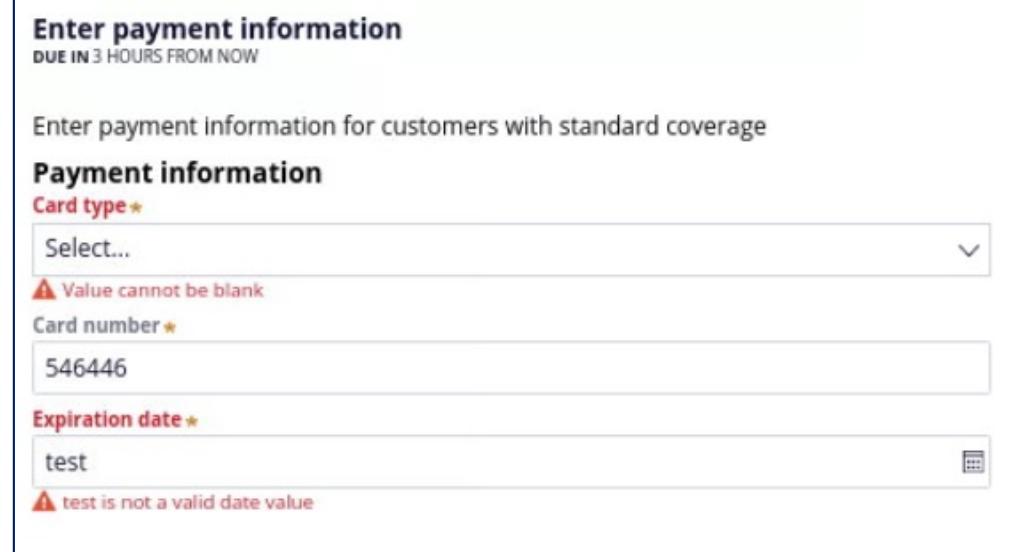
Card number*

546446

Expiration date*

test

⚠ test is not a valid date value



Business Logic Data Validation

Description

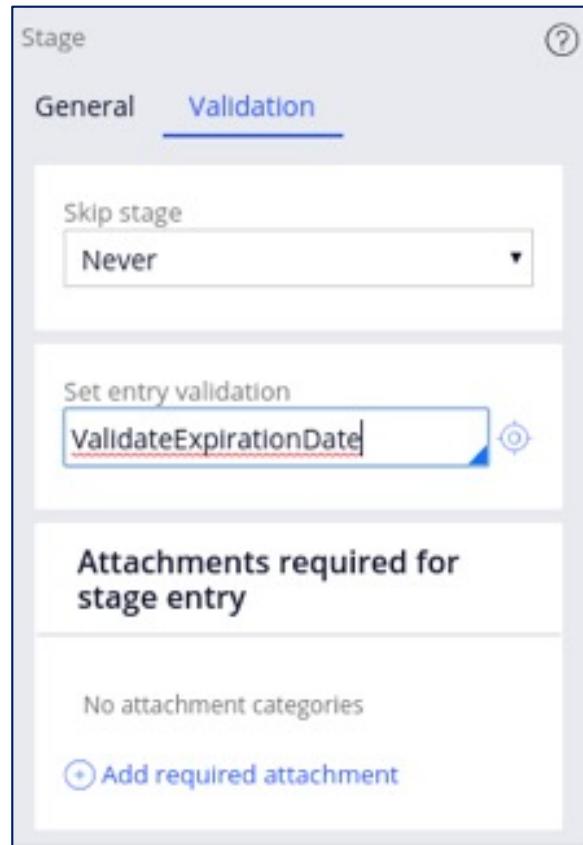
- You use App Studio to perform simple business logic validations that compare the value of a field against a constant value when users submit a form.
- Business logic validations are often associated with processes in the case life cycle, enabling you to validate each field instance based on distinct business logic validations.
- The business logic validations that define acceptable values are separate from the fields that capture data.
- Multiple forms may use the same field and apply different validation conditions for each instance of the field.



Stage Validation

Description

You can create a validate rule in Dev Studio to provide an entry condition for a stage in the case workflow



Validation – Case Data Model

Case Type > Data model tab > Validations

The screenshot shows the 'Edit case type: Job Applicant' interface. The 'Data model' tab is selected. In the 'Validations' section, a validation rule is being configured for the 'Email' field. A modal window titled 'Stage entry validation' is open, showing the condition 'Email is empty' and the error message 'Please enter an email address to continue.' An orange arrow points from the validation message in the table to the corresponding row in the modal window.

Property	Collect Resume	Recruiter Review	Interview	Decision	Offer
401 K match					
Applying for					
Average Rating					
Car Suppliment					
Compensation					
Compensation increase		Add entry validation			
Compensation increase rational					
Compensation Recommendation					
Email		→ Stage entry Please enter an email address to c...			
Employment History					

Stage entry validation

When conditions are met (1) Group ANDS

Email is empty

Then display error message as *

Please enter an email address to continue.

To validate stage entry for the stages

Collect Resume

Recruiter Review

Interview

Decision

Cancel Submit

Validation Rule Types

Definition

There are two types of validation rules: **validate** and **edit validate**.

- **Validate rules** compare a property against a condition when the user submits a form.
- **Edit validate rules** test single value, value list and value group properties for patterns. This rule requires knowledge of Java.

The screenshot shows the 'Property: Email ID [Available]' configuration screen. Under the 'Validation' tab, there are fields for 'Max length', 'Expected length', 'Override sort function', 'Access when', 'Edit input', and 'Use validate'. The 'Use validate' field contains 'ValidEmailAddress' and has a blue link labeled 'Edit Validate'. An orange arrow points from this link to the 'Edit Validate' dialog box below.

Validation

- Max length
- Expected length
- Override sort function
- Access when
- Edit input
- Use validate
ValidEmailAddress

Edit Validate: Validate if the String value entered is a valid email address

Java Source

```
1 if (theValue == null || theValue.trim().equals
2 try {
3     javax.mail.internet.InternetAddress emailAdd
4     emailAddr.validate();
5 }
6 catch (javax.mail.internet.AddressException ex
7     return false;
8 }
```

The screenshot shows the 'Validation conditions' dialog box. It is configured to validate '.Employee.pyPostalCode' and is set to 'Required'. The 'Enable conditions' checkbox is checked. The condition is defined as 'Validation of [.Employee.pyPostalCode] using USZipCode fails'. A message is defined as 'Please enter a valid ZIP code'. The 'Continue validation' checkbox is checked. The dialog includes 'Cancel' and 'Submit' buttons.

Validation conditions

Validate .Employee.pyPostalCode Required Enable conditions

Select a function Validation of [Property Name] using [Edit Validate Name] fails

If Validation of [.Employee.pyPostalCode] using USZipCode fails

Message Please enter a valid ZIP code

Continue validation

Cancel

Validate Rule

Description

- Configuring a Validate Rule requires first specifying a property
Enable conditions to select a function that will perform the validation on the specified property
Add a message to display to the user if the condition is met

The screenshot shows the 'Validate' tab of a Pega application. The title bar indicates 'Validate: CheckList [Available]' and the class is 'WIND-Auto-Work-EvaluateAndSellAVehicle'. The 'ID' is 'CheckList' and the 'RS' is 'A'. The main area is titled 'Default Validation' and contains a 'PROPERTY' section. A dropdown menu for 'Key' is open, showing options like 'Accepted', 'BankDetails', 'CustomerDetails', 'DuplicateKey', 'InsuranceCertificate', 'Invoice', 'Key', 'PollutionControlCertificate', 'PUC', 'RC', 'Reason', and '.Invoice'. The 'Key' option is selected. To the right of the dropdown, there are 'Req' (Required) and 'Conditions' sections. The 'Conditions' section includes an 'If' condition: 'IF !.Key THEN display message: Key is mandatory'. Below this, there are 'Message' and 'Continue validation' sections.

The screenshot shows the 'Validation conditions' dialog box. It has a 'Select a function' dropdown where 'If' is selected. Under 'If', '!.Key' is chosen. The 'Message' section is checked. The 'Submit' button is visible at the bottom right. The 'Conditions' section lists various validation functions:

Condition	Action
[a datetime] is in the [past/future]	pxDateTimeIsPastOrFuture
[assignedOperatorID] is on my staff	pxAssignedToMyStaff
[expression evaluates to true]	1FreeFormExpressionBoolean
[First DateTime] [relation] [Second DateTime]	CompareTwoDateTimes
[first number] [relation] [second number]	CompareTwoNumbers
[first String] [relation] [Second String]	compareTwoStrings
[first string] does not equal (ignore case) [second string]	StringNotEqualsIgnoreCase
[first string] equals (ignore case) [second string]	StringEqualsIgnoreCase

Validate Rule

Implementation

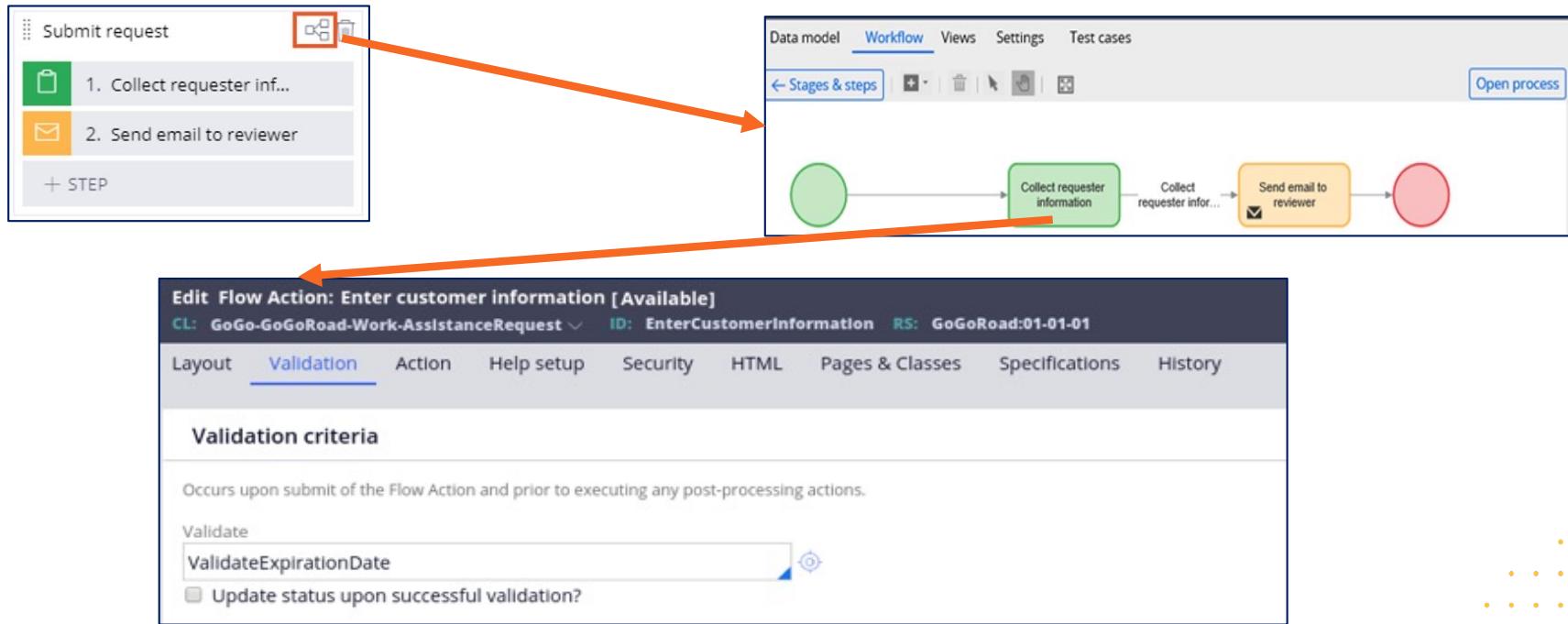
The following example demonstrates applying a validate rule that ensures users enter a key that the invoice cost is greater than a \$1,000, (and provides a message stating so) and the signed documents are mandatory.

PROPERTY	*Req Conditions	
.Key	IF !.Key THEN display message: Key is Mandatory	Edit
.Invoice.Cost	IF .Invoice.Cost >= 1000.00 THEN display message: Invoice is not acceptable	Edit
.SignedDocumentsForRCTran	IF !.SignedDocumentsForRCTransfer THEN display message: Signed Documents are Mandatory	Edit

Flow Rule Validation

Implementation

- Validate a user form/flow action by opening the Flow rule.
- In the Flow rule right click on the connector and open the flow action you like validated.
- In the Validation tab specify the validation rule you want to use.

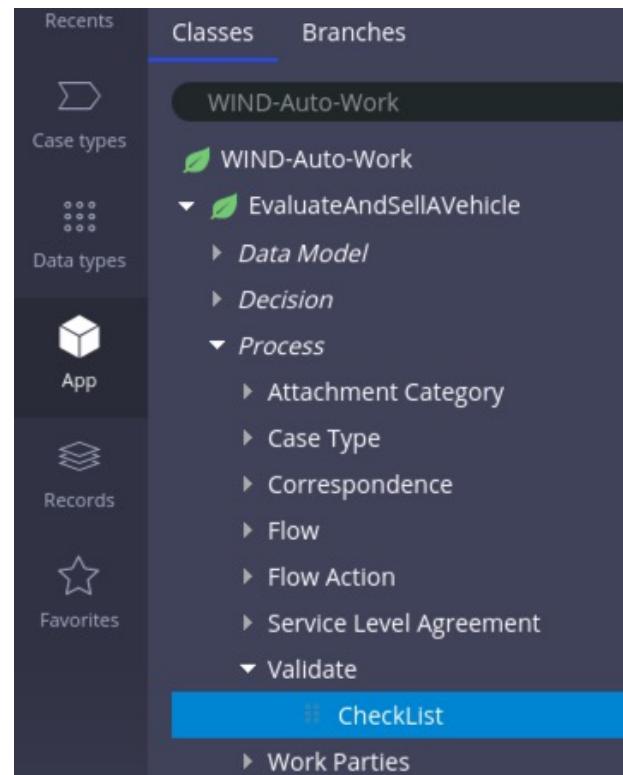


Case Data – Dev Studio

Navigation

Case Data Validation – Dev Studio

1. Dev Studio
2. App Explorer
3. Expand Case Type
4. Expand Process
5. Expand Validate



Input Property

Description

You may qualify validation for a property based on an action performed by the user.

For example, you may qualify a validation based on data provided by the user. This is done by referencing the property in the Input tab.

Edit Validate: CheckList [Available]
CL: WIND-Auto-Work-EvaluateAndSellAVehicle ▾ ID: Checklist

Validate **Input** Pages & Classes Specifications History

None
 Input property
 Proposed work status
 Flow action name
 Stages

Edit Validate: CheckList [Available]
CL: WIND-Auto-Work-EvaluateAndSellAVehicle ▾ ID: Checklist

Validate **Input** Pages & Classes Specifications

None
 Input property .ExperienceLevel
Label Experience Level

Proposed work status
 Flow action name
 Stages

Edit Validate Rule

Definition

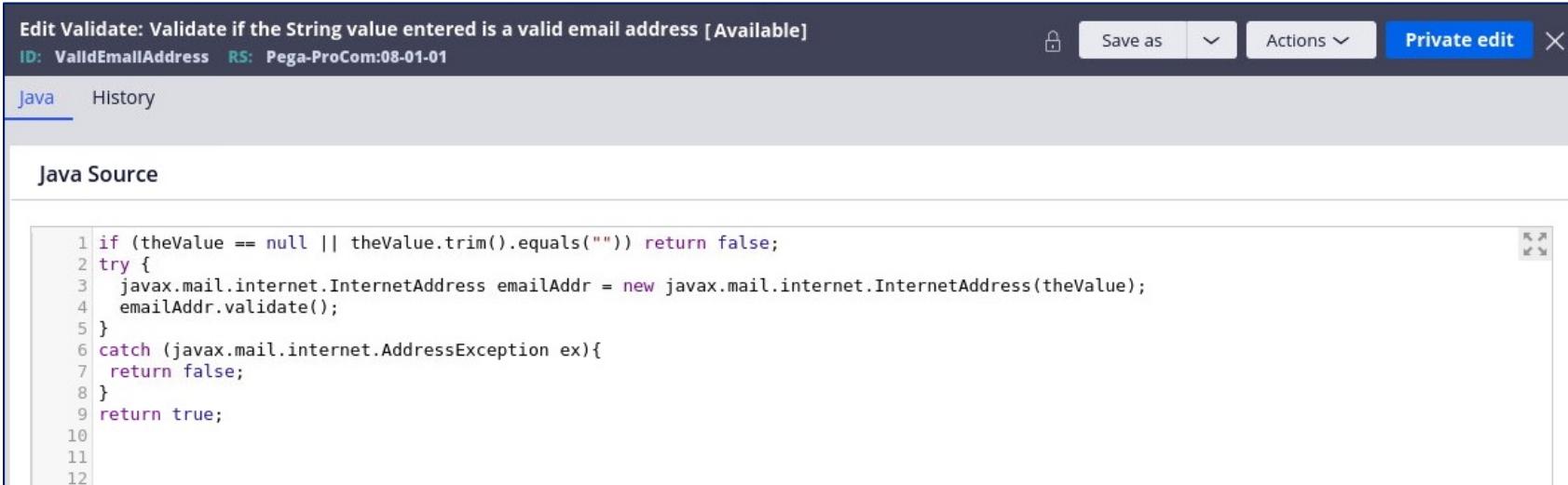
- Edit validate rules validates character patterns.
- For example, you can use an edit validate rule to ensure that the field contains the @ symbol in an email address. If it does not, an error message appears when the form is submitted.



Edit Validate Rule

Implementation

The following example demonstrates applying an edit validate rule to ensure that a user enters a valid email address.



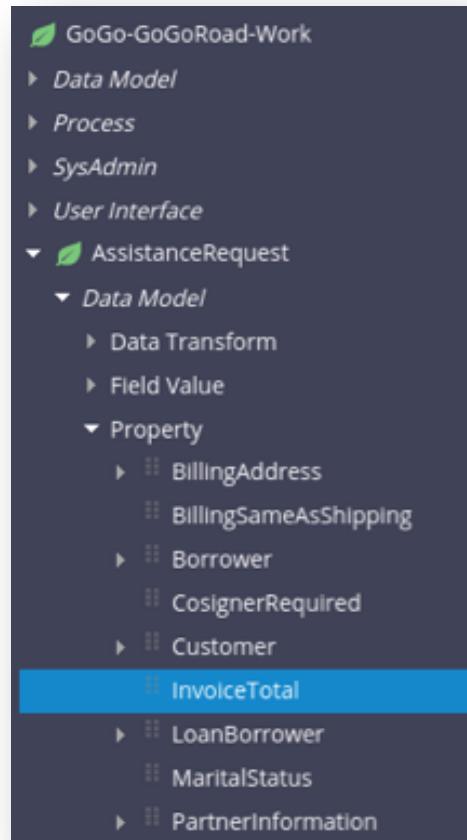
The screenshot shows the 'Edit Validate' interface for a rule named 'Validate if the String value entered is a valid email address'. The Java tab is selected, displaying the following code:

```
1 if (theValue == null || theValue.trim().equals("")) return false;
2 try {
3     javax.mail.internet.InternetAddress emailAddr = new javax.mail.internet.InternetAddress(theValue);
4     emailAddr.validate();
5 }
6 catch (javax.mail.internet.AddressException ex){
7     return false;
8 }
9 return true;
10
11
12
```

Edit Validate Rule

Implementation

1. Dev Studio
2. App Explorer
3. Expand Case Type
4. Expand Data Model
5. Expand Property
6. Select Advanced tab

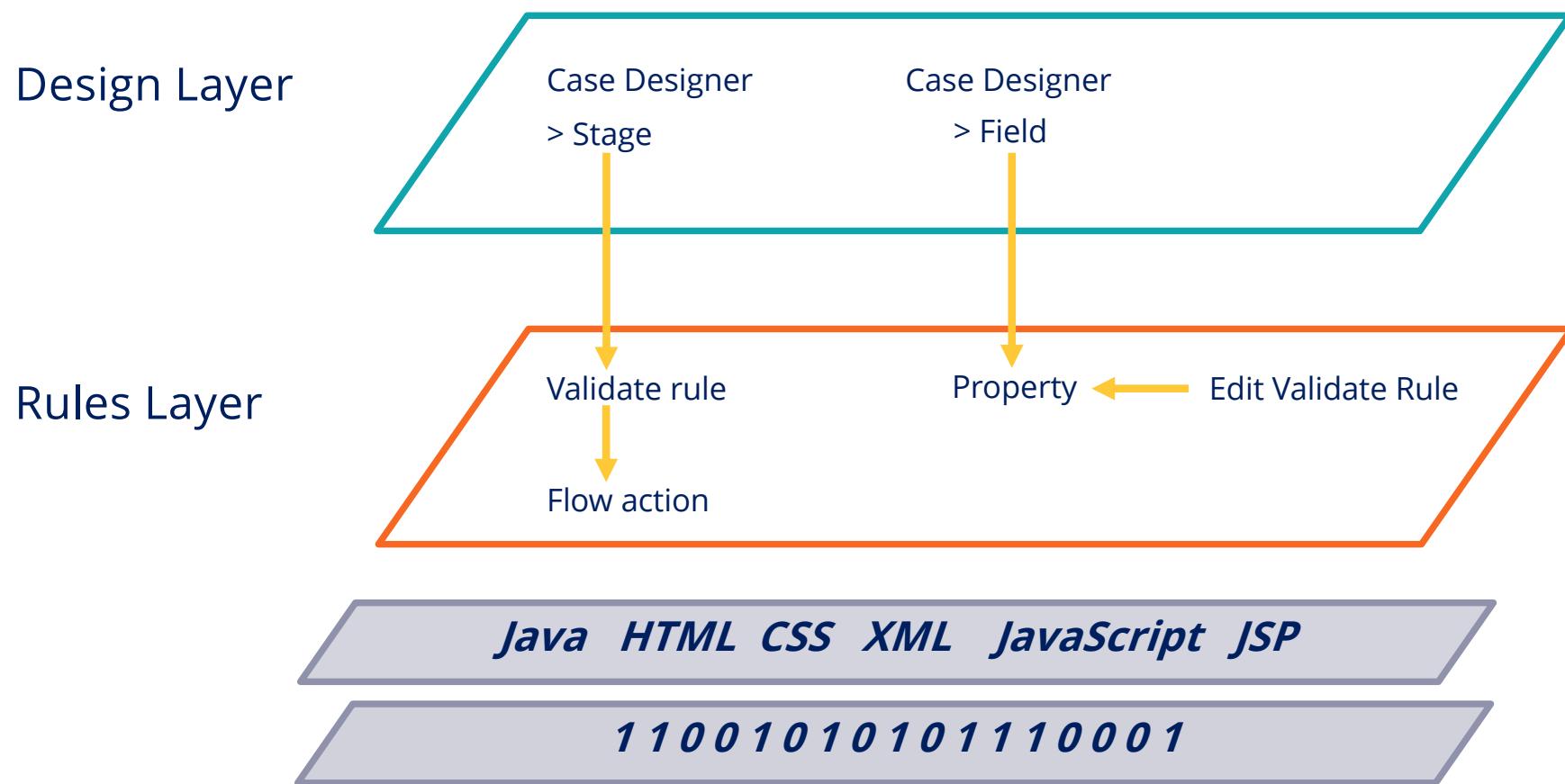


The screenshot shows the 'Edit Property' dialog for the 'InvoiceTotal' property of the 'AssistanceRequest' case type. The 'Advanced' tab is selected. The 'Validation' section contains the following fields:

- Expected length: (empty)
- Override sort function: (empty)
- Access when: (empty)
- Edit input: (empty)
- Use validate: SSN
- Column inclusion: (empty dropdown)



Schematics



Unit Testing and Debugging Guidance

- Create and reference a validate rule in the case
- Run the case and input values that will cause the property to fail validation
- Confirm that the validation message is displayed on the form and the user is not allowed to proceed without fixing the entry

Skill Mastery

- Evaluate the input requirements for discrete pieces of information
- Evaluate the association between related information
- Determine the method of validation
- Configure the validation method
- Data validation in App Studio
- Validating a user's field input
- Validating character patterns
- Adding a validation rule to a form/flow action



SKILL
LESSON

Viewing the data model

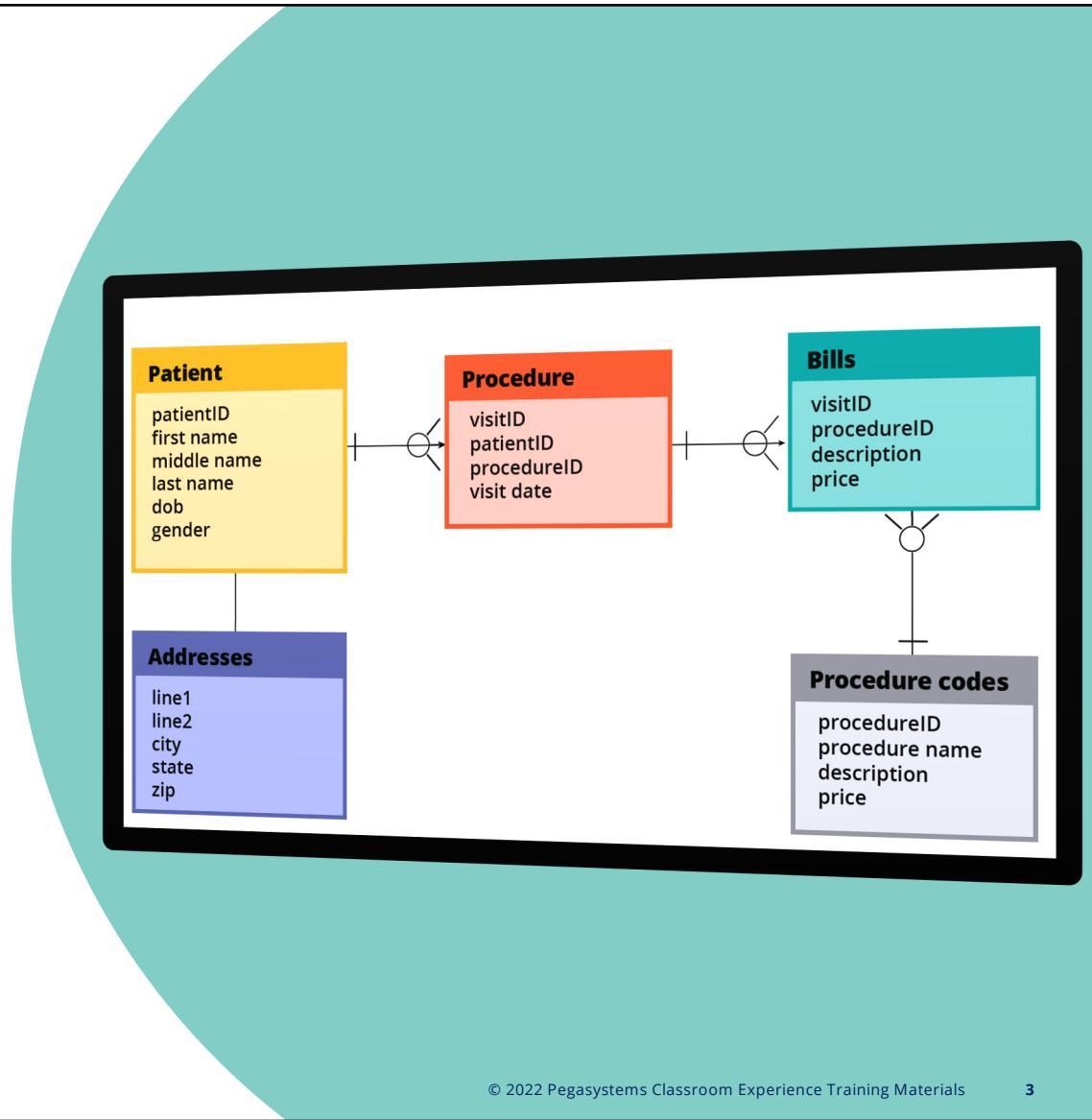
Visualizing data with the Integration Designer

Overview

The data model defines the types of information and relationships between the data in the application. Application developers can view the relationships between data objects to gain insight into the structure of the data model.

This lesson covers:

- The data model
- Viewing the data model
- The Integration Designer
- View integration landscape
- Relationships between data objects, cases, and systems of record
- Navigating the Integration Designer

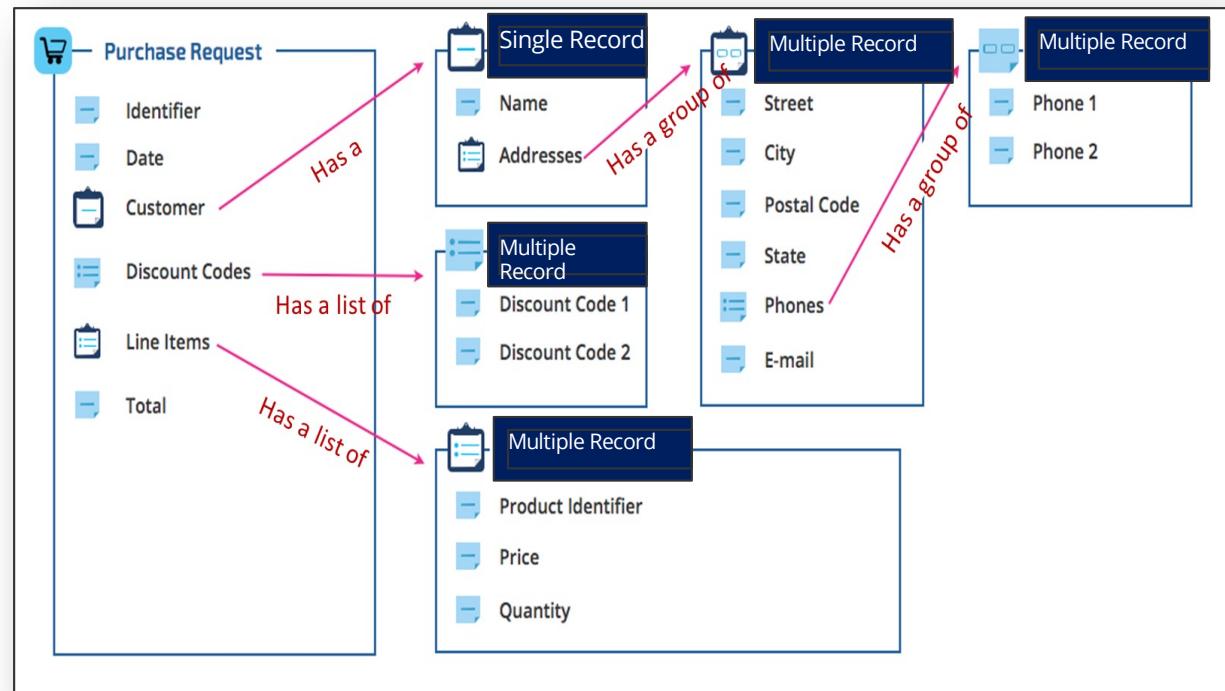




Data Modeling

Use Case

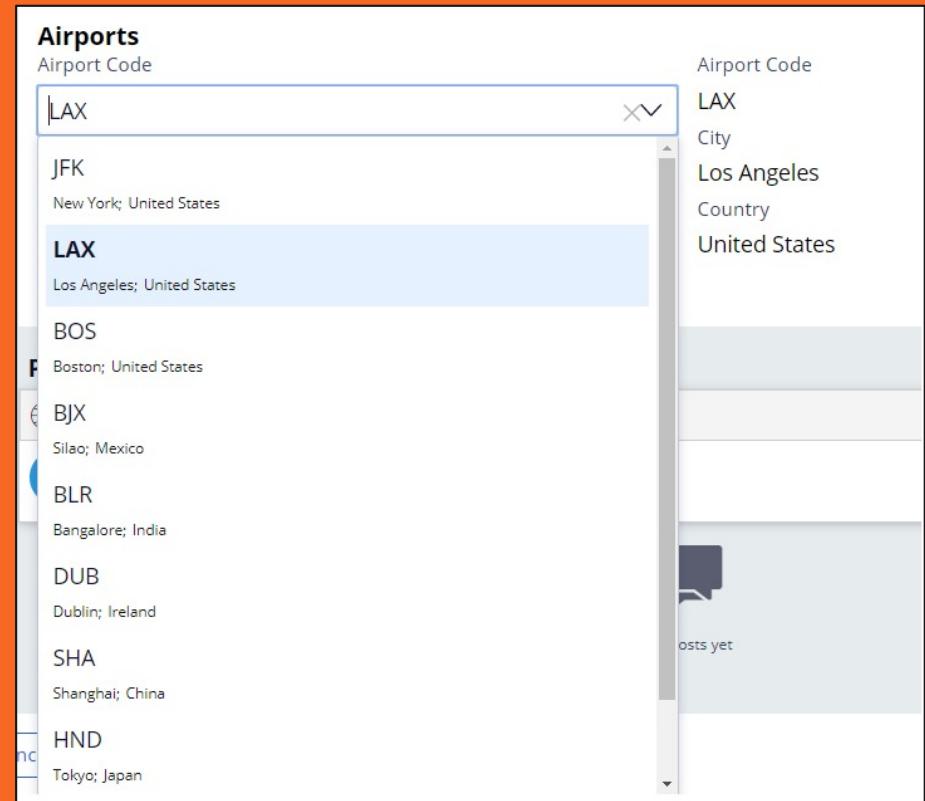
- The data model brings information into your application in a format that makes sense for a business.
- Data modeling allows
 - viewing data and case type relationships.
 - creating new data types.
 - updating data type records.



Data Records

Definition

- Applications often require access to data that is used to process cases but is not directly part of the application.
- Data records define permissible values, limiting input values to reduce errors.



In this example, a UI control is being used to return a list of data records for the user to select one to populate the field.

Data Records versus Case records

Description

- **Data records**

- Represents key business entities (e.g. Customers)
- Contain all the fields necessary to describe it
 - Identifier
 - Name
 - Address
 - Date of creation
 - And more

Data object is a template of the data that will be stored as a data record.

- **Case records**

- Represents business transactions (e.g. Assistance Request)
- Contains all the fields necessary to describe it
 - Customer (who needs assistance) (Identifier, Name, Address, Date of Creation, etc)
 - Location of vehicle (address, city, state)
 - Type of vehicle (make, model, year, color)

Case type is a template of the business process which captures the data that is stored as a case record.

Local data storage and External data sources

Description

- **Local data storage**

- pyGUID – Globally unique ID created by default when created in App Studio
- Stored locally in a database table in Pega. Adding records in AppStudio or DevStudio creates a record in the mapped local database table in Pega. Deleting records removes them from the database table.

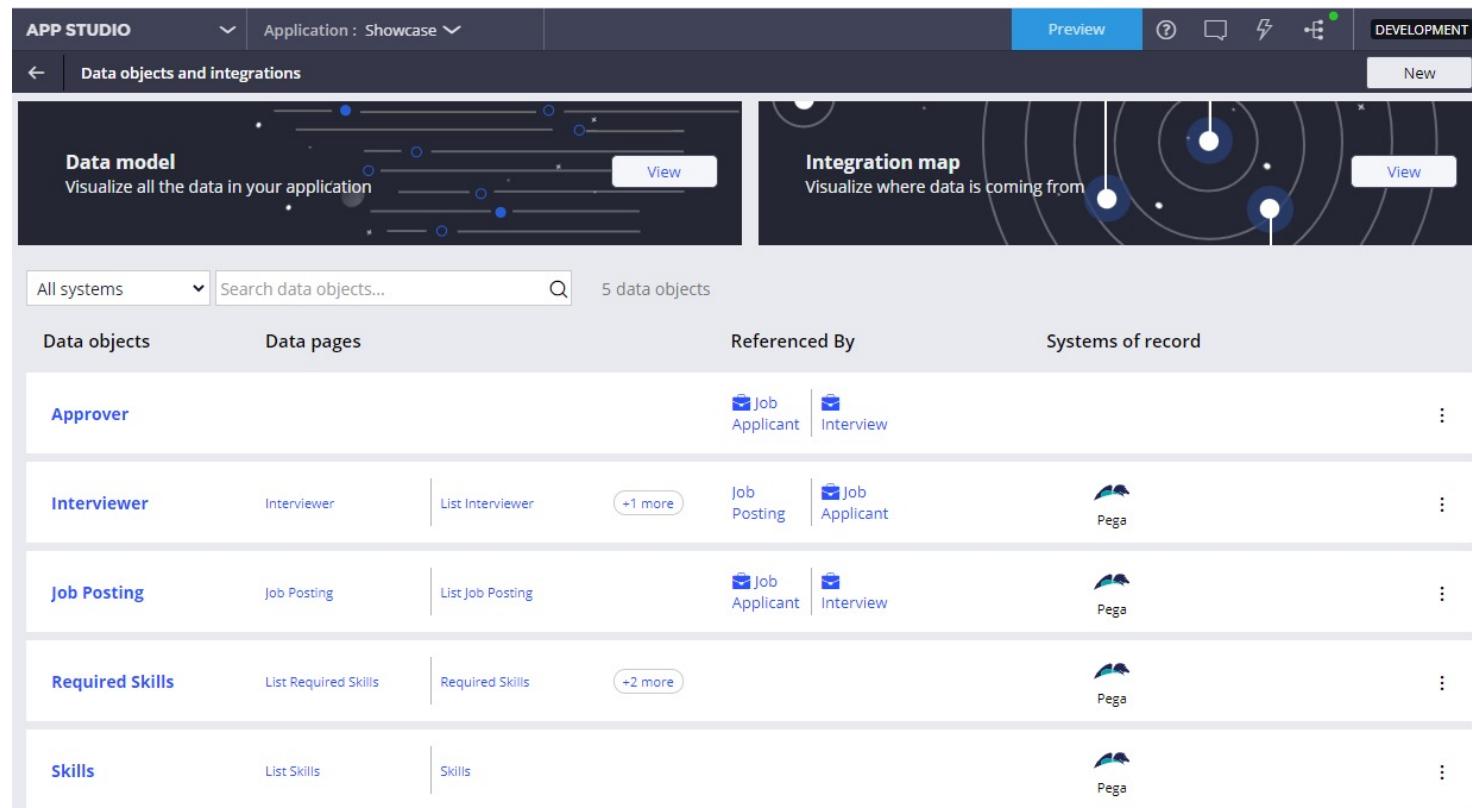
- **External data sources**

- Retrieve records from an external database or web service.
- Configured using data pages.

Data model	Records	Views	Data views	Settings
ID*	Name	Country		
ATL	Atlanta	United States		
BER	Berlin	Germany		
HQ	Cambridge	United States		
LON	London	United Kingdom		
TOK	Tokyo	Japan		
VAN	Vancouver	Canada		
+ Add record				

The integration designer

Definition



The screenshot shows the Pega App Studio interface with the following elements:

- Header:** APP STUDIO, Application: Showcase, Preview, DEVELOPMENT.
- Left sidebar:** Data objects and integrations.
- Data model:** Visualize all the data in your application. (View button)
- Integration map:** Visualize where data is coming from. (View button)
- Search bar:** All systems, Search data objects... (5 data objects found).
- Data objects table:** A grid showing data objects and their relationships.

Data objects	Data pages	Referenced By	Systems of record
Approver		Job Applicant Interview	Pega
Interviewer	Interviewer List Interviewer (+1 more)	Job Posting Job Applicant	Pega
Job Posting	Job Posting List Job Posting	Job Applicant Interview	Pega
Required Skills	List Required Skills Required Skills (+2 more)		Pega
Skills	List Skills Skills		Pega

- Comprehensive view
- Displays data objects, case types and data types.
- Create data objects
- Access data objects
- Update data objects

The integration designer

Implementation

The screenshot shows the Pega Integration Designer interface. At the top, there are two main sections: "Data model" (Visualize all the data in your application) and "Integration map" (Visualize where data is coming from). Below these are search and filter controls: "All systems" dropdown, "Search data objects..." input field, and a "View" button. The main area displays a table of data objects:

Data objects	Data pages	Referenced By	Systems of record
Address	Address	List Address +1 more	Client Pega
Car	List Car	Client	VehicleDetailsAPI
Client		Car Insurance	
Person	List Person SIMULATED	Person SIMULATED	Client Undefined

The data model

Definition

- Helps understanding the relationships among case types, data objects, and properties in an application. To model data, you need the following components:

- **Data objects**

- Categories of data that have fields, field mappings, and connections to data sources.

- **Fields**

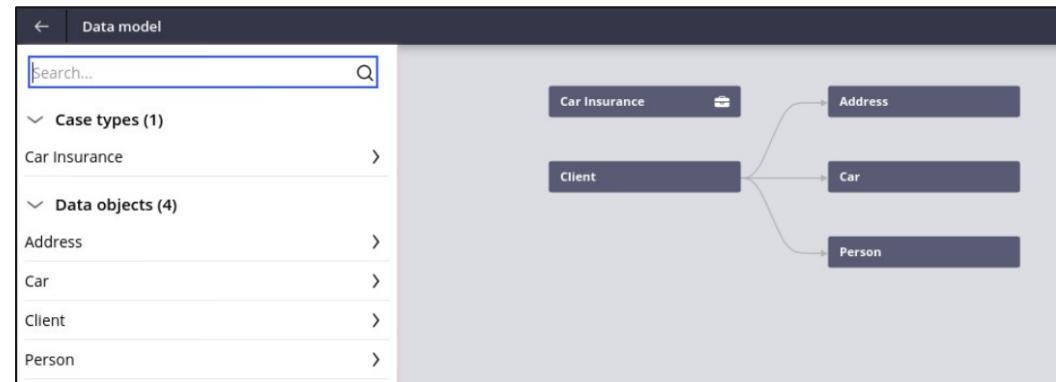
- Properties that store and format the data in your application.

- **Data sources**

- Resources, which can be real or simulated, that host the data.

- **Field mappings**

- Logic that links or transforms the fields in a data source to the fields in an application.

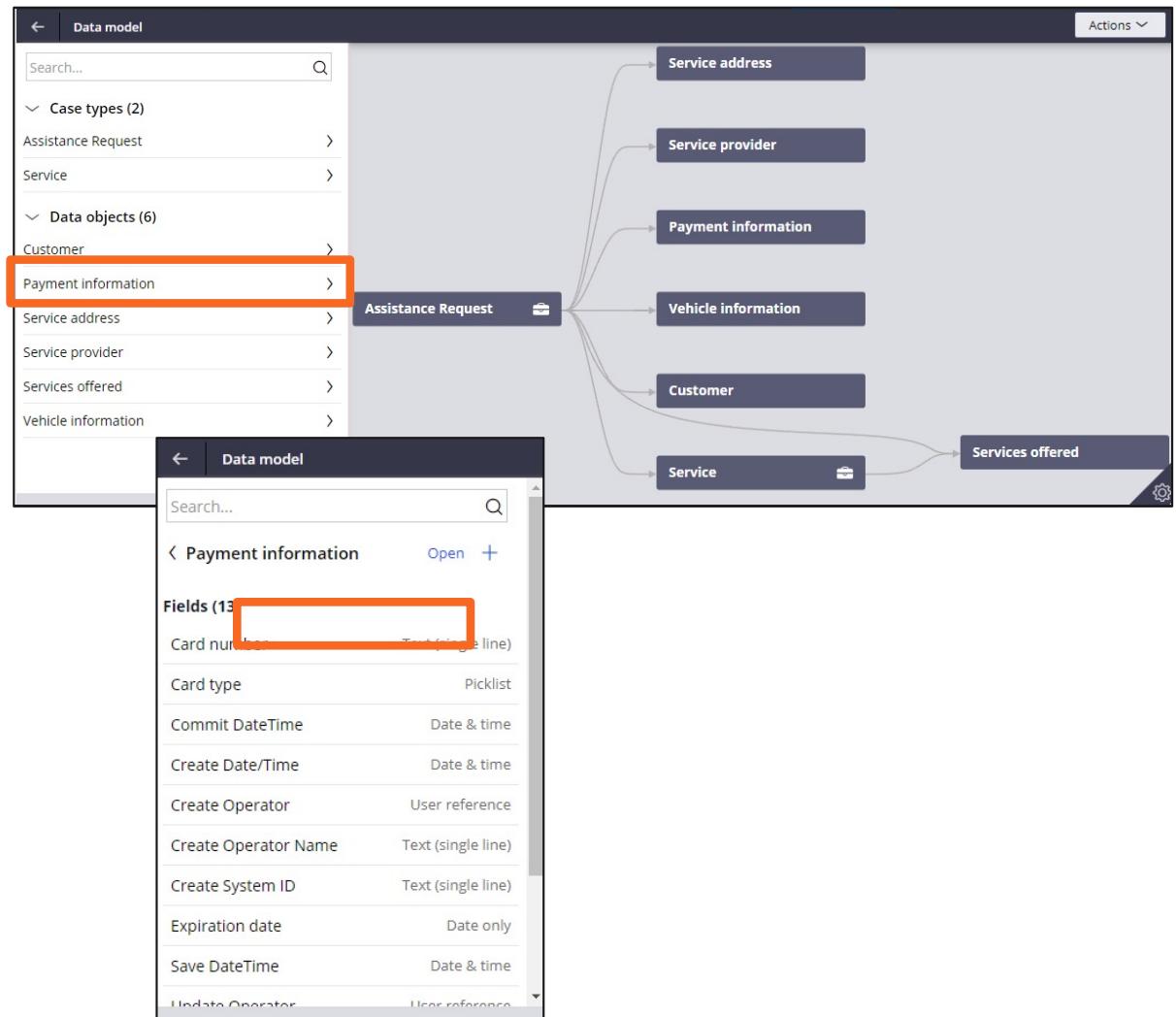


View the data model

Navigation

App Studio > Data > View

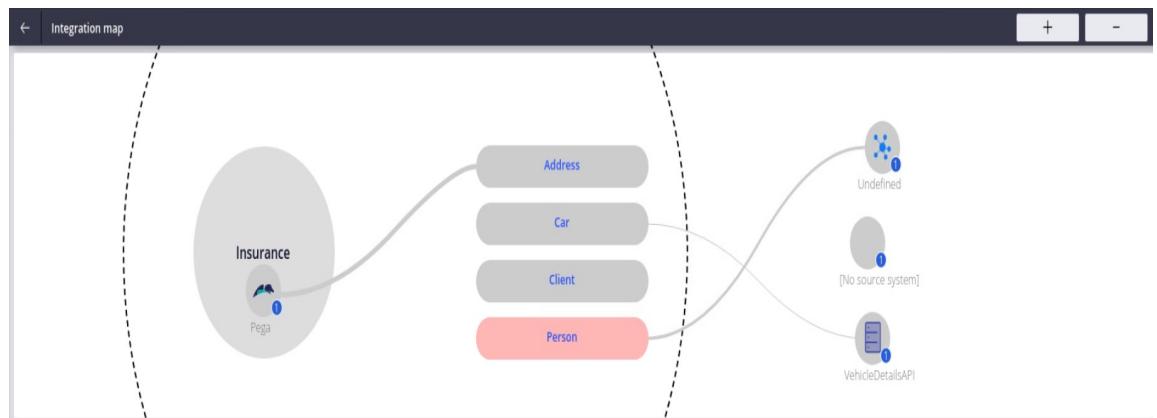
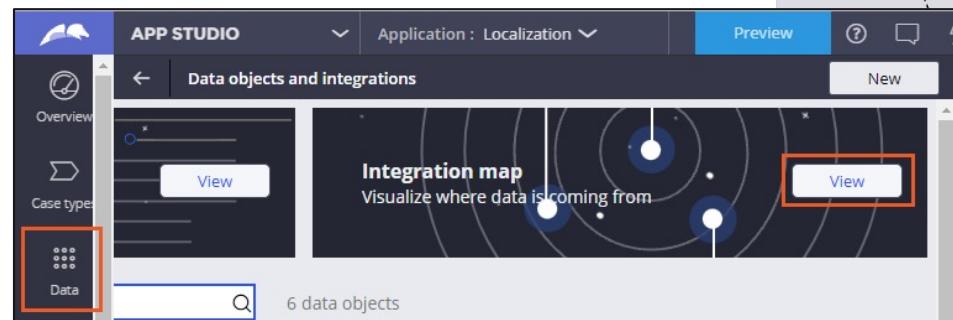
The screenshot shows the Pega App Studio interface with the 'Data' icon highlighted. The main area displays a 'Data model' visualization with various entities like 'Assistance Request', 'Customer', 'Service address', etc., connected by lines. A 'View' button is highlighted with a red box. Below the visualization, there's a search bar for 'Search data objects...' and a dropdown for 'All systems'.



View integration map

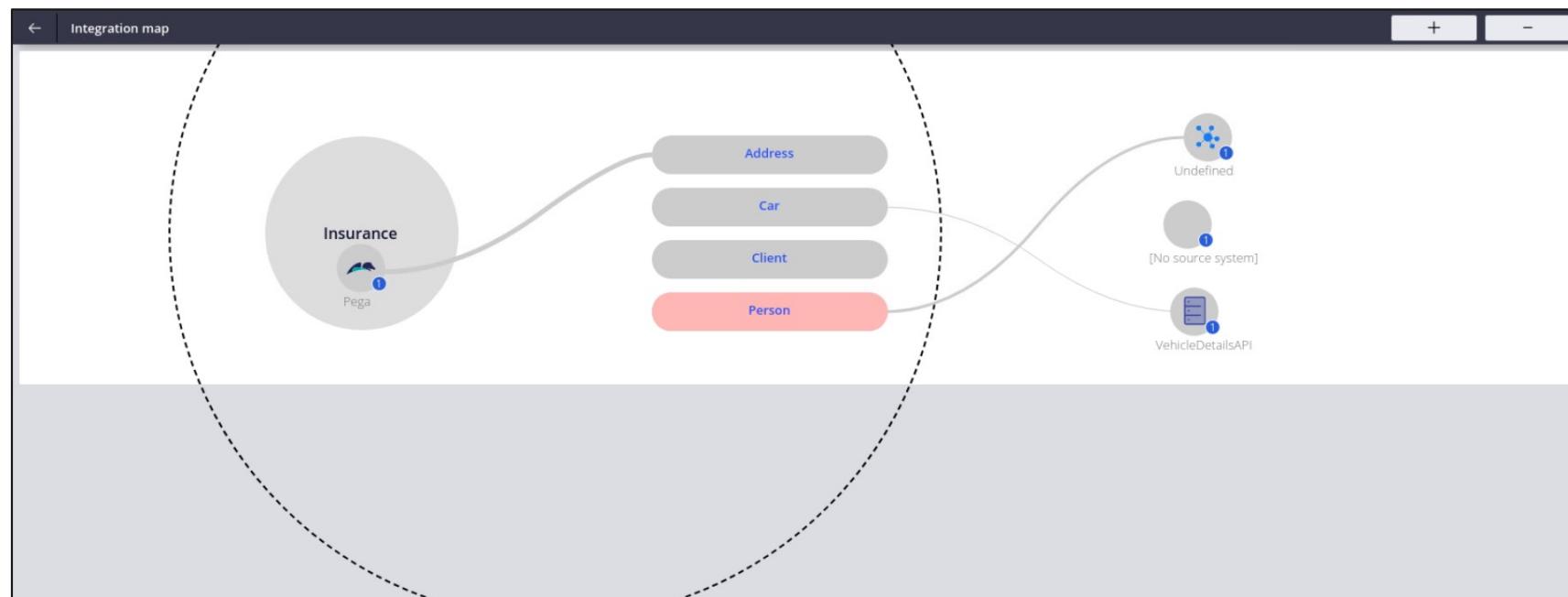
Navigation

- AppStudio > Data > View
- Use the interactive integration landscape to understand the relationships between data and data types and systems of record.



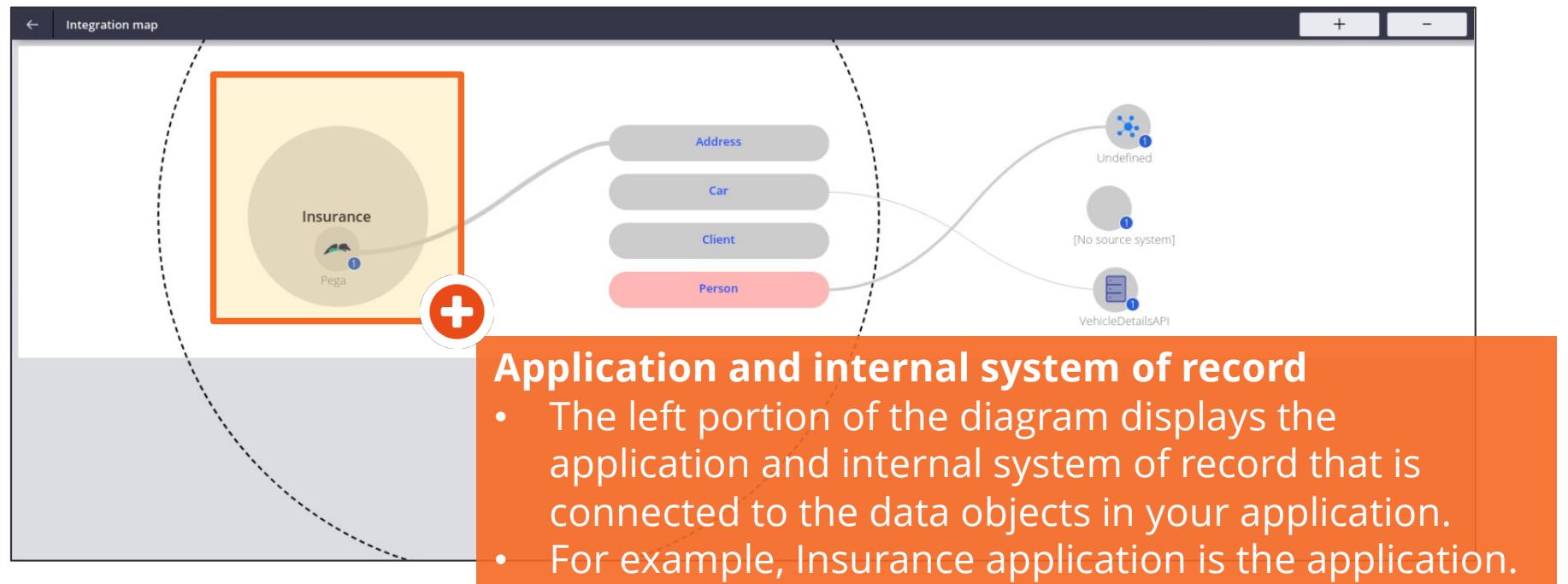
Integration map

Implementation



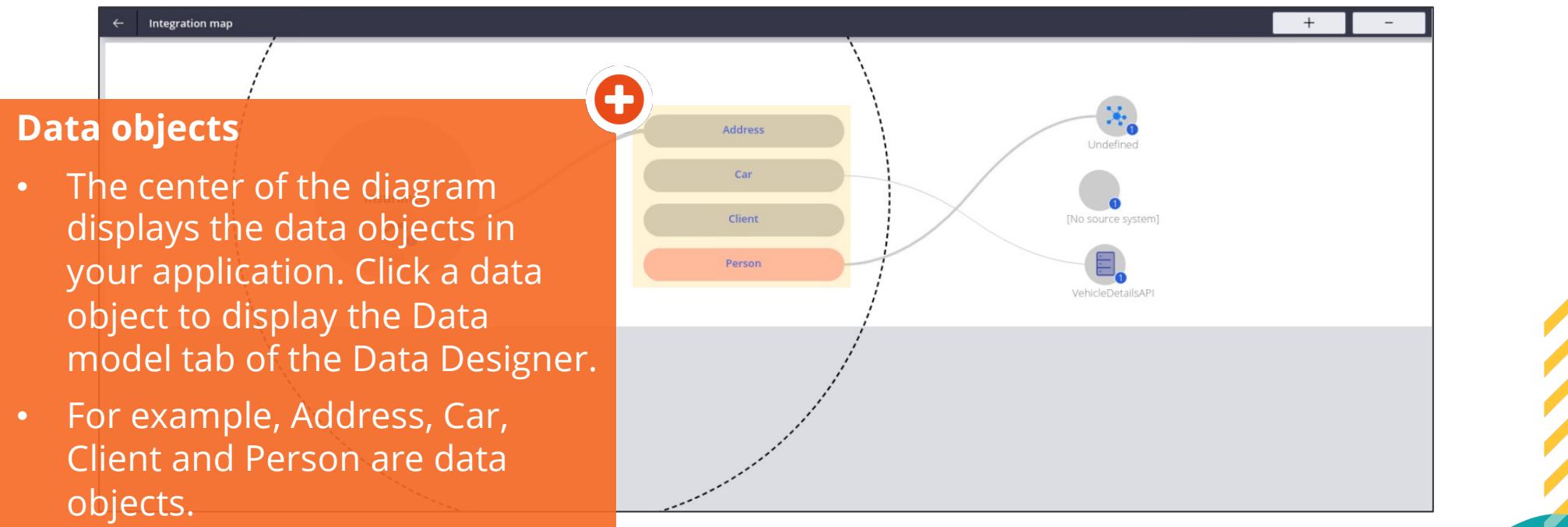
Application and internal system of record

Implementation



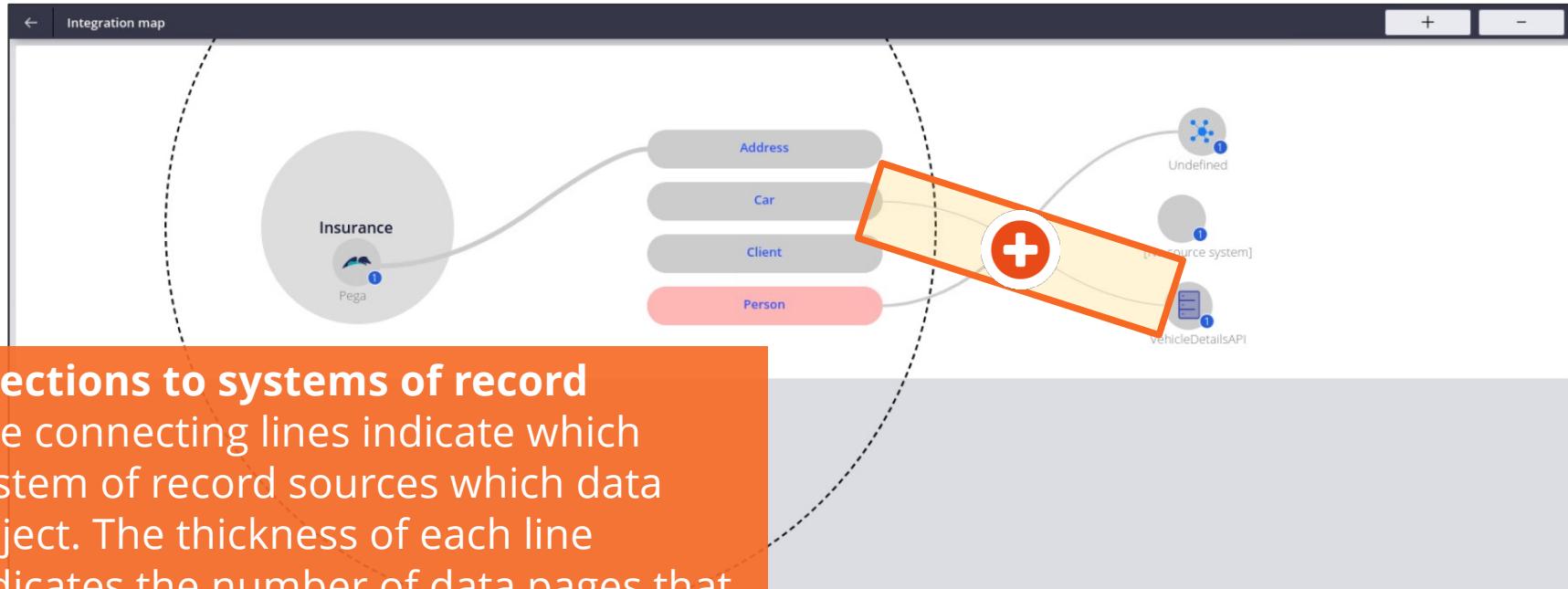
Data objects

Implementation



Connections to systems of record

Implementation

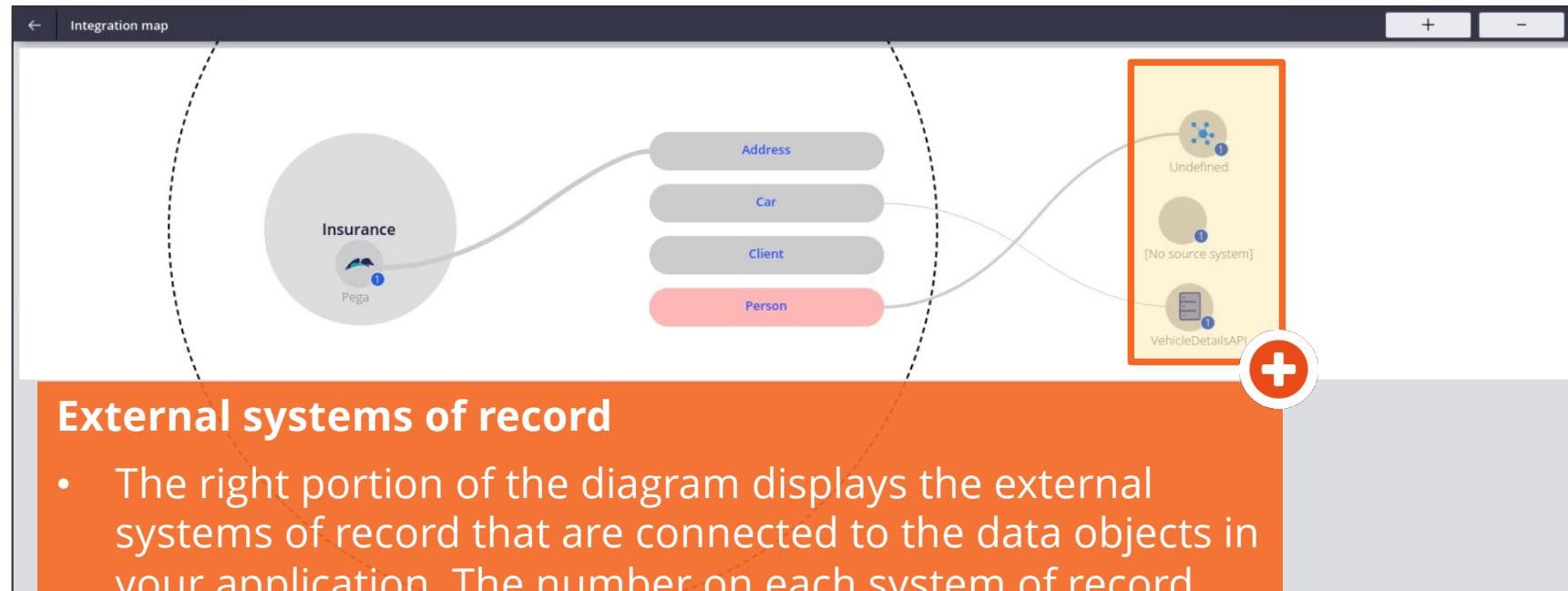


Connections to systems of record

- The connecting lines indicate which system of record sources which data object. The thickness of each line indicates the number of data pages that are sourced from that system of record.

External systems of record

Implementation



External systems of record

- The right portion of the diagram displays the external systems of record that are connected to the data objects in your application. The number on each system of record indicates the number of data objects sourced from that system of record.
- For example, the Car data object references the VehicleDetailsAPI external system of record.

Simulated data views

Implementation

Simulated Data Views

- When a Data Page uses a simulated data source, the Data Object will have a red background. The Data Page with simulated data source will be indicated as well.
- For example, the List Person by Person ID data view for the Person data object uses a simulated data source.



Data objects

Implementation

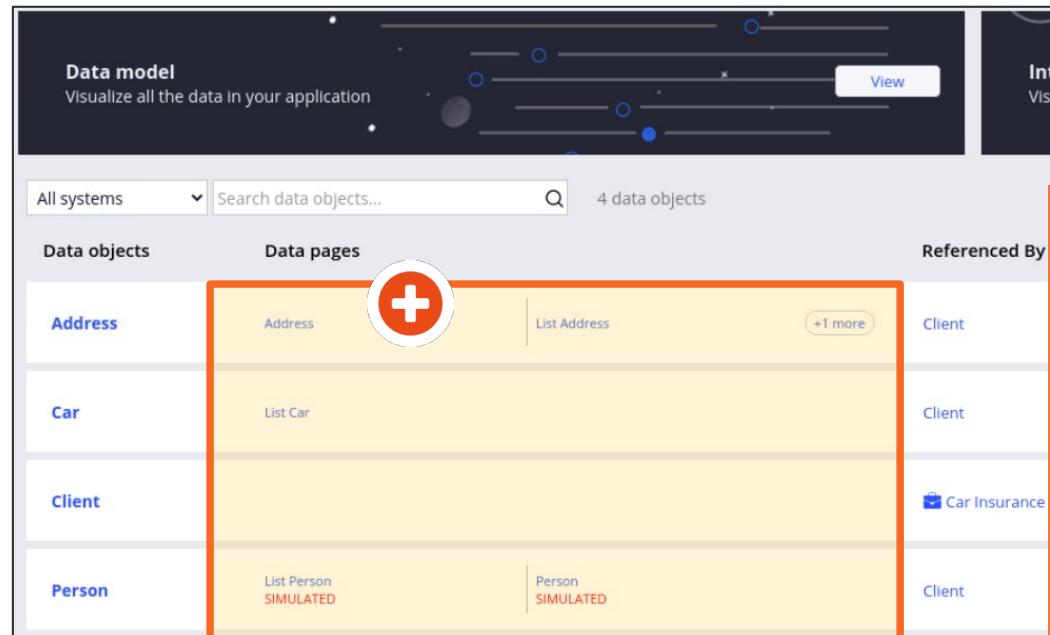
The screenshot shows the Pega Data Designer interface. At the top, there are two cards: "Data model" (Visualize all the data in your application) and "Integration map" (Visualize where data is coming from). Below these are search filters for "All systems" and a search bar. The main area is titled "Data objects" and displays a list of four objects: Address, Car, Client, and Person. A red callout box highlights the "Data objects" section, which contains the following text:

Data objects

- Each row represents a data object, also referred to as a data type, that defines and holds data for the application. Click a **data object** to display the Data model tab of the Data Designer.
- For example, Address, Car, Client and Person.

Data Pages/Views

Implementation



The screenshot shows the Pega Data Designer interface. At the top, there are two cards: "Data model" (Visualize all the data in your application) and "Integration map" (Visualize where data is coming from). Below these are navigation tabs for "All systems" and a search bar. The main area is divided into "Data objects" and "Data pages". The "Data objects" tab lists "Address", "Car", "Client", and "Person". The "Data pages" tab shows three entries: "Address" (with a red circle and plus sign icon), "List Car", and "List Person SIMULATED". The "Address" entry is highlighted with an orange border. To the right, a sidebar titled "Referenced By" lists "Client", "Client", "Car Insurance", and "Client".

Data views (are Data Pages)

- Each data object may have one or more **data pages**. Each data page represents an instance of how your application is using the data. Click a data view to display the Data pages tab of the Data Designer.
- For example, the Address data object is used in two data pages.

Referenced by

Implementation

The screenshot shows the Pega Data Model interface. At the top, there are two cards: 'Data model' (Visualize all the data in your application) and 'Integration map' (Visualize where data is coming from). Below these is a search bar with 'All systems' dropdown and a 'Data objects' tab selected. The main area has a title 'Referenced by' with a red callout box containing the following bullet points:

- The **Referenced By** column displays the case types and data objects that reference the data object.
- For example, the Client reference the Car Insurance.

On the right, there is a 'Systems of record' section listing three entries: 'Pega' (with a person icon), 'VehicleDetailsAPI' (with a database icon), and 'Undefined' (with a blue square icon). A yellow diagonal bar is visible on the right side of the slide.

System of record

Implementation

The screenshot shows the Pega Integration Designer interface. At the top, there are two cards: "Data model" (Visualize all the data in your application) and "Integration map" (Visualize where data is coming from). Below these is a search bar with "All systems" selected and a result count of "4 data objects". A large orange callout box highlights the "Systems of record" section, which lists three entries: "Pega" (with a book icon), "VehicleDetailsAPI" (with a database icon), and "Undefined" (with a blue hexagonal icon). A red box and a red circle with a white plus sign are overlaid on the "Undefined" entry, indicating it is a new or pending addition.

- The **Systems of record** column shows the system or systems of record the data object uses. A system of record is an internal or external system that sources the data. The systems of record referenced in the application are also displayed at the top of the Integration Designer landing page.
- For example, the VehicleDetailsAPI system of record sources the Car data object.

Action menu

Implementation

Data model
Visualize all the data in your application

Integration map
Visualize where data is coming from

All systems Search data objects...

Data objects	Data pages	Referenced By	Systems of record
Address	Address	List Address <small>+1 more</small> Client	Pega
Car	List Car	Client	VehicleDetailsAPI
Client		Car Insurance	
Person	List Person SIMULATED	Person SIMULATED	Undefined

Actions menu
Delete data objects from the Actions menu.

Visual data model

Implementation

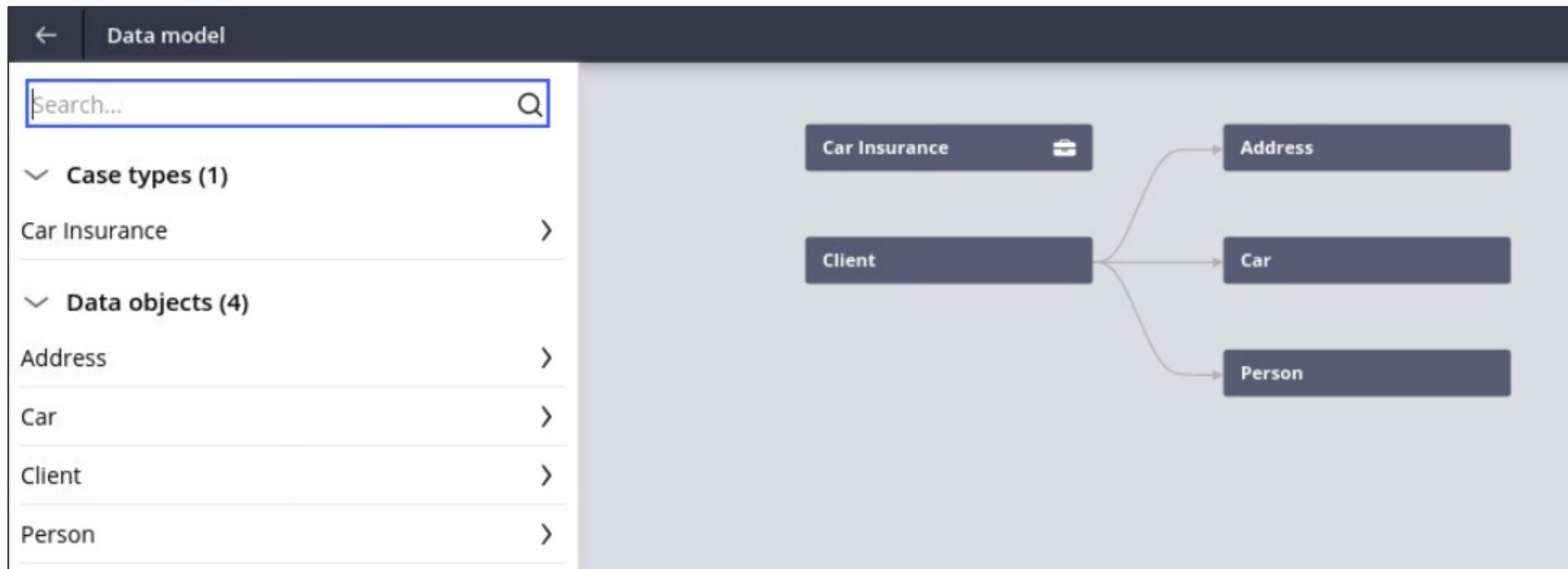
Visual Data Model

- Access the **Visual Data Model** from the **View data model** link.

Address	Description	Referenced By	Systems of record
Car	List Car	Client	Pega
Client		Car Insurance	VehicleDetailsAPI
Person	List Person SIMULATED	Client	Undefined

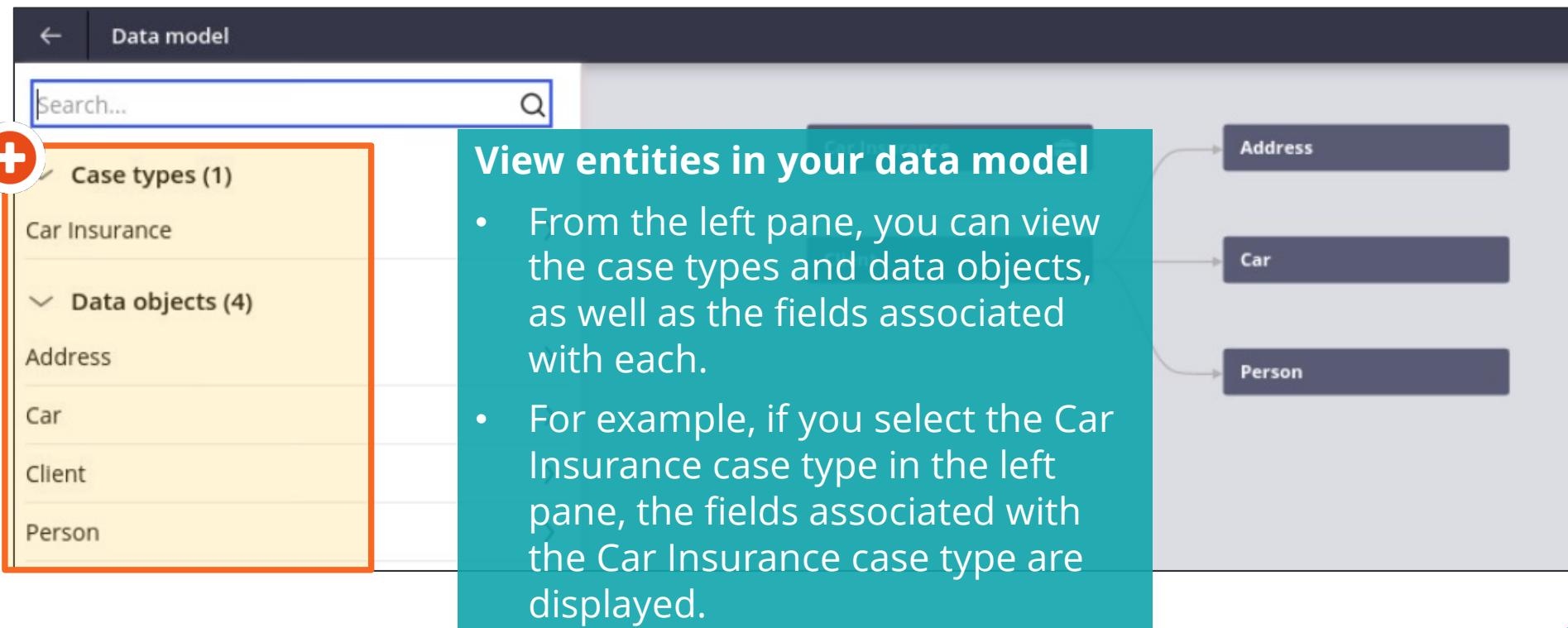
Search data model

Implementation



View entities in your data model

Implementation



The screenshot shows the Pega Data Model interface. On the left, there's a sidebar with a search bar at the top. Below it, a red box highlights the 'Case types (1)' section, which contains 'Car Insurance'. Under 'Data objects (4)', there are four items: 'Address', 'Car', 'Client', and 'Person'. To the right, a large teal callout box contains the title 'View entities in your data model' and two bullet points. The first point says: 'From the left pane, you can view the case types and data objects, as well as the fields associated with each.' The second point says: 'For example, if you select the Car Insurance case type in the left pane, the fields associated with the Car Insurance case type are displayed.' To the right of the callout, three dark grey rectangular boxes are connected by arrows: 'Address', 'Car', and 'Person'.

View entities in your data model

- From the left pane, you can view the case types and data objects, as well as the fields associated with each.
- For example, if you select the Car Insurance case type in the left pane, the fields associated with the Car Insurance case type are displayed.

Open the data model

Implementation

Open the data model

- Click **Open** to open the data model for a case or data type.
- For example, if the Car Insurance case type is selected, clicking **Open** displays the Data model tab of the Car Insurance case type.

Fields (13)	
Active Channel	Text (single line)
Case ID	Text (single line)
Case status	Text (single line)
Client	Embedded object
Commit DateTime	Date & time
Create Date/Time	Date & time
Create Operator Name	Text (single line)
Label	Text (single line)
Native Social Channel	Text (single line)
Save DateTime	Date & time
Update Date/Time	Date & time
Update Operator Name	Text (single line)

Add a field

Implementation

The screenshot shows the 'Fields (13)' section for the 'Car Insurance' case type. A red circle highlights the yellow '+' icon in the top right corner of the list header. An orange callout box contains the following text:

Add a field

- Click the **Plus** icon to add a field or data reference to a case type or data object.
- For example, if the Car Insurance case type is selected, clicking the Plus icon adds a field to the Car Insurance case type.

A diagram on the right illustrates the relationship between the 'Client' data object and other entities: 'Address', 'Car', and 'Person'. Arrows point from 'Client' to each of these three entities.

Field Name	Type
Active Channel	Text (single line)
Case ID	Text (single line)
Case status	Text (single line)
Client	Embedded data
Commit DateTime	Date & time
Create Date/Time	Date & time
Create Operator Name	Text (single line)
Label	Text (single line)
Native Social Channel	Text (single line)
Save DateTime	Date & time
Update Date/Time	Date & time
Update Operator Name	Text (single line)

Edit or delete a field

Implementation

The screenshot shows the 'Fields (13)' list for the 'Car Insurance' entity. A tooltip box is overlaid on the 'Client' field, which has a gear icon (for edit) and a trash icon (for delete). The tooltip contains the following text:

Edit or delete a field

- Hover over a field and click the **Gear** icon to edit a field. Click the **Trash** icon to delete a field. The icons are grayed out if the option is unavailable. For example, you cannot edit or delete system fields like Case ID and Case status.

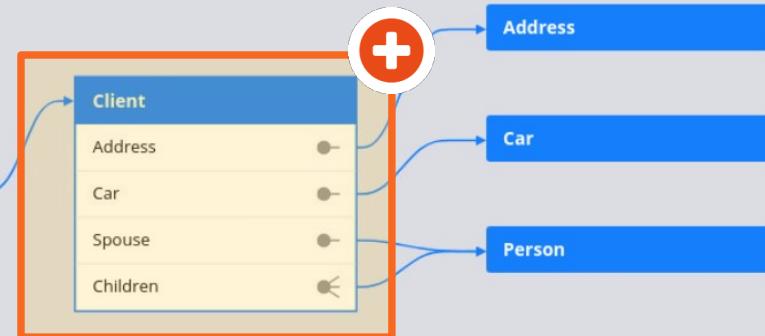
Field Name	Type
Active Channel	Text (single line)
Case ID	Text (single line)
Case status	Text (single line)
Client	Text (single line)
Commit DateTime	Date & time
Create Date/Time	Date & time
Create Operator Name	Text (single line)
Label	Text (single line)
Native Social Channel	Text (single line)
Save DateTime	Date & time
Update Date/Time	Date & time
Update Operator Name	Text (single line)

View case type data objects

Implementation

View case type data objects

- Select a case type in the diagram to view the data objects and case types that the selected case type references. Cases are indicated with a case icon, data relationship are indicated with a single-line icon, and data relationship lists are indicated with a three-line icon.
- For example, the Car Insurance case type references the Client data type.

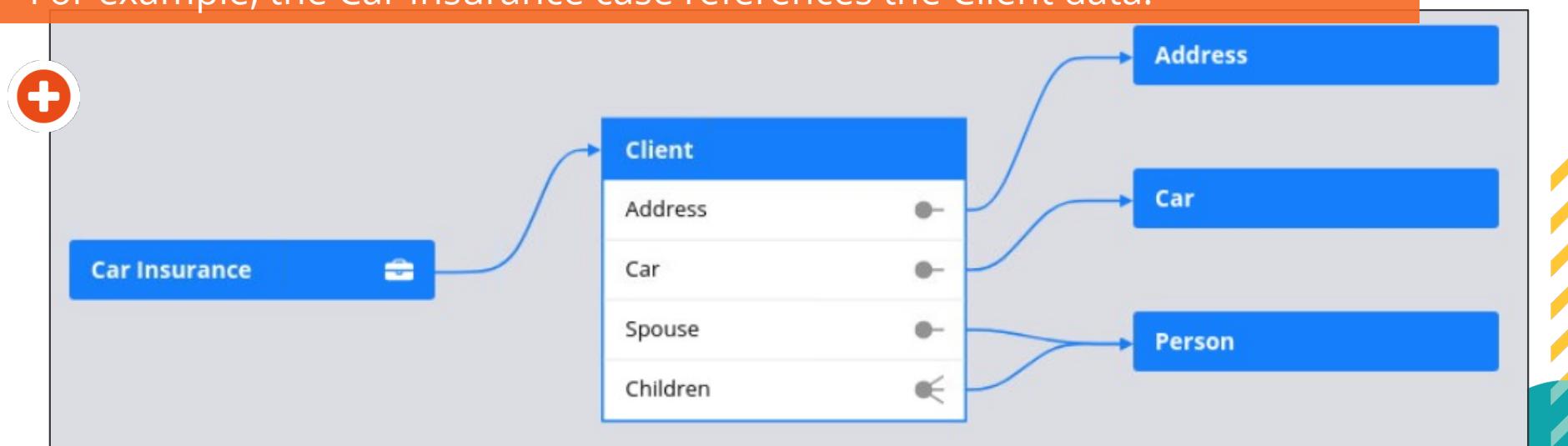


View the relationships among entities

Implementation

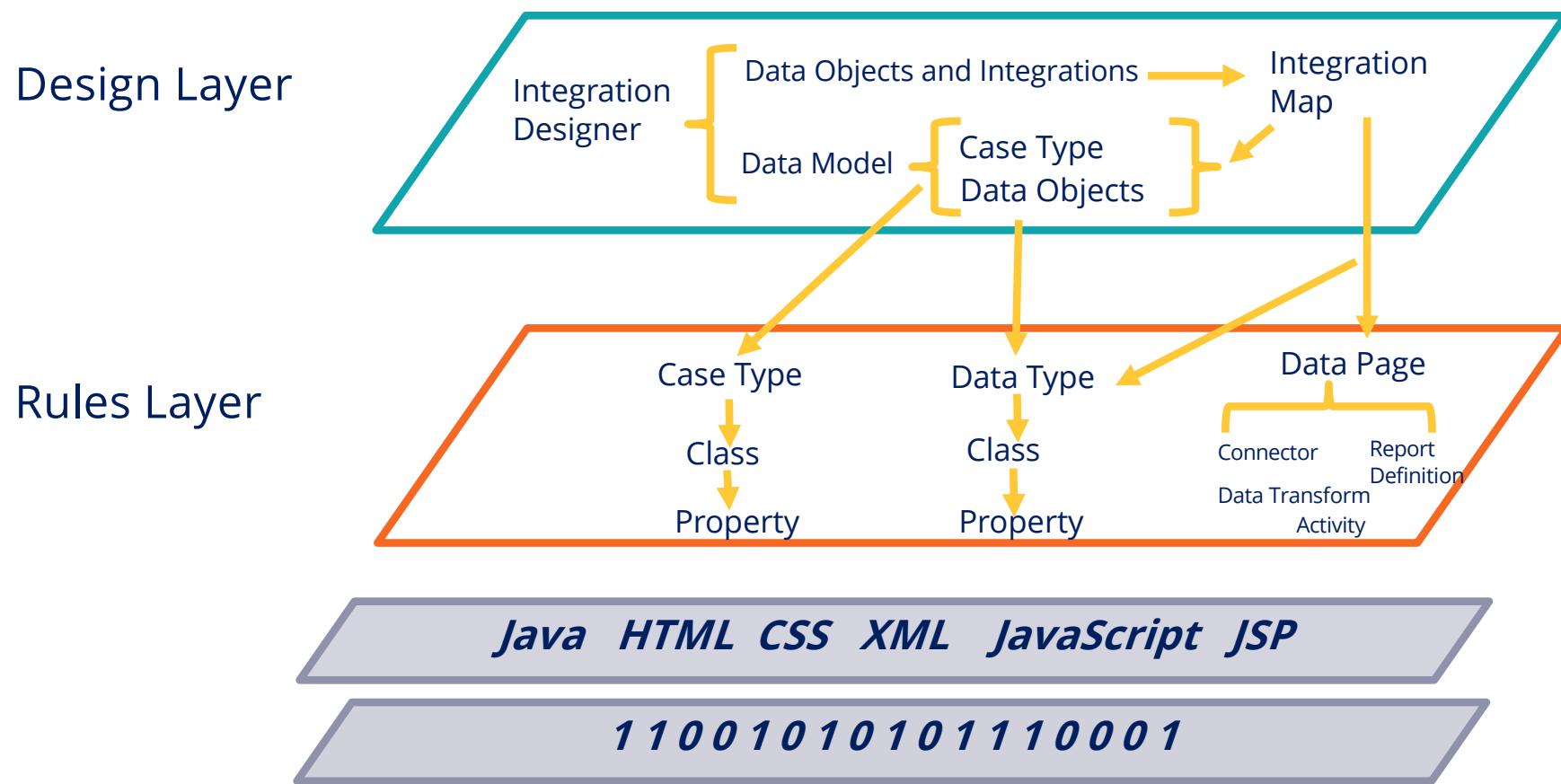
View the relationships among entities

- The diagram displays the relationships among the case types and data objects in your application using arrows. Case types are indicated with a case icon.
- For example, the Car Insurance case references the Client data.





Schematics



Best Practices

- Reuse data objects by promoting a built-on data object or copying the assets from a standard *Data-class*.
- If your data source is unavailable, connect your data objects to simulated data in local storage.
- Repurpose data views instead of creating new ones when your system of record changes.
- Update data sources created in App Studio in Dev Studio only when necessary because some changes made in Dev Studio make the rules un-editable in App Studio.
- Actively remove the data objects and fields that you do not use.
- Provide transformations to automate post-import changes to your data.
- Maintain separate environments for development, staging, production, and testing.
- Save import settings as a template to avoid manually mapping fields during future data imports.

Skill Mastery

Understand:

- the data model
- the relationships between data objects, cases, and systems of record
- how to view the data model
- the Integration Designer
- how to view the integration landscape
- navigating the Integration Designer



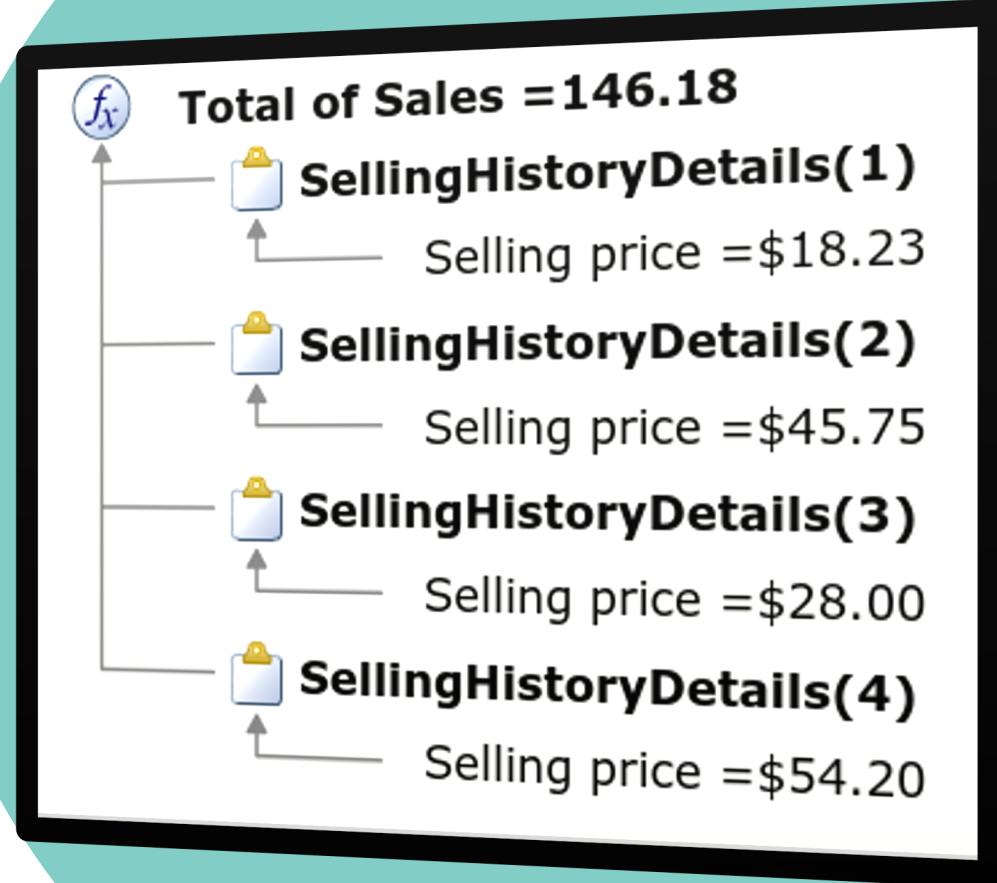
SKILL
LESSON

Declare Expressions



Overview

Use Declare Expressions to define automatic computations of property values based on expressions.





Declare expression

Definition

- A rule used to establish required relationships among properties.
- The rule form is used to build expressions and establish any necessary pages & classes.
 - Do not confuse Declare Expression rules with simple expressions.
 - Declare Expression rules create expressions that are evaluated automatically.
 - Expressions are a syntax used in the Declare Expression rules.

Create Declare Expression

Declare Expression Record Configuration

Target Property * 2

Page Context 3

Context

Development branch
DeclareExp

DeclareExpression

Apply to * 1

WIND-Auto-Work-EvaluateAndSellAVehicle

Add to ruleset * 4

DeclareExpression

Declare Expression

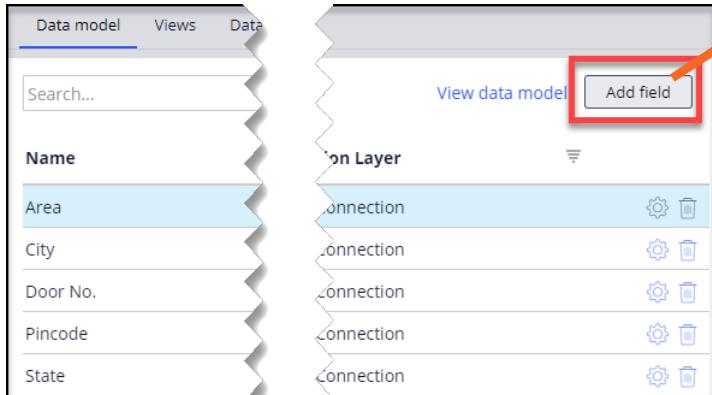
Navigation (1 of 2)

From App Studio:

Data Model > Add field > **Advanced** section

- Select the check box to indicate it's a calculated field.
- Use Expression or a Decision Table from the calculation drop down. Click Submit.
- Verify the Declare Expression rule in the Dev Studio

App Studio > Data Model > Add field



Add field to Current Address

Field name *****
Test

Type
Text (single line)

Advanced

ID *****
Test

Description

Max length
256

Expected length

This is a calculated field (read-only)

Calculation
Use expression

Area + " " + .Pincode

Use '.' (dot) for field prompts to enter a simple equation such as:.Amount*.Quantity

Cancel Submit & add another Submit

A screenshot of a modal dialog titled 'Add field to Current Address'. It contains fields for 'Field name' (Test), 'Type' (Text (single line)), and an 'Advanced' section. In the 'Advanced' section, there are fields for 'ID' (Test), 'Description', 'Max length' (256), and 'Expected length'. A checkbox labeled 'This is a calculated field (read-only)' is checked. The 'Calculation' dropdown is set to 'Use expression', and the expression 'Area + " " + .Pincode' is entered. At the bottom, there are 'Cancel', 'Submit & add another', and a blue 'Submit' button.

Declare Expression

Navigation (2 of 2)

- When a property is created as a calculated field in the App Studio, Pega automatically creates a Declare Expression in the Dev Studio.
- Switch to Dev Studio to view the Declare Expression in the Data class.

The screenshot shows the Pega Dev Studio interface with the following details:

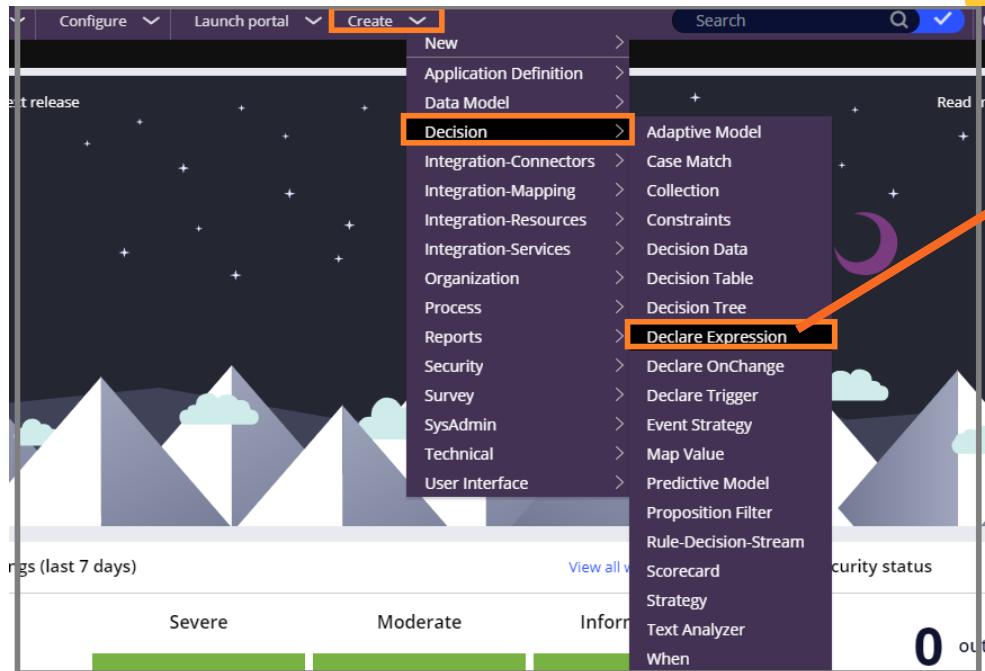
- Top Bar:** Application: ServiceLevelAgreement, Configure, Launch portal, Create, Search, and DEVELOPMENT mode.
- Sidebar:** Shows recent items, case types, data types, and favorites. The "App" section is selected.
- Central Area:** The title is "Declare Expression: .Test [Available]".
 - CL:** WIND-Mobile_C-Data-CurrentAddress
 - ID:** .Test
 - RS:** ServiceLevelAgreementC [Branch: ServiceLevelCLab]
- Actions:** Save as, Delete, Actions, and Check out.
- Content:** Expressions tab is selected. It shows an "Overview" section with "Build Expressions" and "No conditions defined". A "Set Test" dropdown is set to "Value of .Area + " " + .Pincode".

Declare Expressions

Navigation

From the Dev Studio:

Create menu > Decision > Declare Expression



The screenshot shows the 'Create Declare Expression' dialog box. It has tabs for 'Record Configuration' (selected) and 'Script Configuration'. Under 'Record Configuration', there are fields for 'Target Property*' (with a dropdown for 'Page Context'), 'Context' (set to 'Development branch: ServiceLevelCLab'), and 'Apply to*' (radio button selected for 'ServiceLevelAgreement', with other options like 'Mobile_Connection', 'Cosmos Rules', and 'Pega Platform' available). There's also a 'View all...' link and an 'Add to ruleset*' dropdown set to 'ServiceLevelAgreementC'. The 'Script Configuration' tab contains sections for 'Current work item' and 'Work item to associate' (with a 'Select...' button).



Expressions

Description

- A single text element that produces a string value.
 - Looks like formulas in Microsoft Excel but are based on Java language conventions.
 - Can include constants, property references, operators for arithmetic and logical operations, parentheses to control the order of evaluation, and functions.
 - Can perform operations on data including performing mathematical operations, comparing property values, manipulating text, transforming data and converting data types (casting).
- Use conditional logic to determine which calculation takes place. A default expression must always exist which is applied if none of the conditions evaluate to true.

The screenshot shows the 'Expressions' tab selected in a software interface. Under the 'Overview' section, there is a 'Build Expressions' panel. The first condition is an IF statement: 'IF .VehicleDetails.FuelType = "Electric"'. The then part of the IF statement is 'Then set Total Cost = Value of .ItemCost1 * 2'. Below this, there is a '+ Add condition' button and another row for setting 'Set Total Cost = Value of .ItemCost1 + .ItemCost2 + .ItemCost3'.



Build expressions

Navigation

From Dev studio:

1. Create a new Declare expression.
2. Select a function from the expression selector drop-down list.
3. Optionally add a condition to determine when the expression will execute.
4. Set the expression to evaluate.
 - Optionally use the expression builder for guided help on composing an expression.

Edit Declare Expression: .TotalCost [Available]
CL: WIND-Auto-Work-EvaluateAndSellAVehicle ID: .TotalCost RS: Auto [Branch: TestB02]

Expressions Pages & Classes Test cases Specifications History

> Overview

Build Expressions

No conditions defined

+ Add condition

Set Total Cost = Value of .ItemCost1 + .ItemCost2 + .ItemCost3

Value of

- Value of
- Sum of
- Minimum of
- Maximum of
- Average of
- Value of first matching property in parent page
- Count of
- Index for Minimum of
- Index for Maximum of
- Result of Decision Tree
- Result of Decision Table
- Result of Map Value



Using the Expression Builder

Navigation

- To the right of the expression select the gear icon to open the expression builder.
 - This allows more complex expressions to be created as well as access to function libraries.

The screenshot illustrates the process of opening the Expression Builder. On the left, a screenshot of the 'Edit Declare Expression: TotalCost [Available]' screen shows a dropdown menu with 'Value of' selected. To its right, a red box highlights a gear icon (representing the Expression Builder) inside a blue-bordered box. An orange arrow points from this box to a larger screenshot on the right. This second screenshot shows the 'Expression builder' window open, displaying a search bar, a 'Functions' section, and a 'Properties' section. At the bottom of the builder window are 'Cancel' and 'Submit' buttons. The background of the slide features a yellow decorative pattern in the bottom right corner.

Edit Declare Expression: TotalCost [Available]
CL: WIND-Auto-Work-EvaluateAndSellAVehicle ID: .TotalCost RS: Auto [Branch: TestB02]

Expressions Pages & Classes Test cases Specifications History

> Overview

Build Expressions

No conditions defined

+ Add condition

Set Total Cost = Value of .ItemCost1 + .ItemCost2 + .ItemCost3

Value of

- Sum of
- Minimum of
- Maximum of
- Average of
- Value of first matching property in parent pages
- Count of
- Index for Minimum of
- Index for Maximum of
- Result of Decision Tree
- Result of Decision Table
- Result of Map Value

Expression builder

Search

Functions

Properties

Cancel

Submit

© 2022 Pegasystems Classroom Experience Training Materials

10

Calculations

Definition

- Expresses a relationship between fields by setting the value of a calculated field based on one or more input fields.
- Occurs whenever users update the value of an input field.
- Reduces the chance for error and improves process efficiency by eliminating the need for manual calculation.
- Can be used as an input to another calculation, creating a network of dependent calculations.
 - When a user updates the value of a field that serves as a calculation input, the resulting calculation can trigger additional calculations for dependent fields.

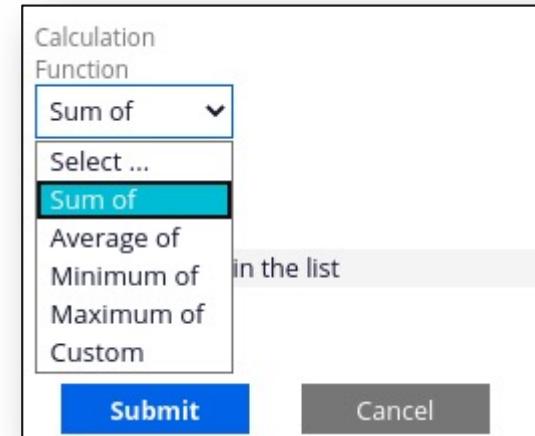
Item Total = Quantity * Unit Price



Calculation types

Description

- App Studio allows two types of calculation, functions and custom calculations.
- **Functions** iterate over items in a multiple source data relationship.
 - *Sum of, Average of, Maximum of, Minimum of*
- **Custom calculations** can reference any combination of simple and fancy fields.
 - Reference fields by name using the period character (".".).
 - If the field is part of a data relationship, add the data relationship name as a prefix.
 - Custom calculations support common operations, such as addition, subtraction, multiplication, division, grouping (using parenthesis), Boolean AND, Boolean OR, concatenation.



Declarative network analysis

Description

- To identify the relationships between fields, the Pega Platform establishes and updates a network of calculations for an application.
- When you define a field calculation, the Pega Platform adds that calculation to the calculation network.
- This calculation network allows the Pega Platform to update all relevant fields whenever a value changes.
- Shows the target property and all potential inputs that might affect its final value.
- Unit test a Declare Expression rule to reduce the number of processing errors.

Users add an item to the list or update the quantity of an item already in the list. Pega Platform calculates the **Line total as .UnitPrice * .Quantity**.

The updated line total triggers a calculation to update the **order total**, using the **Sum of function** to add each line total in the list of items.

You added the following items to your order:

Item	Quantity	Unit price	Line total
Team logo hat	3	USD 7.75	USD 23.25
Team logo magnet	4	USD 4.25	USD 17.00

Order total: USD 40.25

Tax (8%): USD 3.22

Your cost: **USD 43.47**

The updated order total triggers a calculation to update the **tax** applied to the order as the **value of .TaxRate * .OrderTotal**.

The updated tax amount triggers a calculation to update the **total cost** as the **value of .Tax + .OrderTotal**.



Declarative network display

Navigation

From Dev Studio

1. Configure menu > Select Case Management > Business Rules > Declarative Network.
2. Click application to select the app for review.
 - o To view the rule and related properties, click the "Display this top-level declarative network" icon.
 - o Open the declare expression, click the Display declare expression icon.
 - o Open the target property, click the property name.
 - o Open the class to which a rule belongs, click the class name.

Case Management: Business Rules

Declarative Network Analysis

1 application

- WIND-Auto-Work-EvaluateAndSellAVehicle Evaluate and Sell a Vehicle (3 Networks)
 - Calculate Value : Whenever inputs change
 - Calculate Value : Whenever inputs change
 - .UserAgreement Calculate Value : Whenever used
- WIND-Auto-Work-RVCalculation RV Calculation (1 Networks)
 - .RoundedResidualValue Calculate Value : Whenever inputs change



Declare Expression

Unit test and debug

- A rule can be unit tested by selecting **Actions > Run**
 - Enter values for the input properties to see the target automatically updated

The screenshot shows the Pega Test Page interface. At the top, there are settings for 'Data Context' (set to 'STANDARD') and a radio button for 'Create or reset Test Page' (which is selected). Below this, a message says 'Test Page temp_HireMe_ApplicantMgr_Work_JobApplicant created.' There is a 'Reset Page' button. The main area displays a tree structure under the heading 'HireMe-ApplicantMgr-Work-JobApplicant AverageRating'. The tree shows:

- (radio button) Average Rating =
- Number of covered objects = 0
- Total Rating =

On the right side, there is an 'Options' panel with checkboxes for 'Show description' (checked), 'Nodes expand/collapse' (+/-), and 'Help' (with a question mark icon). Below the options is a 'Convert To Test' button.



Schematic

Design Layer

Calculated value

Rules Layer

Declare expression → Property

Java HTML CSS XML Javascript JSP

11001010101110001

Best Practices

- Because declarative expressions extend the definition of a property value, use nouns in the declarative expression names for better distinction from property values.
- Do not change the value of a property computed by a Declare Expression rule through the Property-Set method, application of a data transform, or user input into a form. As you develop your application, the Pega Platform attempts to detect and prohibit such situations.
- To reduce the risk of similar rule conflicts, adopt a naming convention for properties designed to be computed in Declare Expression rules, and create Declare Expression rules early, using stub or dummy expressions.

Skill Mastery

Understand:

- What are declare expressions.
- How to use and create expressions.
- What are the calculation types.
- What are functions and custom calculations.
- What is a calculation network.

Integration



SKILL
LESSON

Managing data objects

Managing data objects with the Integration Designer



Overview

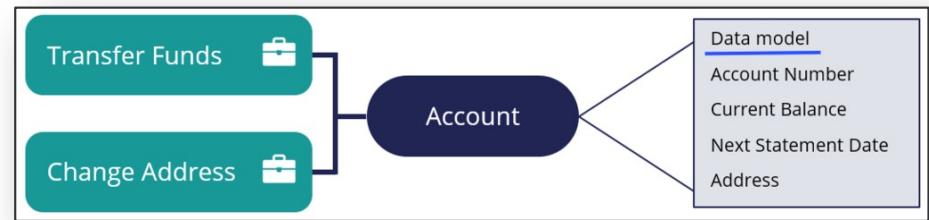
The Integration Designer can be used to manage data objects that collect data while processing cases.

The screenshot shows the Pega Integration Designer's Data Model interface. At the top, there's a toolbar with icons for New, View integration landscape, and other options. Below the toolbar is a header bar with the title "Data model". The main area is a grid-based table with columns for "Data types", "Data views", "Referenced By", and "Systems of record".

Data types	Data views	Referenced By	Systems of record		
Accumulators - Family (Benefits)	Family Accumulators SIMULATED	List Family Accumulators SIMULATED	Medicare Inquiry Policy	Plan benefits	
Accumulators - Individual (Benefits)	Individual accumulators SIMULATED	List Individual Accumulators SIMULATED	Medicare Inquiry Policy	Plan benefits	
Authorizations	Fetch Authorizations By Policy Number SIMULATED	Fetch Member Authorizations SIMULATED	+1 more	Interaction - Inbound phone Appeal a Prior Authorization	Claims
Benefit Plan	List Dental Plans by Product	Get Custom Plan	+9 more	View Member Benefits Provider - Contract	Pega Plan benefits
Claim History	Get Claim History SIMULATED	List Claim History SIMULATED	+5 more	Claims Inquiry Claims Research	Claims
Claims	List Claims SIMULATED	List Claims by Claim ID SIMULATED	+2 more	Claims Research Claims Inquiry	Robotics
COB History	COB History By History ID SIMULATED	List COB History For Member SIMULATED		View Member Benefits Coordination Of Benefits	Membership System
Customer Address	List addresses			Update Contact Profile	Pega

Use Case

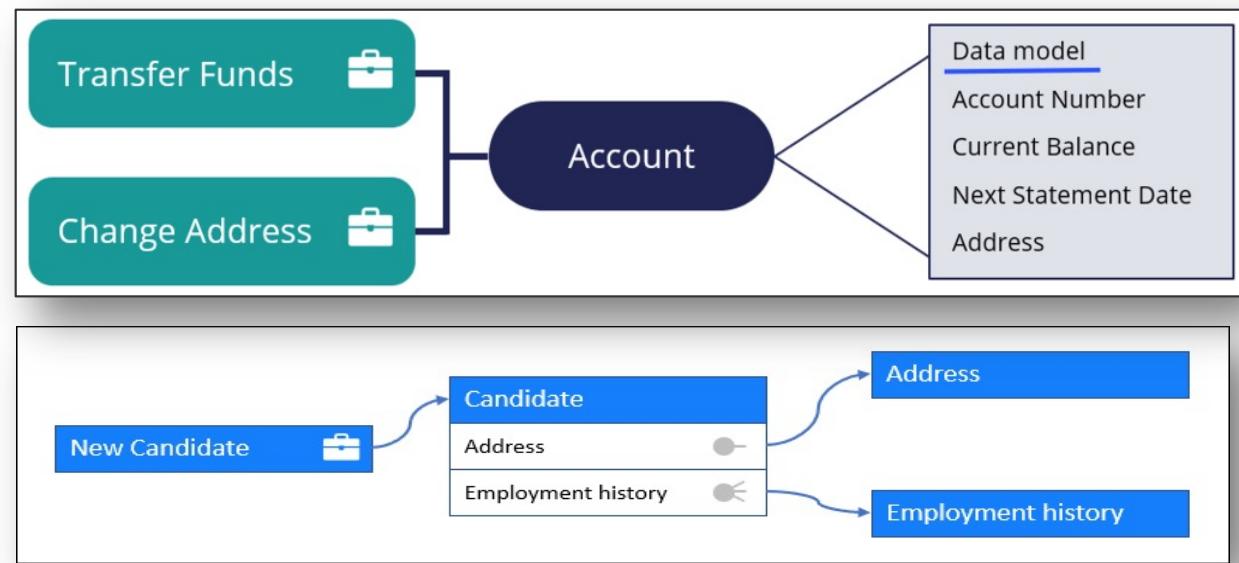
- A data object
 - is created to model an entity such as an Account.
 - is used to collect data during case processing.
 - can be used to store records that define accounts a customer owns.



Data Object

Definition

- A template for describing an entity, such as a person or an item, by grouping a set of related fields.
- The pairing of two pieces of information: the *name* of the data element and the *value* assigned to it.
- In addition to grouping data elements, data objects can group views and other rules related to the data object.
- Extend the structure of a data object by referencing other data objects.
- Reuse a data object as many times as needed in an application.



Data Pages

Definition

- Also known as **Data views**, can be used to interact with a system of record to retrieve or save data.
- Data objects are associated with **Systems of record** through data pages which are sourced in two ways: **local** data source, **external** data source
- Data objects can also have **no** associated system of record which means they are simply used to provide data structures.

The screenshot shows the Pega Data Pages interface. At the top, there are icons for 'SIMULATED' (0), 'Pega' (1), 'Customer' (1), and 'Orders' (1). On the right, there are links for 'View integration landscape' and 'View data model'. A search bar with a magnifying glass icon is also present.

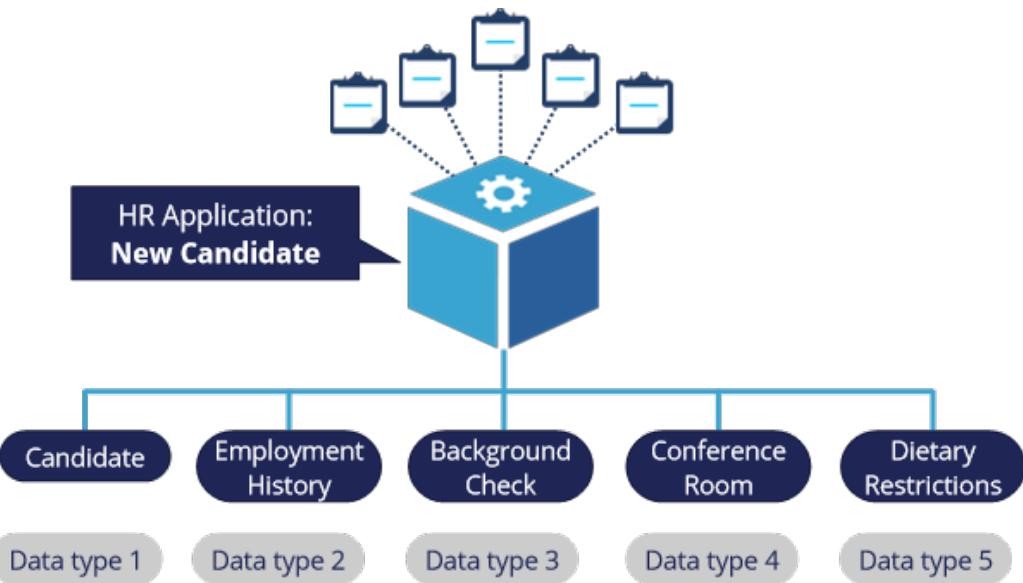
Data objects	Data views	Referenced By	System of record
Customer	List customer address info List customer contact info	Online Order Order Inquiry	Customer
Orders	List orders by Customer ID List orders by Order ID	Online Order Order Inquiry	Orders
Order History	Get order history	Online Inquiry	Pega

At the bottom left is the PEGA logo. The bottom right contains the text '© 2022 Pegasystems Classroom Experience Training Materials' and the number '6'.

Data type

Definition

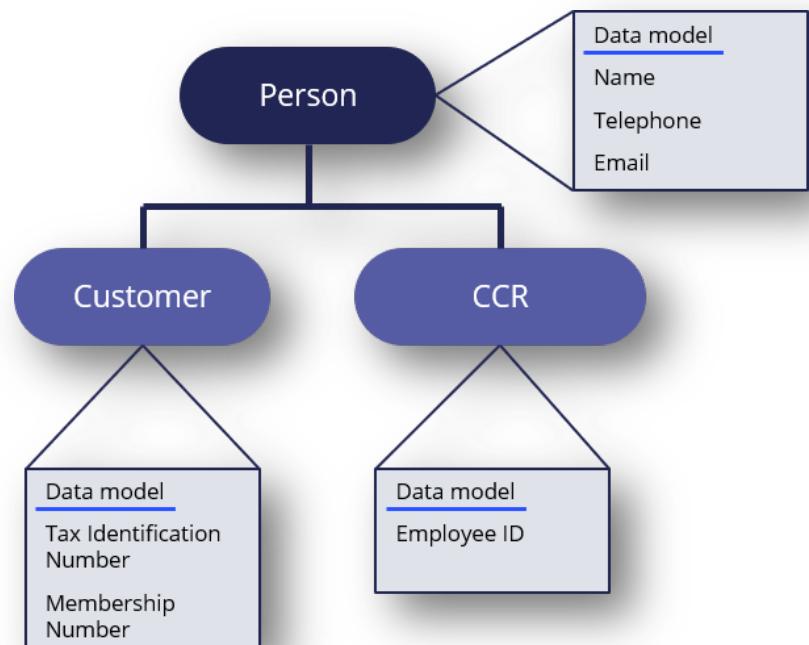
- In Dev studio, a data type is a class created to uniquely define and hold data for your application.
- When creating a data object, the Pega Platform automatically creates the corresponding data type.
- Within each data object, a **data type** represents the technical implementation for the data object, such as the names and types of fields used to capture and present information about the entity.
- Collectively, the different fields represent a single type of object and define the structure of the data object.



Inheritance

Description

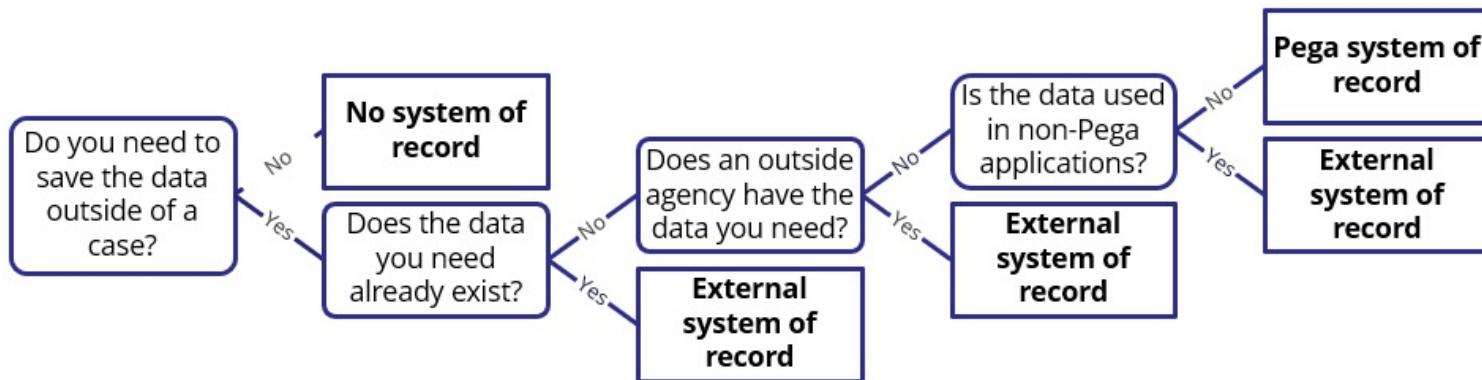
- The reuse of data objects assets from an existing data object.
- For example, *Person* is a generic data object, or a parent data object, while *Customer* and *Call Center Representative (CCR)* are more specialized data objects.
 - Person-Customer and Person-CCR
 - All three data objects have common fields, including *Name*, *Telephone*, and *Email*.
 - By creating the common fields in the Person data object, the fields can be reused in the Customer and Call Center Representatives data objects.
 - The fields *Tax Identification Number* and *Membership Number* apply to customers only, so you define them in the Customer data object.
 - The field *Employee ID* applies to employees only, define the field in the CCR data object.



Sourcing

Description

- Data objects can be sourced locally from a Pega Platform system of record, from an external system of record, or not associated with any system of record.
- Determine how to source a data object by considering the questions in the image.



Does the data need to be saved outside of the case?

- *No, data object with no SOR*

Does the data already exist?

- *Yes, data object with existing external SOR database*

Does an outside agency have the data needed?

- *Yes, data object with external SOR connection*

Is the data used in non-Pega applications?

- *No, data object with local SOR*
- *Yes, data object with external SOR in a third-party application*

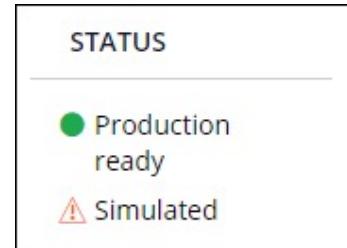
Simulated External Data Source – App Studio

Description

- In **App Studio**, when you create a data view, you identify the source of the data.
- The same data views are created regardless of whether you simulate the source data or connect to a system of record (actual data).

The screenshot shows the Pegasystems App Studio interface. At the top, there's a navigation bar with a back arrow, the title 'Data objects and integrations', and a 'New' button. Below the title, there are two main sections: 'Data model' and 'Integration map'. The 'Data model' section has a heading 'Visualize all the data in your application' and a 'View' button. The 'Integration map' section has a heading 'Visualize where data is coming from' and a 'View' button. Below these sections, there are four tabs: 'Data objects' (selected), 'Data views', 'Referenced By', and 'Systems of record'. Under the 'Data objects' tab, there is a table with one row for 'Vehicle models'. The table has two columns: 'List Vehicle models' (with a red 'SIMULATED' label) and 'Vehicle models' (with a red 'SIMULATED' label). At the bottom of the screen, there is a status bar with the text 'Undefined' and a blue hexagonal icon.

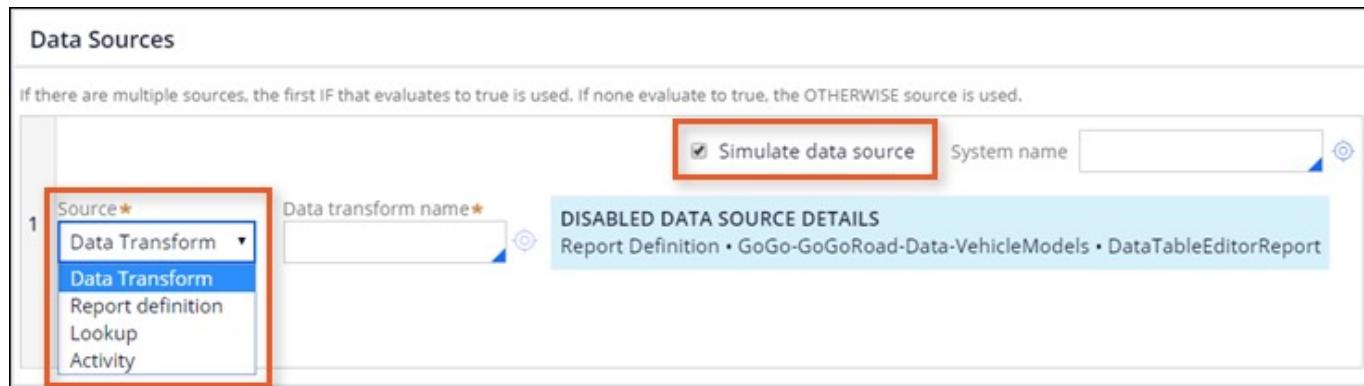
- Use Dev Studio to get an overview of simulated data pages in your application.
- In **Dev Studio**, click **Configure > Data Model > View external data entities** to open the landing page and get an overview of simulated data pages in your application.
- Source systems marked with a green dot are production-ready. Source systems marked with an orange triangle are simulated.



Simulated External Data Source – Dev Studio

Description

- Use Dev Studio to identify the source data directly on the data page.
- You select the **Simulate data source** option on the data page to simulate the data source.
- Then, enter or select the name of the system to use for simulated data.
- You can simulate a data page by using a data transform, activity, report definition, or lookup. Selecting simulation disables any data source configured.

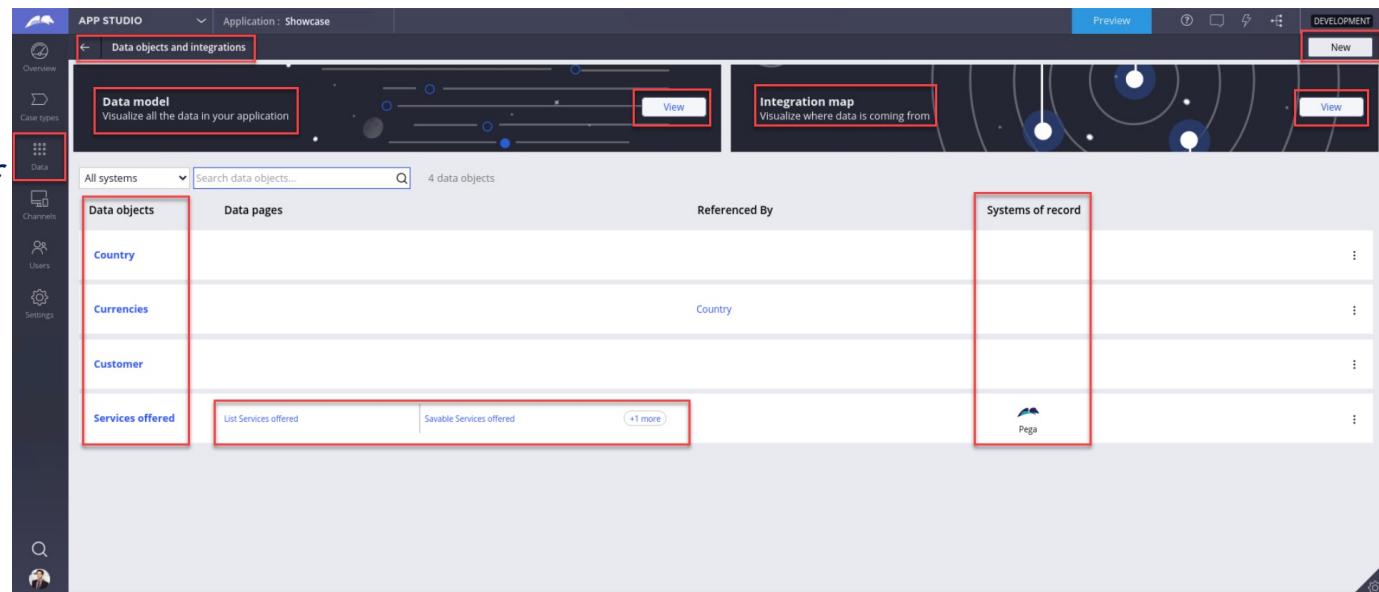


Data model and integration map

Navigation

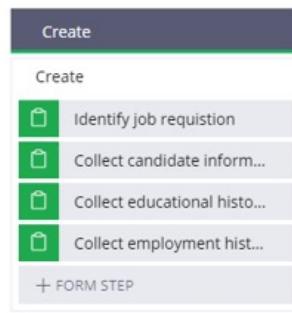
From App Studio:

1. Click the Data explorer
 - o *Defined Data objects display here with associated data pages, references and systems of record.*
2. Click the view button next to Data model to review.
3. Click the view button next to Integration map to review.
4. Navigate back to this screen by selecting the arrow in the top left.



Data object and data type relationship

Implementation



Data usage in the case type

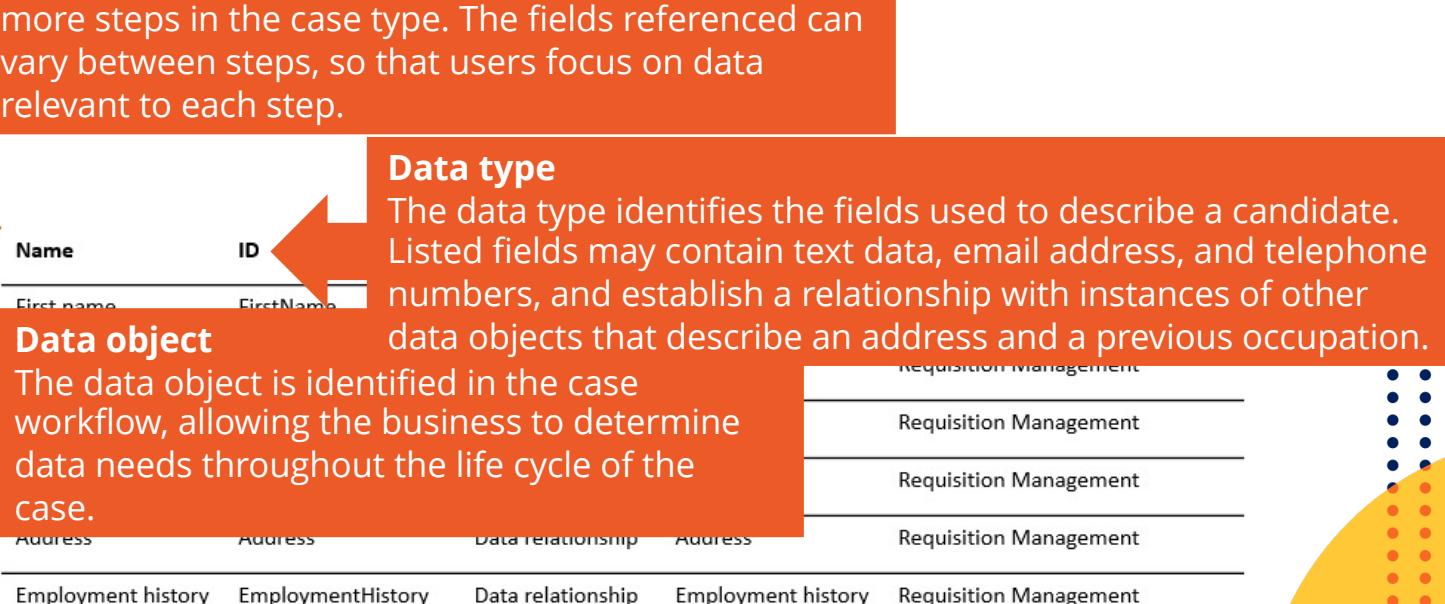
An instance of the data object is referenced in one or more steps in the case type. The fields referenced can vary between steps, so that users focus on data relevant to each step.

Data type

The data type identifies the fields used to describe a candidate. Listed fields may contain text data, email address, and telephone numbers, and establish a relationship with instances of other data objects that describe an address and a previous occupation.

Data object

The data object is identified in the case workflow, allowing the business to determine data needs throughout the life cycle of the case.



Field types and data object location

Implementation

Data relationship field type	Data Source	Use Case
Embedded data	User-supplied data such as a name and address sourced from inside a case type.	A company needs to capture shipping addresses.
Query	A data page or view that is not sourced from inside the case type. The data page defines that the Query data relationship is configured to use.	An application needs to update the current weather.
Case reference	Single or multiple records from a selected case type.	A user selects from a list of service cases from the Service Case type.
Data reference	Single or multiple records from a selected data page.	A user selects from a list of products to order.



Schematics

Design Layer

- Data object → Data page → System of record
- REST Response

Rules Layer

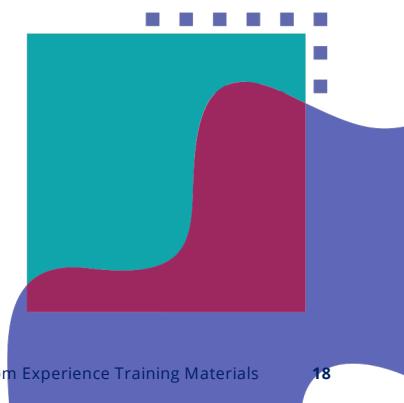
- Data class → Data page → Data sources
 - Connector
 - Data Transform
 - Report Definition
 - Activity
- Property

Java HTML CSS XML JavaScript JSP

11001010101110001

Best Practices

- Whenever possible, use standard, commonly-used data objects that Pega Platform provides, such as Address-Postal and Address-Email.
- Extend data objects that only partially meets requirements.
- Create a new one data object when an existing data object isn't suitable.
- Rename the fields when mapping the REST response fields to the data page.
- Ensure the property name conforms to the Pega naming convention.



Skill Mastery

Understand:

- a data object
- a data type
- the relationship between data objects and data types
- a data page
- data inheritance
- how and why data objects are sourced
- how to use the integration map and data model views

Decision, Approval and Routing



SKILL
LESSON

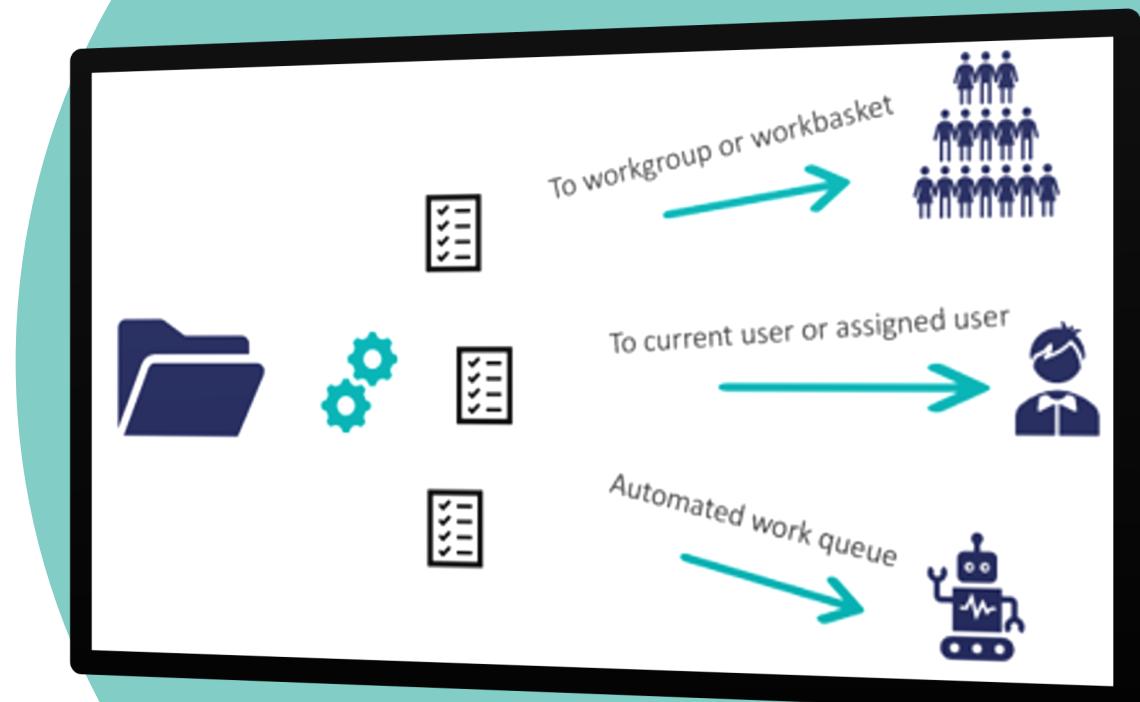
Routing

Who is going to do the work?



Overview

Routing distributes work efficiently within an organization by determining the best user for the task based on skills, availability, and team.



Use Case

- When there is more than one operator to complete work on a case, define who should do the work on each assignment while designing a process.
- Assignments that go to the individual or group of individuals most capable of completing a specific assignment increases business efficiency.



Routing

Definition

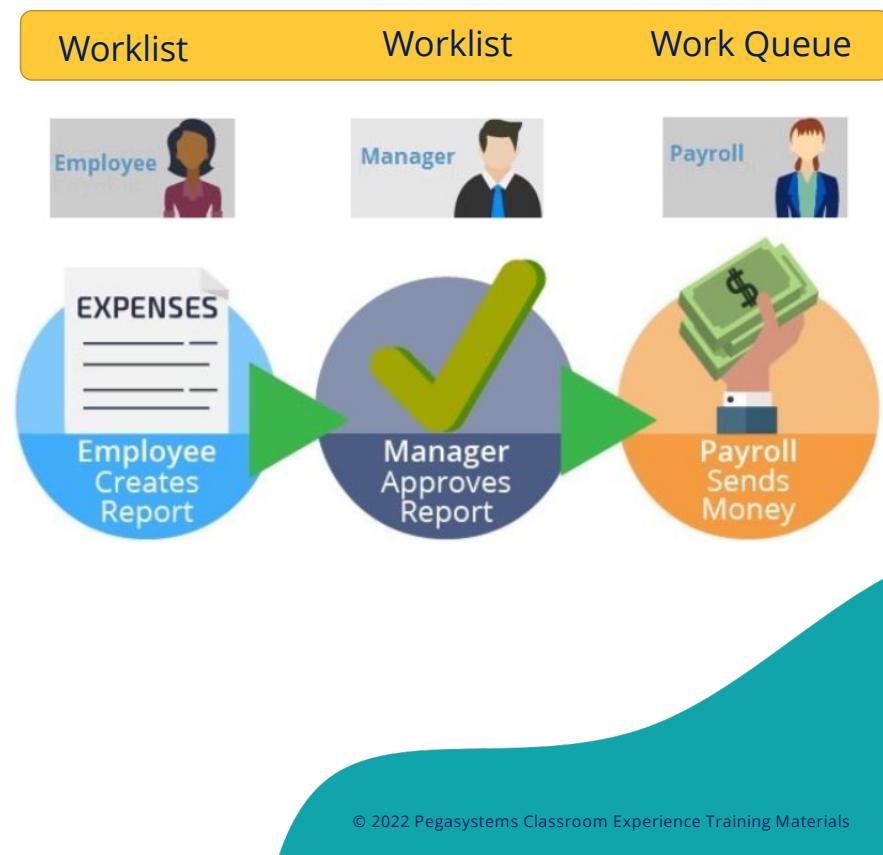
- The process of comparing the characteristics of a new assignment with the characteristics of the workforce to route the assignment to the most appropriate operator.
- The routing algorithm can examine many factors, such as backlogs, the presence or absence of operators, operator skills, the urgency or priority of an assignment, and a customer's location or time zone.

▼ Team members	
 TM	Traffic Dept Manager Traffic Dept Manager 5
 PM	Parks Dept Manager 4
 AM	Admin MyTown System Administrator 3
 MC	Municipal Services Coordinator Municipal Services Coordinator 3
 AM	Author MyTown Application Developer 1

Routing types

Definition and description

- Assign work to the most appropriate user.
- Route a step to a single user or to a team of users using a:
 - Worklist
 - Route an assignment to a specific user responsible for performing selected tasks.
 - Work Queue
 - Route an assignment to a group of users who can complete the task.



Worklist

Definition and description

- Displays a list of all open assignments, in order of importance, for a specific user.
 - Default importance is Urgency.
- Configure routing for the operator using the work tab on the Operator ID.
 - Each operator has an assigned organizational unit.
 - Defining a team is required to save the Operator ID rule so if work needs to be assigned, it can easily be routed using the team configuration.
 - In addition, you can also configure reporting structure, skill and rating, setup a work queue and urgency threshold for the operator.

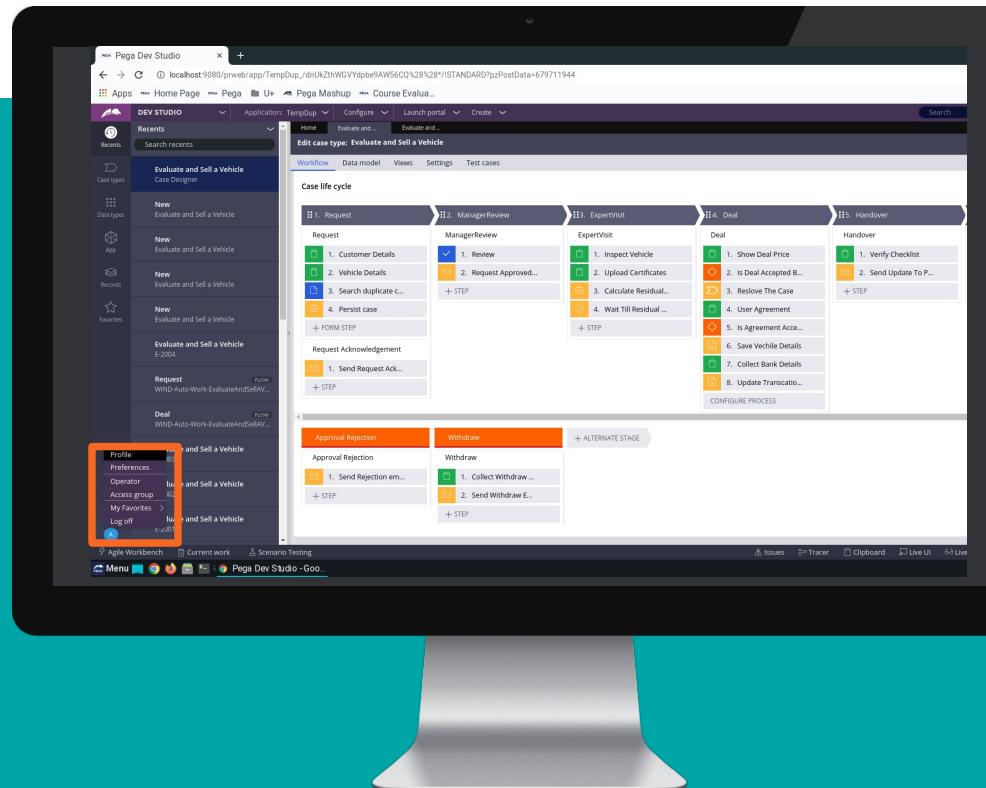
The screenshot shows the 'Edit Operator ID: Bill' screen. At the top, it displays the ID: Bill@TGB RS: TGB [Edit]. Below this, there are tabs for Profile, Work (which is selected and highlighted in blue), Security, and History. The main content area is titled 'Routing'. It includes sections for 'Organizational unit' (set to TGB / Div / Unit with an 'Update' button), 'Work group' (listing 'default@TGB' and 'Payroll@PegaHR' with edit and delete icons), and 'Reports to' (set to Manager@TGB with a 'Reporting structure' button). There are also sections for 'Skill' (French, rating 7) and 'Work queue' (PayrollReviewWB, Urgency Threshold). At the bottom, there are three checkboxes: 'Get from work queues first', 'Use all work queue assignments in user's workgroup', and 'Merge work queues'.

Configure worklist for an operator

Navigation

From Dev Studio:

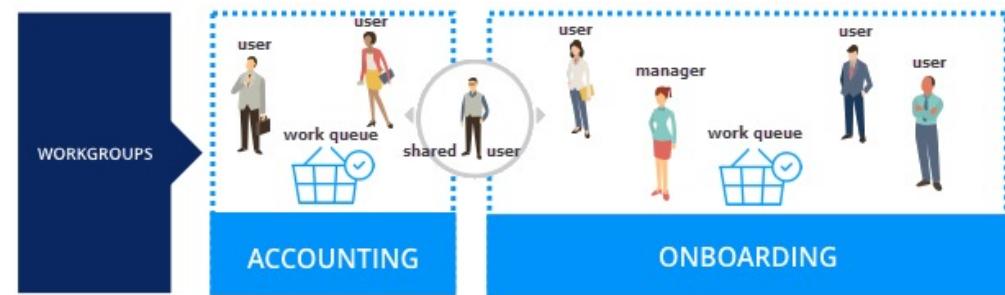
1. In the lower left corner, click on the Operator icon
2. Select Operator from the list
3. Click on the work tab from the operator ID rule
4. Configure the routing options for the operator



Work group

Definition and description

- A cross-functional team that contains a manager, users (operators), and a work queue.
- The work queue contains shared work and resources across the business.
- Operators associated with a work group can share work among operators in different business units.
- A work group identifies one operator as the work group manager. Managers have the option to assign, monitor, and report on work performed in each work group.
- Work groups can contain authorized managers to help transfer assignments.
- The *case manager portal* refers to work groups as teams.



Routing options

Description

Current User

The default setting which routes an assignment to the **current user** if the user who completed the preceding assignment should complete the current task.

Specific User

Route an assignment to a **specific user** if an individual user must complete the assignment by assigning the task to a Username, User reference field, Reporting manager or Participant.

Work Queue

Route to a **work queue** if anyone in the group can complete the assignment. The open assignments will appear in order of importance. Assignments stay in the work queue until a user associated with the work queue selects an assignment, or a manager sends an assignment to a specific user.

Business Logic

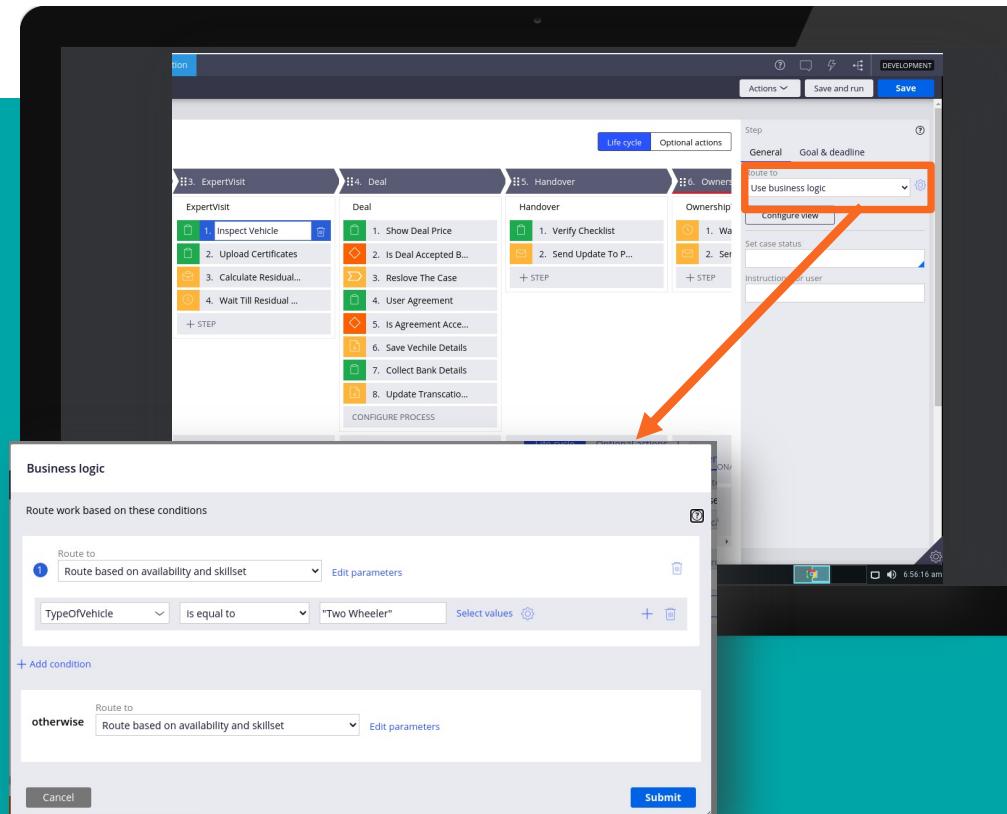
Route work based on **business logic** conditions that uses an algorithms or decisioning rules to route an assignment to the appropriate person or work queue.

Business Logic in Routing

Navigation

From app or dev studio

1. Select the case type
2. Select the assignment step
3. Select business logic from the configuration pane
4. Click the gear icon to display the business logic configuration options
5. Input the conditions for routing





Display operator work list

Navigation

From Dev Studio:

1. Launch portal menu > User portal
2. Click on Home option on the left pane

The screenshot shows the Pega Dev Studio interface. On the left, there is a dark sidebar with various icons: a person (highlighted with an orange box), a magnifying glass, a plus sign, a blue arrow pointing up (highlighted with an orange box), a clock, a grid, a bar chart, and a magnifying glass. The main area displays the 'Showcase' page. At the top, it says 'Welcome to Showcase' with a sub-headline: 'We've launched a brand new experience to accelerate your workflow. Check out the guides to help you get the most of Cosmos.' Below this is a 'Discover Cosmos' button. To the right, there is a section titled 'My worklist' with a count of 4 items. Each item is a 'Collect Resume' task assigned to 'Dwight Schrute' due '2 days from now' in 'JOBAPP-1', 'JOBAPP-2', and 'JOBAPP-3'. Each task has a 'Go' button. At the bottom of this list is a 'View all' link. An orange arrow points from the 'My worklist' section towards the right edge of the screen.



Review manager work queue

Navigation

From Dev Studio:

1. Launch Portal menu > User portal
2. Review the work queue, team and worklist from the dashboard

The screenshot shows the Pega Dev Studio interface with the User portal selected. The dashboard is divided into several sections:

- Worklist for S'chn T'gai Spock**: A table with columns ID, Description, Category, Due, Urgency, and Owner. One row is visible: C-1002, Collect Professional Details, Candidate, Due 10, Urgency 10, Owner S'chn T'gai Spock.
- Work queues**: A list of queues with counts:
 - Account Payable: 0
 - AccountPayable: 0
 - Authors: 0
 - HRApps:Authors: 0
 - Default team for TGB: 0
 - default@TGB: 0
 - Users: 0
 - HRApps:Users: 0
- Cases entered by me**: A table showing cases entered by the user:

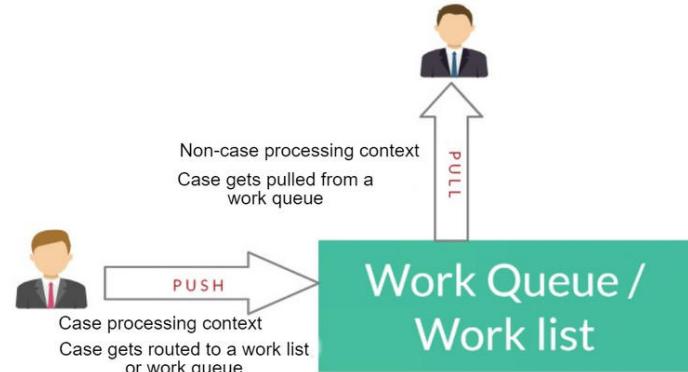
ID	Description	Status	Created date
C-1002	Candidate	New	12/7/21 1:42 PM
E-1001	Employee Evaluation	Open	12/7/21 1:41 PM
- Teams**: A list of teams managed by the user:
 - Facilities@TGB: Managed by Susie Collins
 - My team: Managed by Administrator
 - My team (primary team): Managed by Aaron Alizadeh
 - Payroll@PegaHR: Managed by Payroll Manager
 - Recruiting@TGB: Managed by John Jackson
- Case volume**: A line chart showing the count of cases over time. The chart shows two data points: a peak of approximately 8 cases at the start and a lower point of approximately 4 cases later.



Push vs pull routing

Description

- **Push routing** logic is invoked during case flow processing to determine the next assignment for the case.
 - When routing to a worklist assignment, Pega can use multiple criteria to select the ultimate owner, such as availability (whether an operator is available or on vacation), the operator's work group, operator skills, or current workload.
- **Pull routing**, also known as the **system-selected assignment model**, occurs outside the context of a case creating an assignment.
 - Get Next Work -will pull the next most urgent assignment to work on from a set of assignments list.





Push routing activities

Description

- There are four main categories of **push routing activities**, as shown in the following table.

Common	Organization-Based	Decision-Based	Skills-Based
<i>ToAssignedOperator</i> <i>ToCreateOperator</i> <i>ToCurrentOperator</i> <i>ToWorkParty</i> <i>ToNewWorkParty</i> <i>ToWorkbasket</i> <i>ToWorklist</i>	<i>ToWorkGroup</i> <i>ToWorkGroupManager</i> <i>ToOrgUnitManager</i>	<i>ToDecisionMap</i> <i>ToDecisionTable</i> <i>ToDecisionTree</i>	<i>ToLeveledGroup</i> <i>ToSkilledGroup</i> <i>ToSkilledWorkbasket</i>



ToSkilledGroup routing

Description

- The assignment is routed to an operator in the work group who is available with the required and desired skills.
- If none of the work group operators meets the criteria, the assignment is routed to the work group manager.

Step (?)

General Goal & deadline

Route to Custom

Assignment type Worklist

Router ToSkilledGroup

Parameters

workgroup * ?
default@GoGoR

Skill	Level	Required
French	5	<input checked="" type="checkbox"/>

[Update Skill](#)

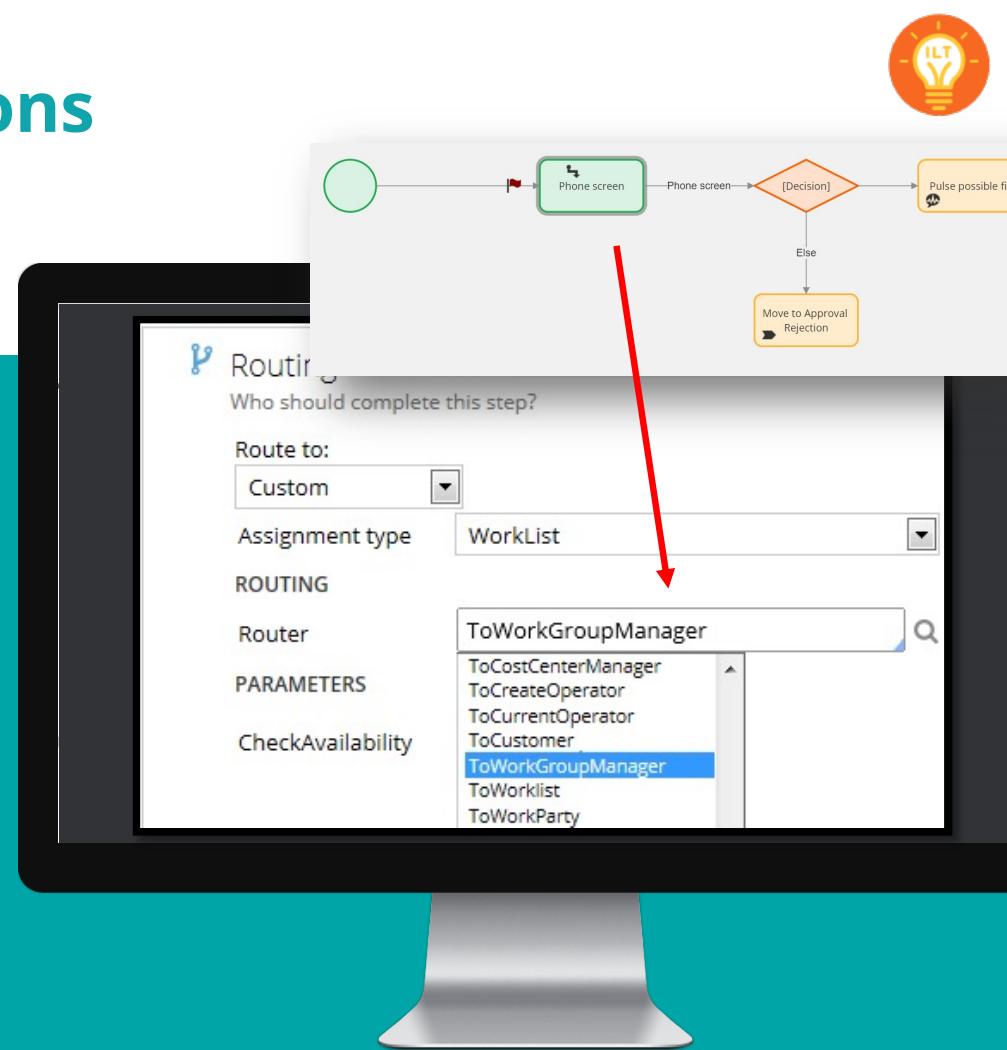


Configure push routing options

Navigation

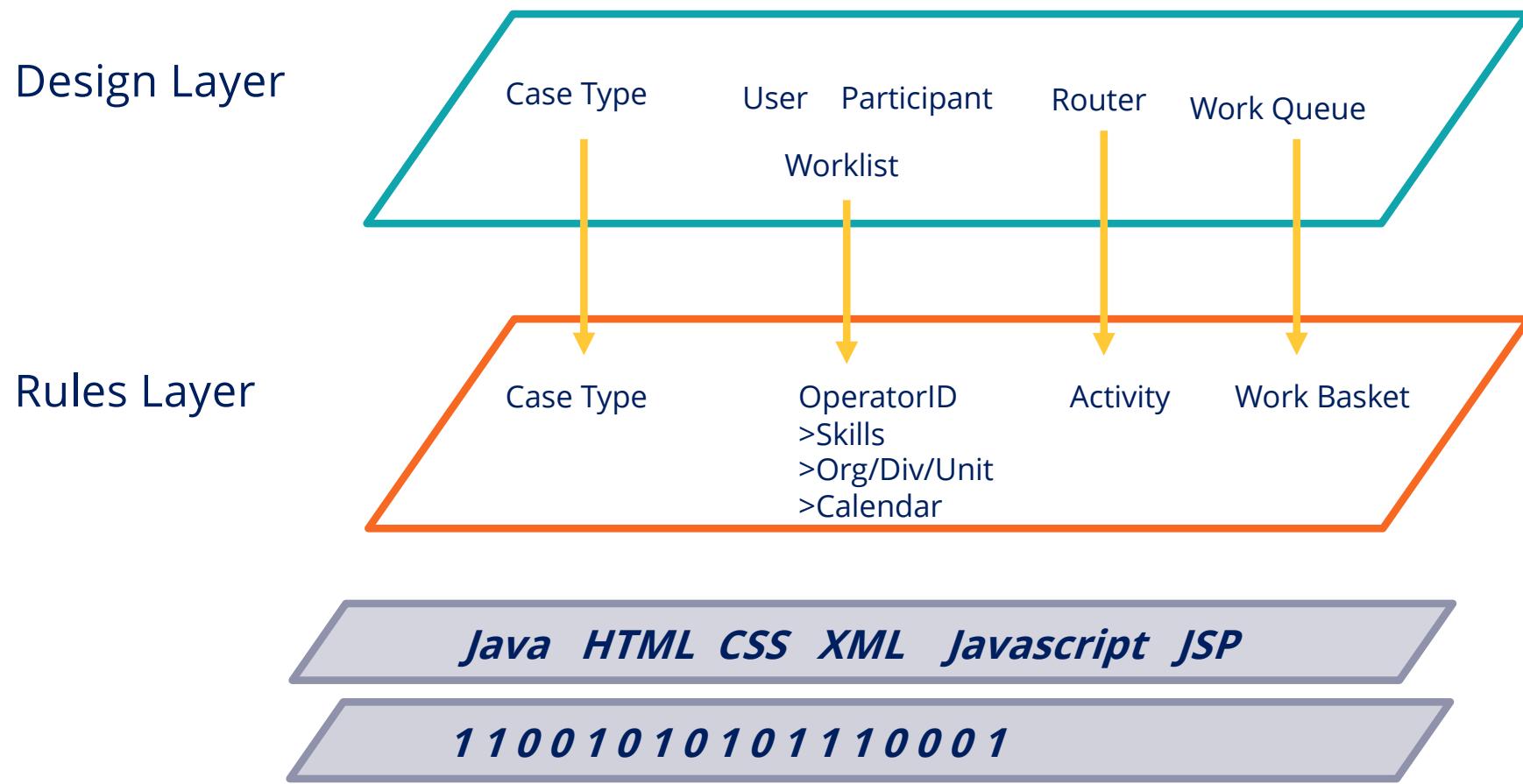
From Dev Studio

1. In the process modeler, double click an Assignment shape
2. The properties dialog box will display
3. Select the routing configuration options





Schematic





Unit Test and Debug

Routing is best tested by running the case type.

1. Select preview application.
2. Run the desired case type.
3. Click on Actions > History; verify the assignment area and confirmation messages stating the task has been routed appropriately.

The screenshot shows the Pega Case Management interface for a case titled "Evaluate and Sell a Vehicle" (E-9). The main screen displays various case details like Priority (10), Status, and Created/Updated dates. A red box highlights the "Actions" dropdown menu, which includes options like "View history", "Manage tags", and "Manage notifications". The "View history" option is selected and shown in a modal window. The modal window contains a green success message: "Thank you! The next step in this case has been routed appropriately." followed by a "Get next" button. Below this, the "History" section is visible, showing a table of events with columns for Time, Description, and Performed by. One specific event is highlighted with a red box: "Assigned to mohith to 'complete task'". The "CASE HISTORY" tab is selected in the history panel.

Time	Description	Performed by
9/30/21 12:12 PM	Assigned to mohith to 'complete task'	mohith
9/30/21 12:12 PM	Case moved from ManagerReview to ExpertVisit via automatic stage transition.	mohith
9/30/21 12:12 PM	Correspondence has been attached: Request Approved.	mohith
9/30/21 12:11 PM	Assignment to 'Please approve or reject this (1)' completed by performing a Evaluate and Sell a Vehicle.	mohith
9/30/21 12:11 PM	Correspondence has been attached: Evaluate and Sell a Vehicle (E-9) from mohith is waiting for app.	mohith

Best Practices

- Route to a participant, role or perform a lookup rather than directly to an Operator ID.
 - If an operator changes roles or is no longer with the company, the case type does not have to be modified to reflect the change.

Skill Mastery

Understand:

- Routing functionality in App Studio.
- Routing functionality in Dev Studio.
- Routing to an individual "Worklist"
- Routing to a group of users "Work Queue"
- Routing using Business Logic
- Push and pull routing



SKILL
LESSON

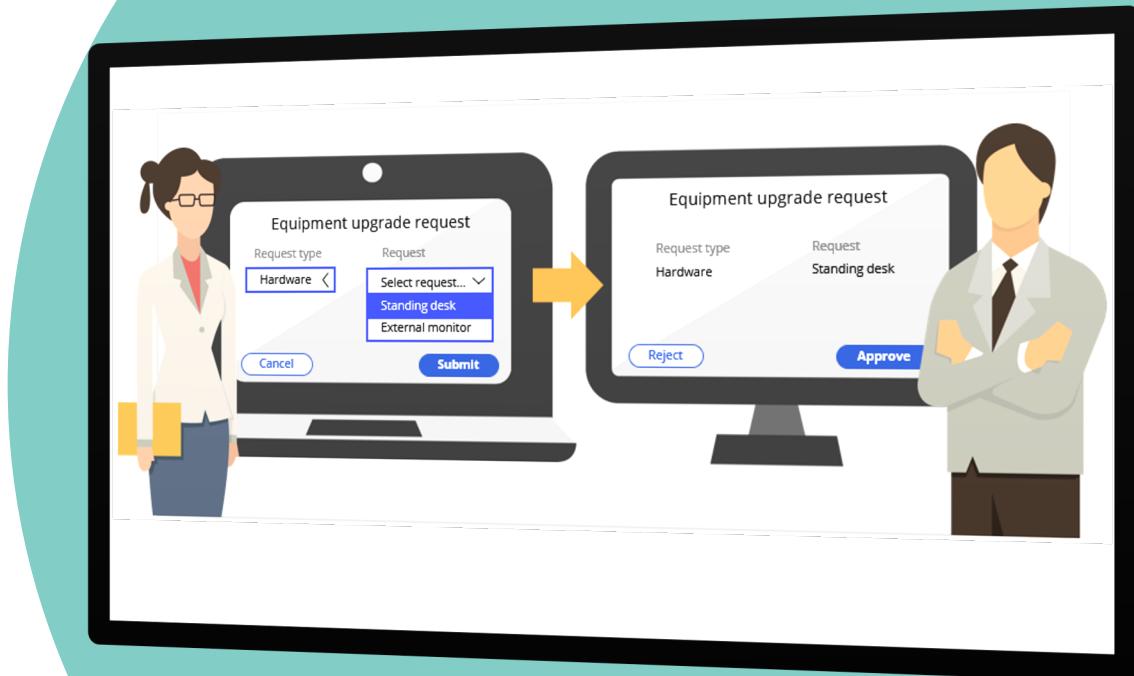
Case Approvals



Overview

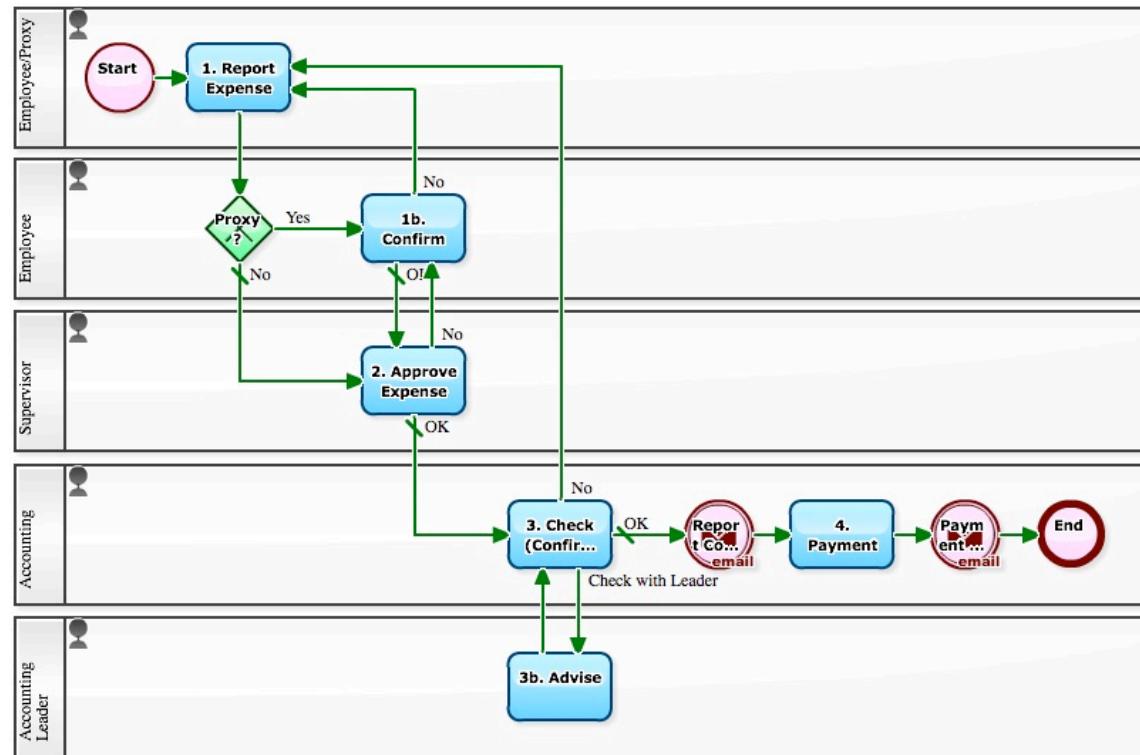
Organizations can ensure that business processes meet policies and requirements by requesting approval from users.

After reviewing case details, users can approve or reject a case and bring the case closer to resolution.



Use Case

- An employee submits a request for an equipment upgrade expense. Equipment upgrade expense requests are sent to a manager, who either approves or rejects the request.
- If the manager approves the request and the upgrade expense costs more than \$500 dollars, then the manager's manager also needs to approve the request.



Case approvals

Definition

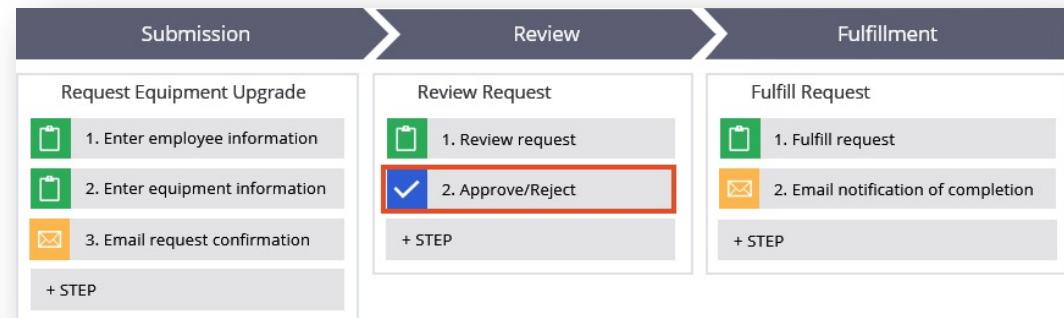
- Case approvals are decision points at which one or more users decide whether to approve or reject a case.
- Approvals can be added to a business process to seek consent from users with different roles or levels of expertise.
- Email and mobile push notifications can be configured for approvals.

The screenshot shows a user interface for managing a case approval. On the left, a modal window titled "Approval" displays the message: "Please approve or reject this New vendor". Below the message is a "Notes" input field. At the bottom of the modal are three buttons: "Cancel", "Reject", and "Approve", with "Approve" being highlighted in blue. To the right of the modal is a sidebar titled "Case details" which includes information such as "Last updated by Author (1m ago)" and "Created by Author (4m ago)". Below this is a section titled "Open assignments" with a button labeled "Get Approval" and "(Current) Compliance officer". The background of the slide features a yellow decorative pattern in the bottom right corner.

Approval/Reject step

Definition

- Assigns a task to a user, typically a manager, to review case information and then decide whether the case is approved.
- To configure an Approve/Reject step, define who is assigned to the approval and how the case proceeds if the case is approved or rejected.
- The approval step routes a case to one or more reviewers, based on a username, reporting structure, or authority matrix.

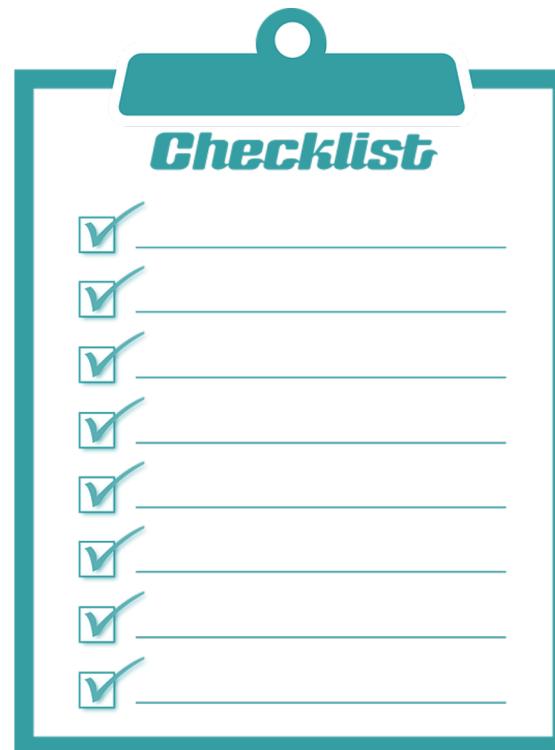


Approval/Reject step

Implementation

To configure the approval/reject step, be sure to complete the following checklist:

- Add the Approve step as an automation step in a process.
- Configure routing on the General tab.
- Configure a user view for approval step on General tab.
- Configure action to take upon Approval or Rejection on the Flow tab.
- Configure email or mobile approval.



Approval flow type

Description

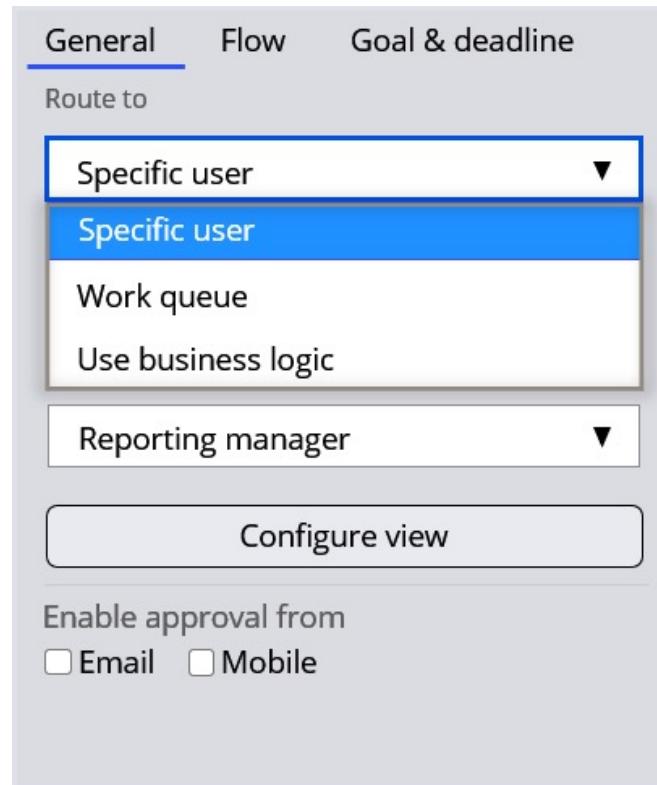
- Single level
 - Receive approval from a single user by indicating a specific user, work queue or business logic.
- Cascading
 - Configure reporting structure or authority matrix
 - Receive approval from people on different levels of your organizational chart or in different parts or departments of your organization.

Conditions	Actions
<input type="radio"/> TotalPRCost >=	<input checked="" type="radio"/> Approver ID
<input type="radio"/> when 0	→ "CSM@Pco.com"
<input type="radio"/> when 25000	→ "VP@Pco.com"
<input type="radio"/> when 75000	→ "VPFinance@Pco."
<input type="radio"/> when 100000	→ "SVP@Pco.com"
<input type="radio"/> when 500000	→ "CFO@Pco.com"
<input type="radio"/> when 500001	→ "CEO@Pco.com"
otherwise	→ ""

Single level approval

Description

- Assign approvals to the worklist of a specific user, a work queue, or use business logic to assign work based on custom conditions.
- **Specific user**
 - A single user will be assigned the task of approving or rejecting an action.
- **Work queue**
 - A list of approvals for a group of users, usually managers, assigned the task of approving or rejecting an action.
- **Business logic**
 - Create custom routing options to ensure approvals are routed to the most appropriate user, usually a manager.

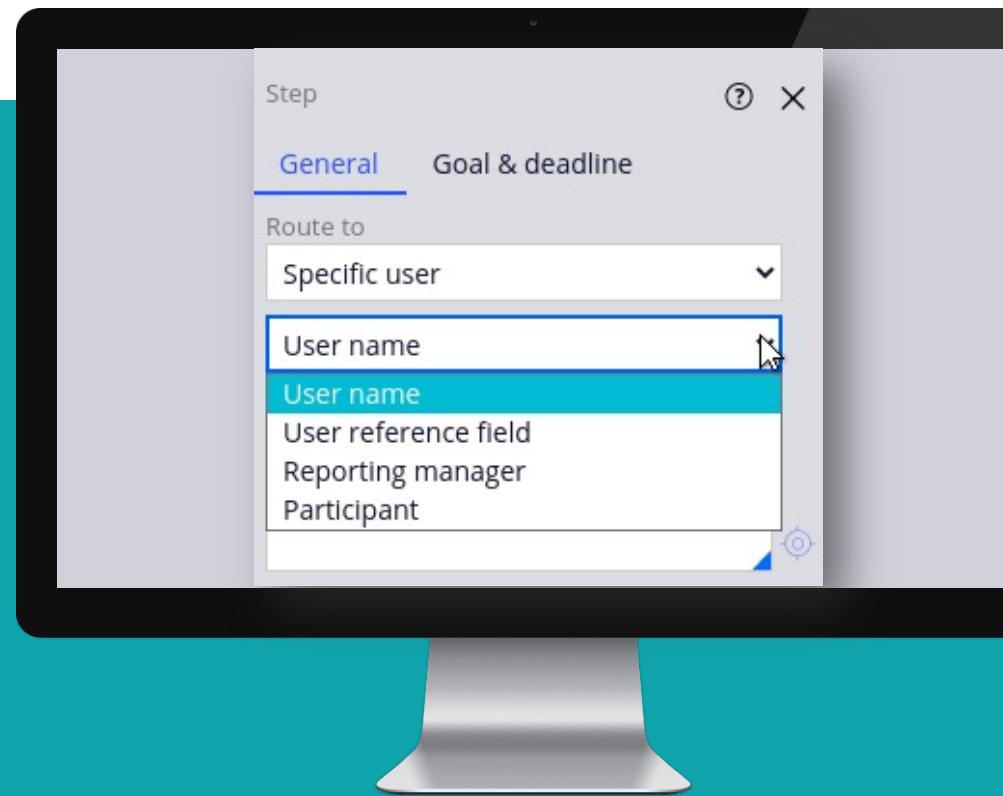


Single level approval

Navigation

From App or Dev studio:

1. Select the case type
2. Click +Step to add the approval/reject step
3. In the configuration pane, select single level as the approval flow type
4. In the Route to field select: Specific user, Work queue, or business logic



Cascading approval

Definition

A list of approvers is processed until the action is rejected or the last approver in the list approves the action.

An Approval list is based on:

- Reporting structure
- Authority matrix (separate lesson)

Step (?)

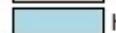
General	Flow	Goal & deadline
Approval flow type	Cascading	
Approval based on	Reporting structure	
Approval to be completed by	Workgroup manager	
	Reporting manager	
	Workgroup manager	
<input type="checkbox"/> Enable email approval		
Configure view		

Reporting structure

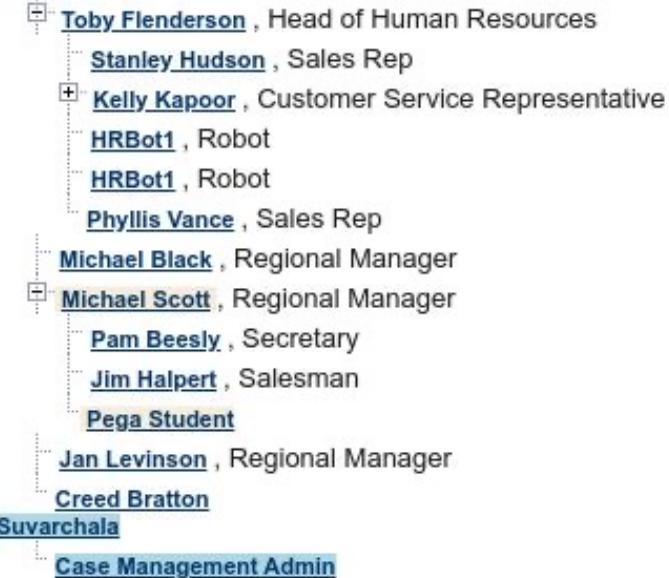
Definition

- Type of cascading approval that allows receiving approval from people on different levels of an organizational chart.
- Determines which operator is the starting point in the reporting structure.
 - Reporting manager
 - Reporting manager of current user.
 - Workgroup manager
 - Manager of the current user's default workgroup.

Reporting Structure

 Highlighted operators shows reporting structure
 Highlighted operators are not available to receive work

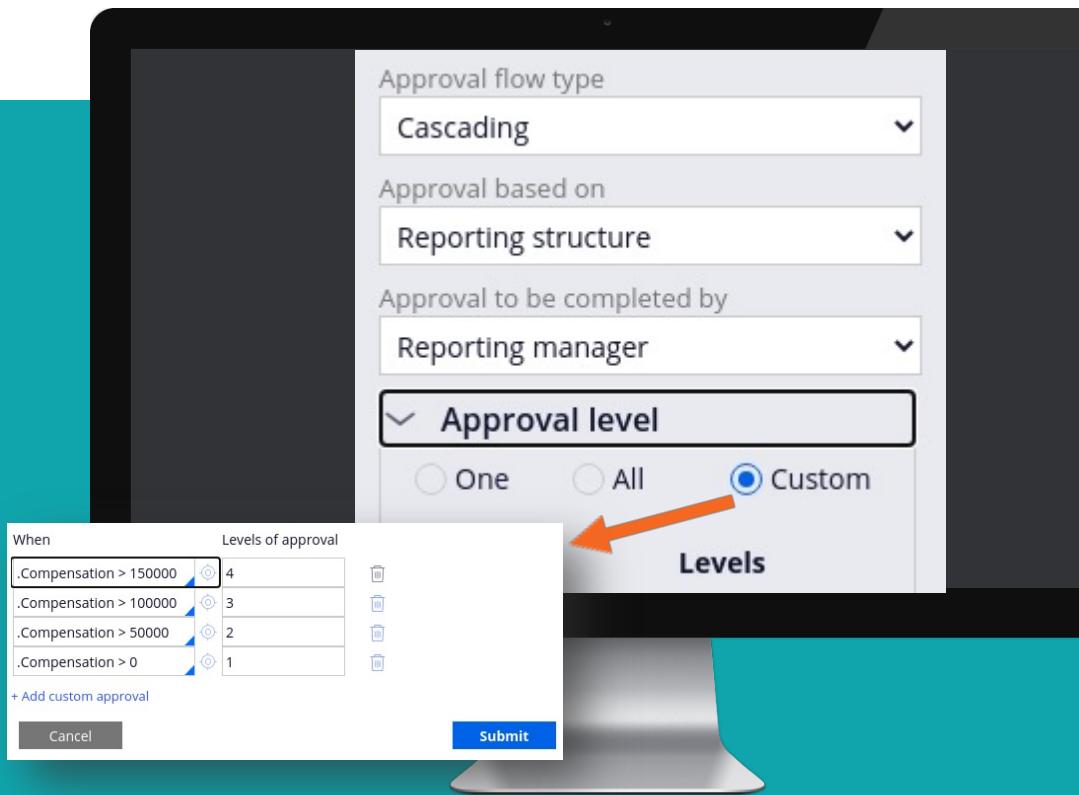
David Wallace , CEO



Reporting structure

Navigation

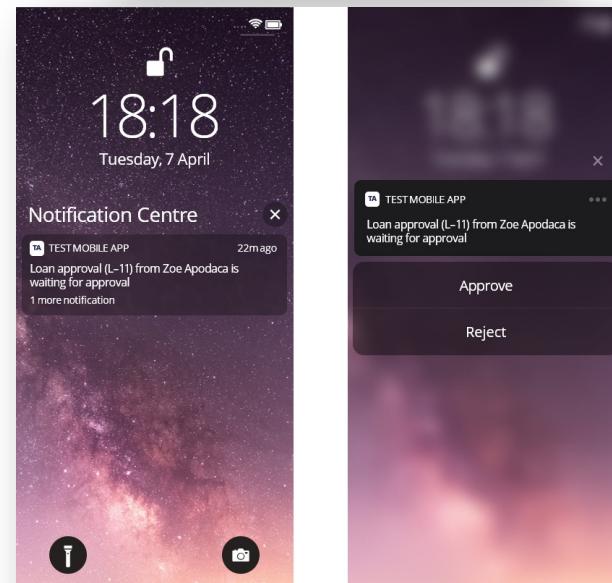
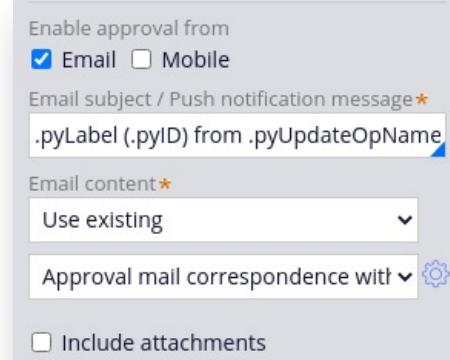
1. Select the approval/reject step
2. In the configuration pane, set the Approval level to One, All or Custom.
 - Condition can be a **When rule** or hard coded directly
 - Only first true condition used, make sure conditions are in descending order



Email Approval

Description

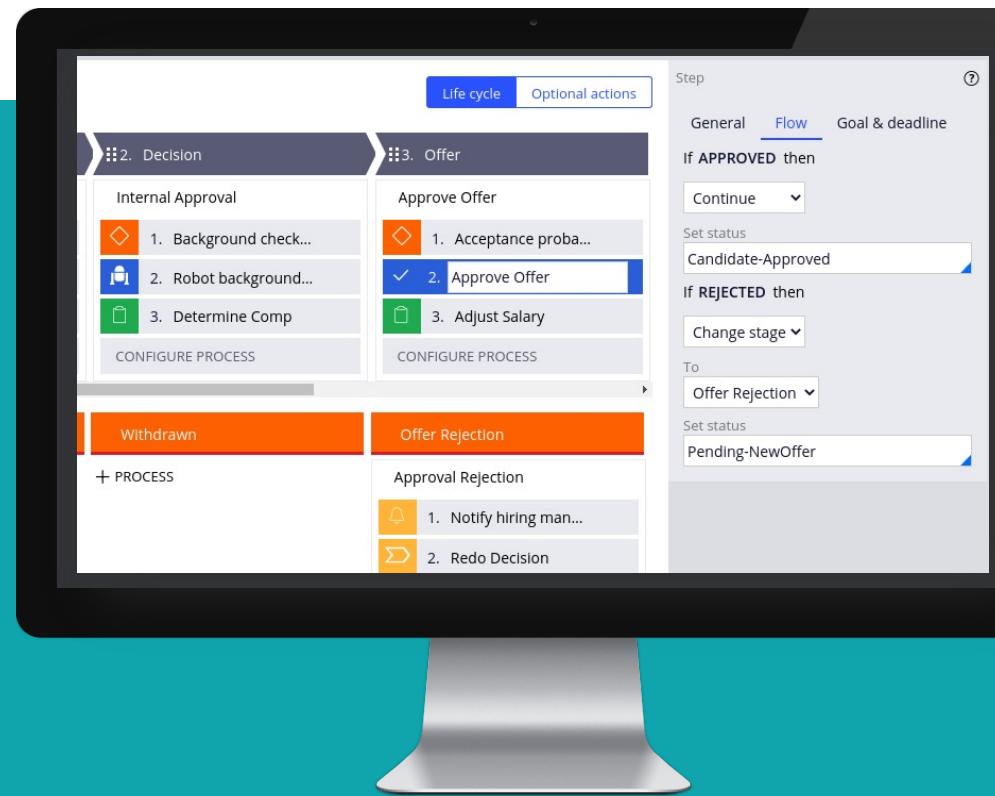
- Enable email approval to accept, reject, or take other actions on a case without login to the application.
 - The email includes an embedded prompt to accept or reject the case step.
 - The receiver clicks a button in the email message, which generates an email response that includes their decision and sends the email back to the application.
- Configure at least one email account in the application that can send and receive emails.



Approval flow

Navigation

1. Click the Flow tab in the configuration pane
2. Set If APPROVED then to:
 - Continue - moves case to next step
 - Change Stage - select stage
3. Optionally set the status
4. Set If REJECTED then to Continue, change stage or
 - Resolve -ends case
5. Set status must be set
 - Defaults to Resolved-Rejected





Schematic

Design Layer

Case Type

> Approval Step

Rules Layer

Flow

Sub Flows
Assignments
Router
OperatorID(s)
When

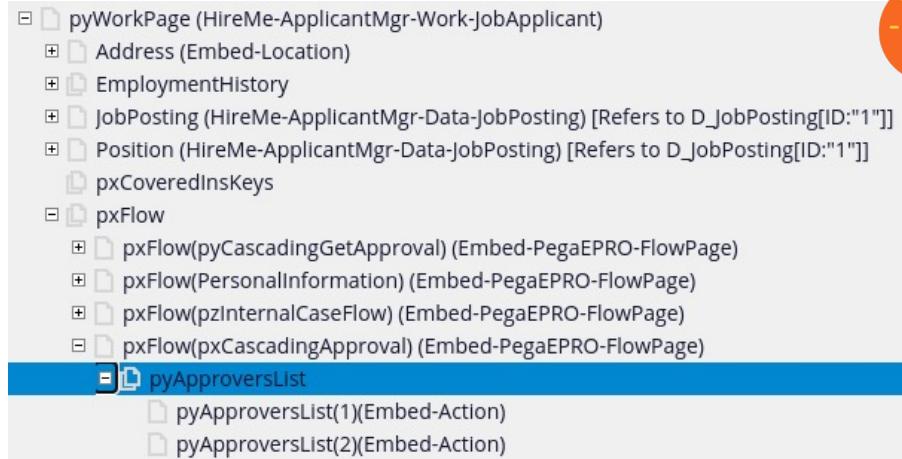
Java HTML CSS XML Javascript JSP

11001010101110001

Approvals

Unit Test and Debug

- While in an approval flow there are several useful properties on the *pyWorkpage* to observe on the clipboard
 - pyCurrentApprovalIteration*
 - pyCurrentApprover*
 - pxFlow(pxCascadingApproval).pyApproversList*
 - The list is lazy populated for reporting structure
 - The list is fully populated right away for authority matrix



pyApprovalResult	Approved
pyApprovalSectionName	
pyCancelLabel	Cancel
pyCaseAssetFormat	Case secondary content
pyCaseFilter	Both
pyCaseFilterDescription	Current Assignments and Subcases
pyChargeAmount	0
pyChargeRate	
pyConfirmationNote	pyStepRoutedForApproval
pyCurrentApprovalIteration	2
pyCurrentApprover	scotm

Best Practices

- Ensure every possible current user for the step has the Reporting manager field completed in the operator record
 - Every operator record should be assigned a reporting manager
- Every operator should have a default workgroup with a manager depending on which is used for Approval to be completed
- Every workgroup should have a manager
- Failure to have a manager will result in a runtime error due to the approver list being empty

Skill Mastery

Understand:

- Case approvals
- Approval and reject step
- Approval flow
- Single level and Cascading approvals
- Reporting structure and authority matrix
- Email and push notifications
- Unit testing approvals



SKILL
LESSON

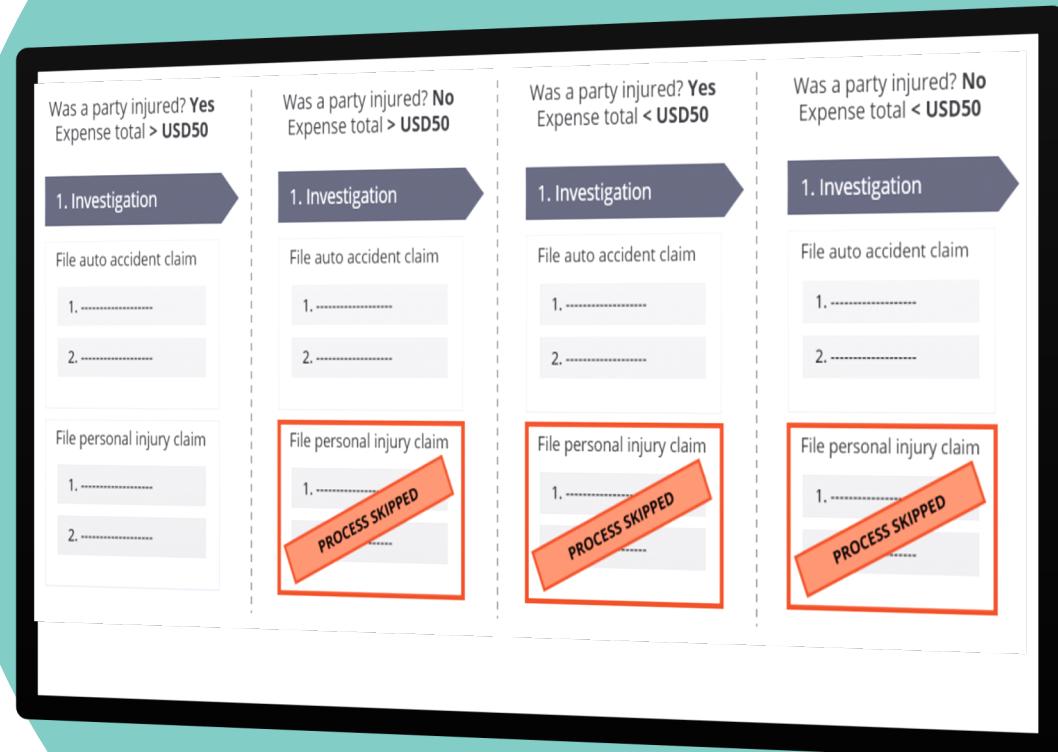
Skipping Stage or Process



Overview

Skipping a process or a stage allows the flow of a case to include only needed processes and stages.

A process or stage can be configured to run or skip conditionally.



Use Case

- Business logic is used to conditionally run a Stage or Process during case processing.
- *For example:* A vehicle was damaged, but no one was injured. During claim processing, we could skip the personal injury part of the claim.



Conditional Processing

Definition

- **Conditional processing** defines conditions that control whether a process or stage runs in a case.
 - A condition consists of a field, comparator, and value.
 - Processes and stages are either executed or skipped if the case run-time values match the conditional logic defined.
- Conditional processing is configured to enable the application to select when a process or stage is skipped.
- Ensures that a case moves to the next stage in the sequence when the current stage is not relevant, by defining the conditions that cause a case to skip a stage.
- Ensures that a process starts only when needed in a business case by defining conditions for running the process.

Configure condition

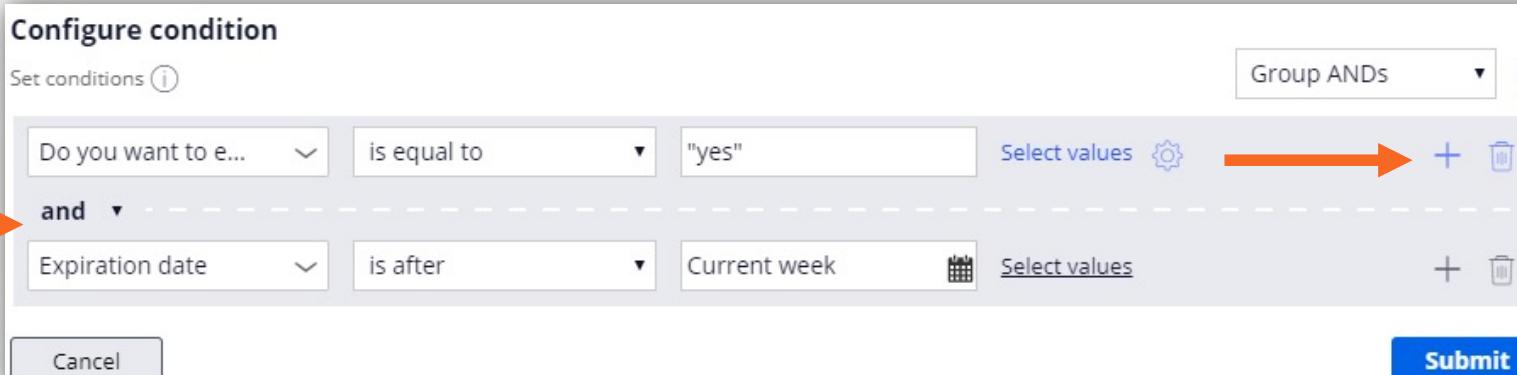
Set conditions [\(i\)](#)

Group ANDs [▼](#) [⋮](#)

Do you want to e... [is equal to](#) "yes" [Select values](#) [⚙️](#) [+](#) [trash](#)

and [Expiration date](#) [is after](#) Current week [Select values](#) [+](#) [trash](#)

[Cancel](#) [Submit](#)

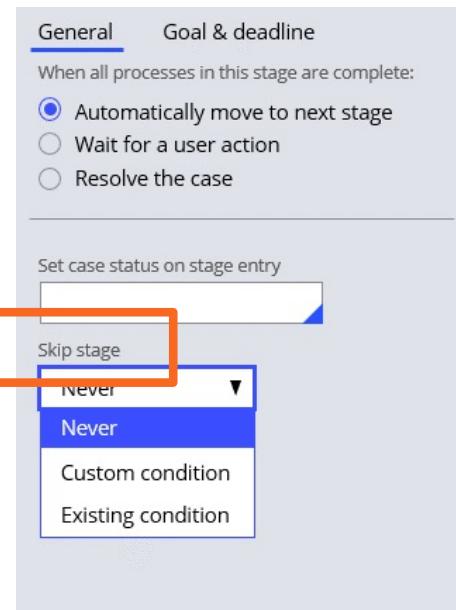


Conditionally skip a Stage vs a Process

Description

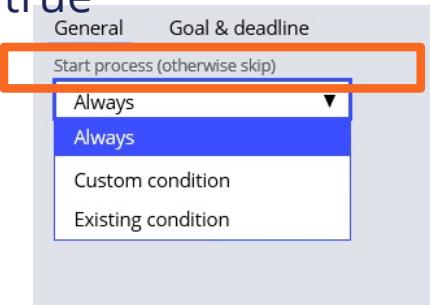
Skipping a Stage uses:

Skip Stage *When* condition evaluates to true



Skipping a Process uses:

Start Process (otherwise skip)
When condition evaluates to true



When rule

Implementation

When: WHEN_30_Inspect_Vehicle [Available]
CL: WIND-Auto-Work-EvaluateAndSellAVehicle ▾ ID: WHEN_30_Inspect_Vehicle RS: Auto:01-01-01

Conditions Advanced Parameters Pages & Classes Test cases Specifications History

Updating the condition here would restrict the editing at Conditions tab and App Studio

1 .VehicleDetails.TypeOfVehi EQUALS "Two Wheeler" Select values

Add condition

Logic string
(1)

Display property labels

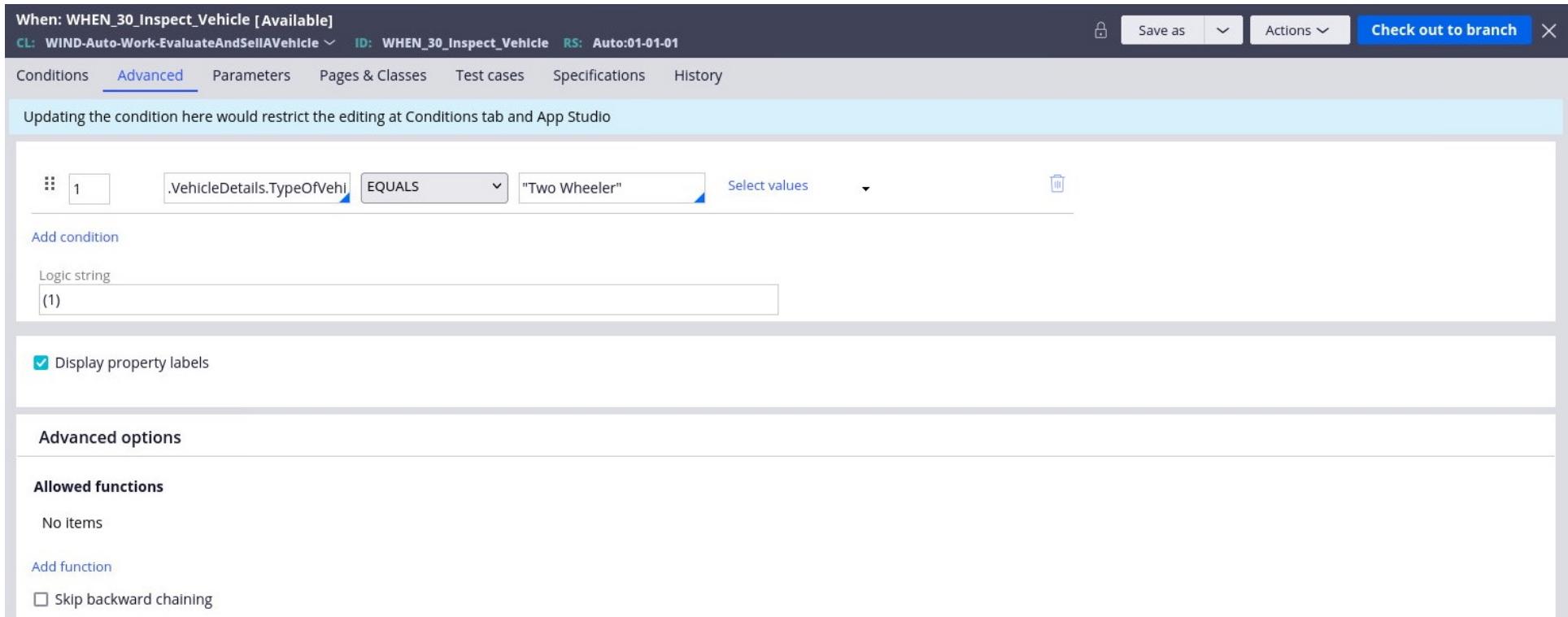
Advanced options

Allowed functions

No items

Add function

Skip backward chaining



Skip stage

Description

When skipping a Stage in **App Studio** the options are:

Skip Stage

- **Never** – default – Never skips the stage
- **Custom Condition** – Skips the stage if condition(s) evaluate to true
- **Existing Condition** – Uses an existing condition – lists when rules

When skipping a Stage in **Dev Studio** the options are:

Skip Stage

- **Never** – default – Never skips the stage
- **When Rule** – Skips the stage if when rule evaluates to true
- **Expression** – Skips the stage if expression evaluates to true

Skip process

Descriptions

When skipping a Process in **App Studio** the options are:

Start Process

- **Always** – default – Always starts a process
- **Custom Condition** – Starts process when condition(s) evaluate to true
- **Existing Condition** – Uses an existing condition – lists when rules

When skipping a Process in **Dev Studio** the options are:

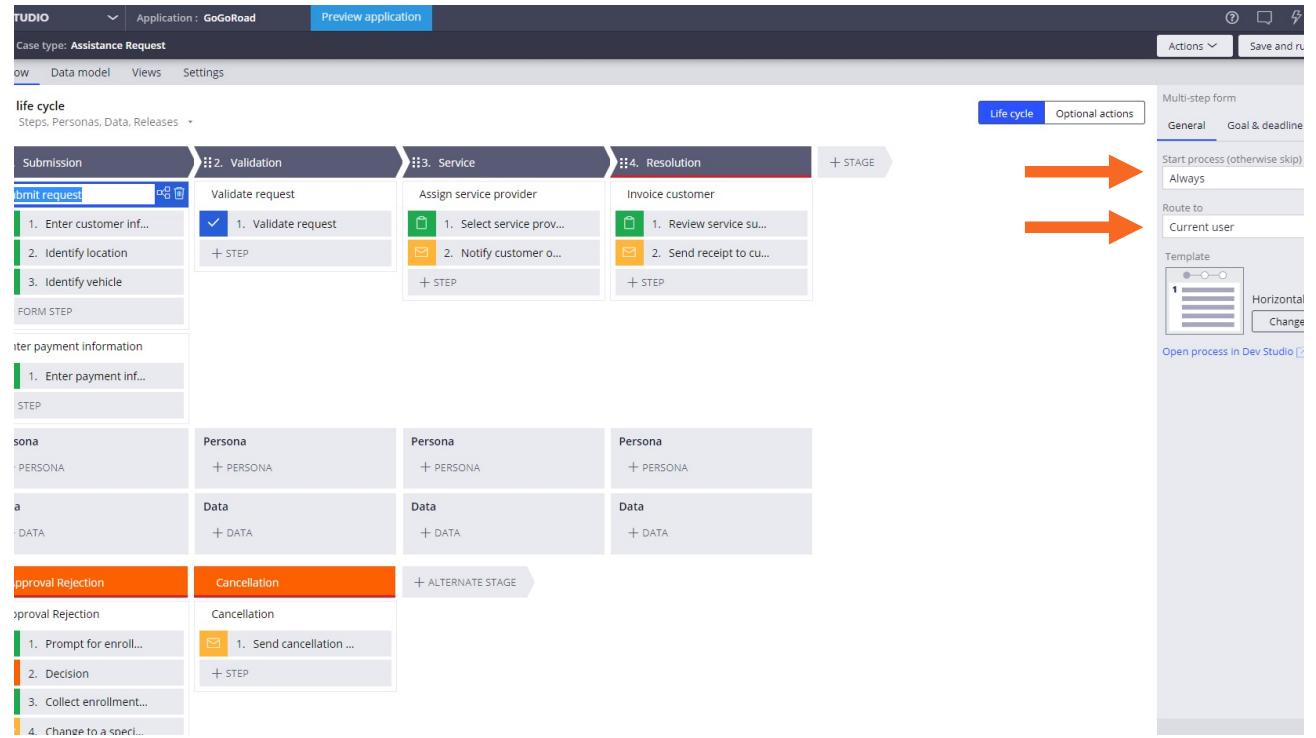
Start Process

- **Always** – default – Always starts a process
- **When Rule** – If When rule evaluates to true the process starts
- **Expression** – If the Expression evaluates to true the process starts

Skipping a Process

Navigation

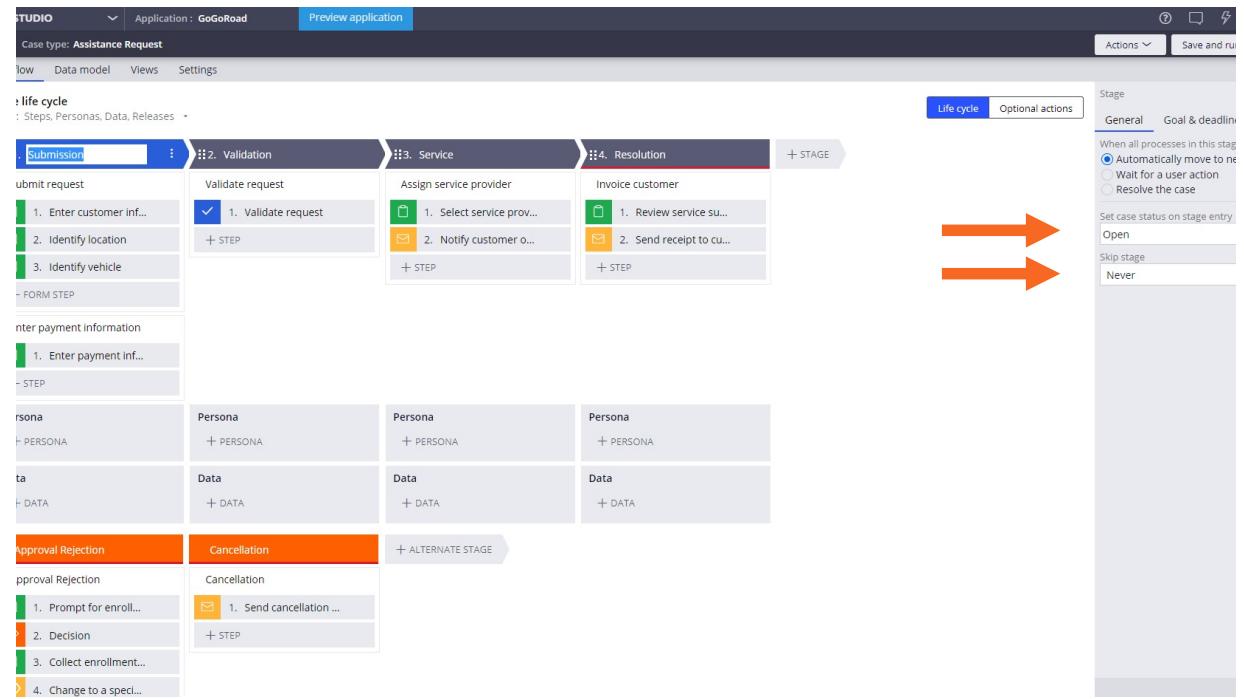
1. App or Dev Studio
2. Case Type
3. Workflow tab
4. Click a process name
5. Right context menu
6. Start process(otherwise skip)



Skipping a Stage

Navigation

1. App or Dev Studio
2. Case Type
3. Workflow tab
4. Click a stage name
5. Right context menu
 - If in Dev Studio > Validation Tab
6. Skip stage





Conditional processing configured on Case Type

Implementation

Case type: Onboarding [Available]
CL: TGB-HRApps-Work-Onboarding ID: pyDefault RS: HRApps:01-01-01
This record has 1 info warning (including 1 unjustified) [View](#)

Processes Calculations **Stages** Attachment categories Advanced Specifications History

Initialization Stage

Pre-arrival Setup

Primary Stages

Equipment Selection (highlighted with a red box)

Benefits Enrollment

Verification

Settings

Skip stage: **Never**

When all processes in this stage are complete:

Automatically move to next stage

Wait for a user action

Resolve the case

Set case status on stage entry:

Automatically launched processes

Process: FacilitiesSetup

Start process: When Rule

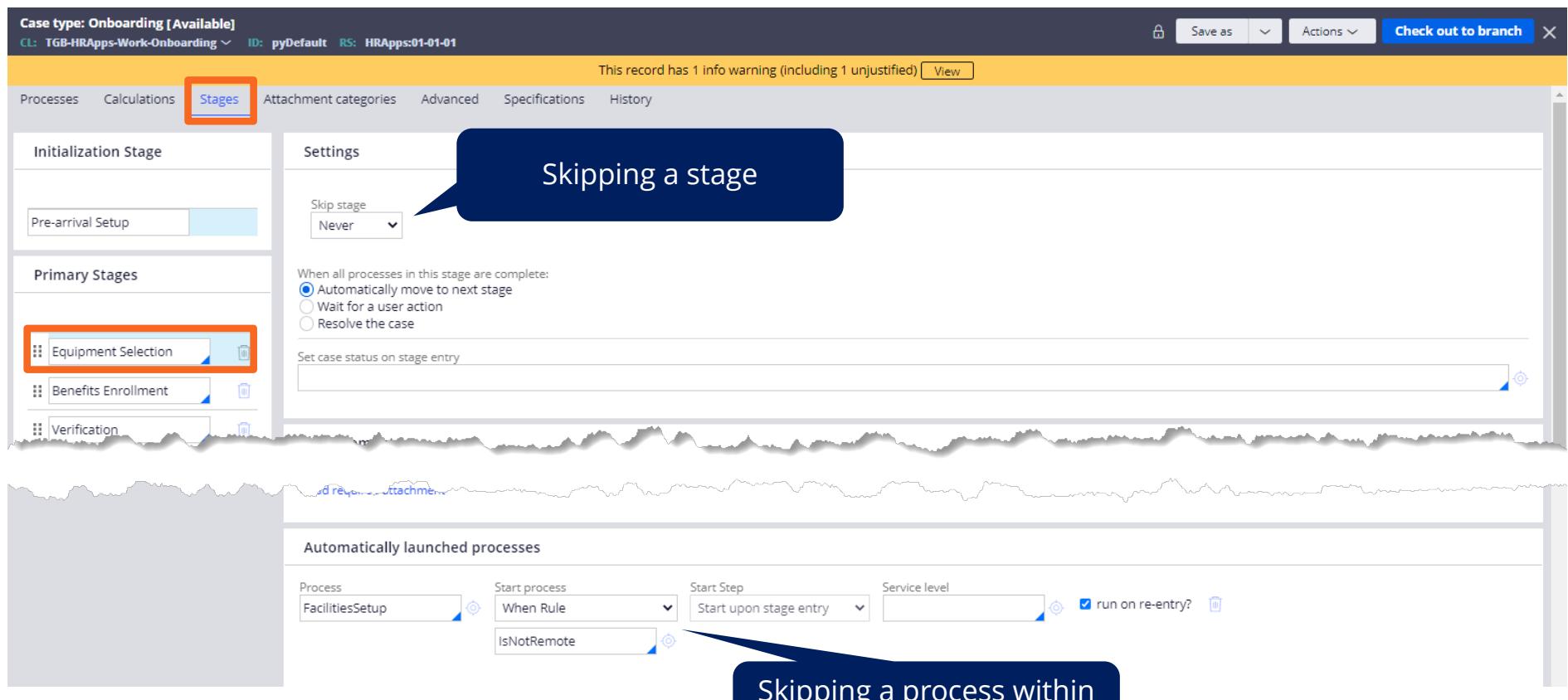
Start Step: Start upon stage entry

Service level:

run on re-entry?

Skipping a stage

Skipping a process within a stage.





Schematic

Design Layer

Condition
Expression

Rules Layer

When → Case Type

Java HTML CSS XML Javascript JSP

1 1 0 0 1 0 1 0 1 0 1 1 1 0 0 0 1

Skill Mastery

- Skip a stage
- Skip a process
- Conditional processing

Unit Testing and Debugging Guidance

- Run the case and input values that will cause a process or stage to be skipped.
- Check that the process or stage was skipped
- When rule unit testing



SKILL
LESSON

Duplicate Search



Overview

Does the system have the same request more than once?

Pega provides specific conditions that help identify potential duplicate submissions to avoid duplication and reduce inefficiency.

The screenshot shows the Pega Case life cycle interface. At the top, there are tabs for Workflow, Data model, Views, and Settings. Below this, the 'Case life cycle' view is displayed with a 'View: Steps, Personas, Data, Releases (All)' dropdown. The workflow consists of four stages: Create, Start Application, Approve Loan, and Loan Origination. The 'Start Application' stage is currently selected. This stage contains several steps: 'Create' (with 'Create' and '+ FORM STEP' options), 'Customer Details' (with '+ PERSONA'), 'Loan Selection' (with '+ STEP'), and 'Search duplicate cases' (which is highlighted with a red box). To the right of the workflow, there is a 'Life cycle' tab and an 'Optional actions' section. A modal window titled 'Step' is open, detailing 'Potential duplicate cases must meet the following basic conditions'. It lists three conditions with weighted scores: 'When Last Name is same' (score 10), 'And weighted conditions sum at least * 10' (score 5), and 'When Email address is equal to "tedsmith@test.com"' (score 7). There is also a '+ Add weighted condition' button.

Use Case

- Use the Search Duplicate Cases to match potential duplicate case instances.
- A customer enters a claim for a theft online and the confirmation email is not received. But the case has been created.
- The customer calls the contact center and has the same claim entered by a CSR. A new case is created.
- The two case instances are identified as duplicates.



What is a Duplicate Case?

Definition

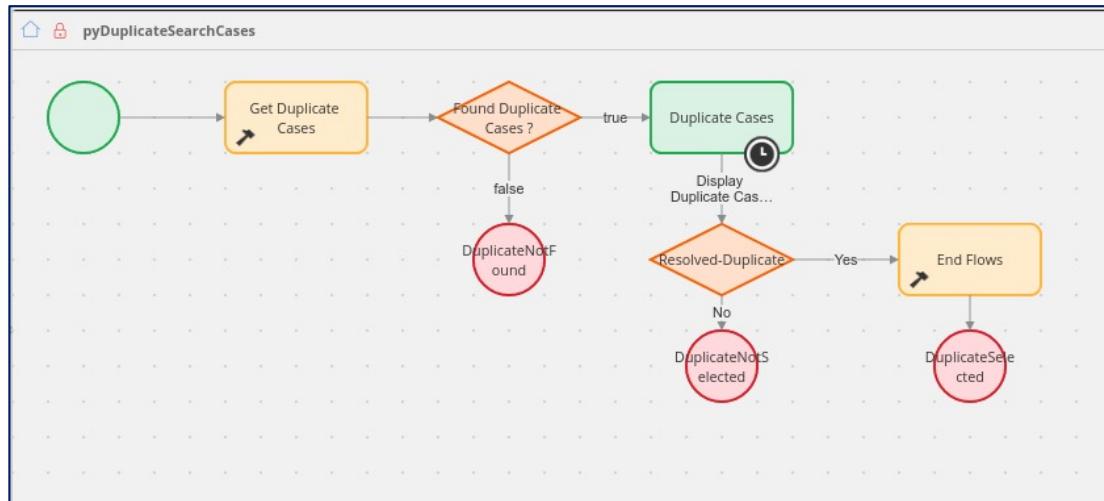
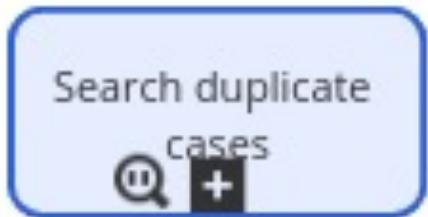
- A user may create a case which has the same data values as another case in the system.
- For instance, two purchase requests may have the same request date, the same items, or the same customer's name. However, if a specific combination of data values match, the new case is possibly a **duplicate case**.
- Pega provides the **search duplicate cases step** to help users identify and resolve duplicate cases.
- This process is implemented in the case life cycle as a **Search duplicate cases step**.



Duplicate Search?

Description

- The Search Duplicate Cases shape may also be added directly to a flow rule.



Duplicate Search uses Case Match Rule

Description

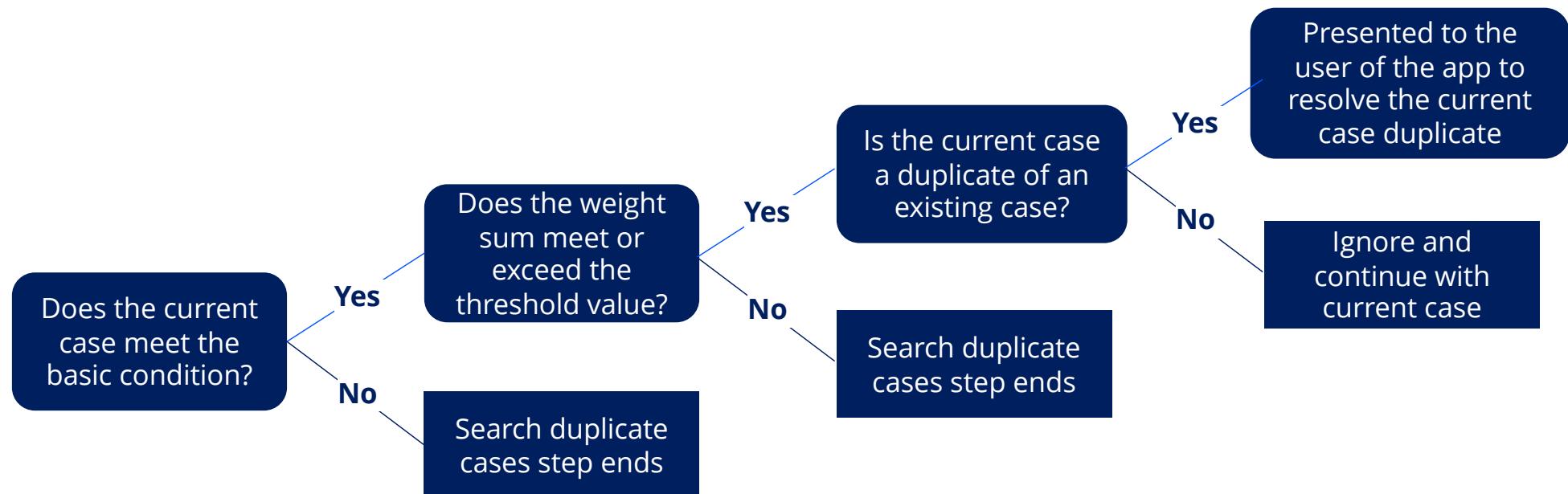
- When a **Search duplicate cases** step is added to the case type, the system creates a **Case Match** rule to define the matching criteria.
- This rule can be viewed or modified from Dev Studio.
- The rule can also be used for reporting purposes.

The screenshot shows the 'Case Match' configuration page in Pega Dev Studio. The top navigation bar includes 'Home', 'Evaluate and ...', and 'pyDefaultCase...'. The main title is 'Case Match: pyDefaultCaseMatch [Available]'. Below it, 'CL: WIND-Auto-Work-EvaluateAndSellAVehicle' and 'ID: pyDefaultCaseMatch' are listed, along with 'RS: Auto [Branch: AUTOTESTG]'. The 'Evaluation' tab is selected, showing the 'Case Matching Type' section with 'Seek cases' and a 'Must match conditions' section. The 'Must match conditions' section lists three conditions under 'Potential duplicates': '.VehicleDetails.Make' (Relationship: 'is same'), '.VehicleDetails.Model' (Relationship: 'is same'), and '.VehicleDetails.Registration' (Relationship: 'is same'). Below this is a '+ Add must match condition' link and a 'Show filter logic' link. The 'Weighted match conditions' section follows, with a note about summing weighted conditions and a table for defining them. The table has columns for 'Weight' (25), 'Condition' (.VehicleDetails.TypeOfVeh), 'Operator' ('EQUALS'), and 'Value' ('"Four Wheeler"'). A '+ Add weighted condition' link is at the bottom. The right side of the slide features a decorative pattern of blue dots and a yellow semi-circle.

Functionality of the Duplicate Search

Description

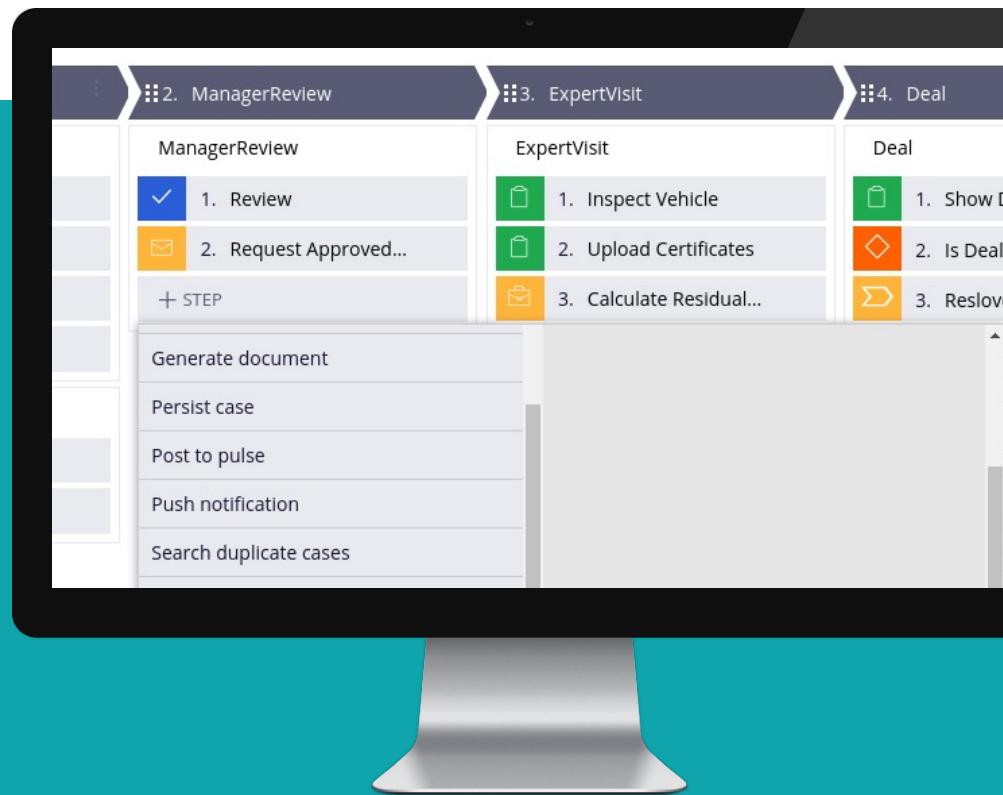
- When a case enters the **Search Duplicate Cases** step, the system uses **basic conditions** and **weighted conditions** to compare specific property values with cases already present in the system.



Search Duplicate cases Step

Navigation

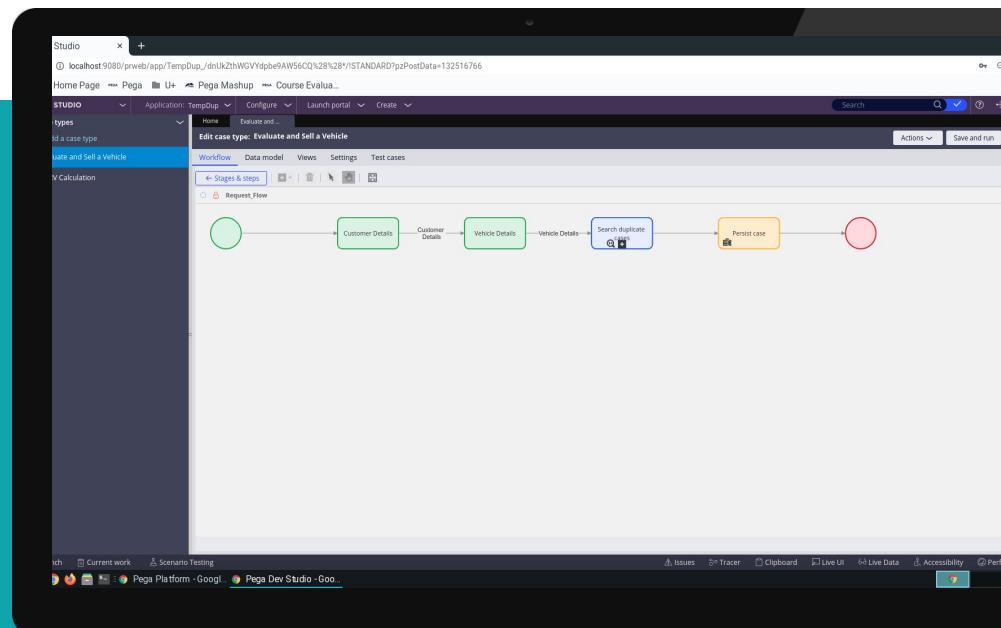
- From Case Designer, add the step to the flow.
- Select **+ STEP > More > Automations > Search duplicate cases**



Search Duplicate cases shape in a Flow rule

Navigation

- Access the workflow by opening the Flow Rule in Dev Studio.
- Right click and add the Search Duplicate cases shape to the flow.

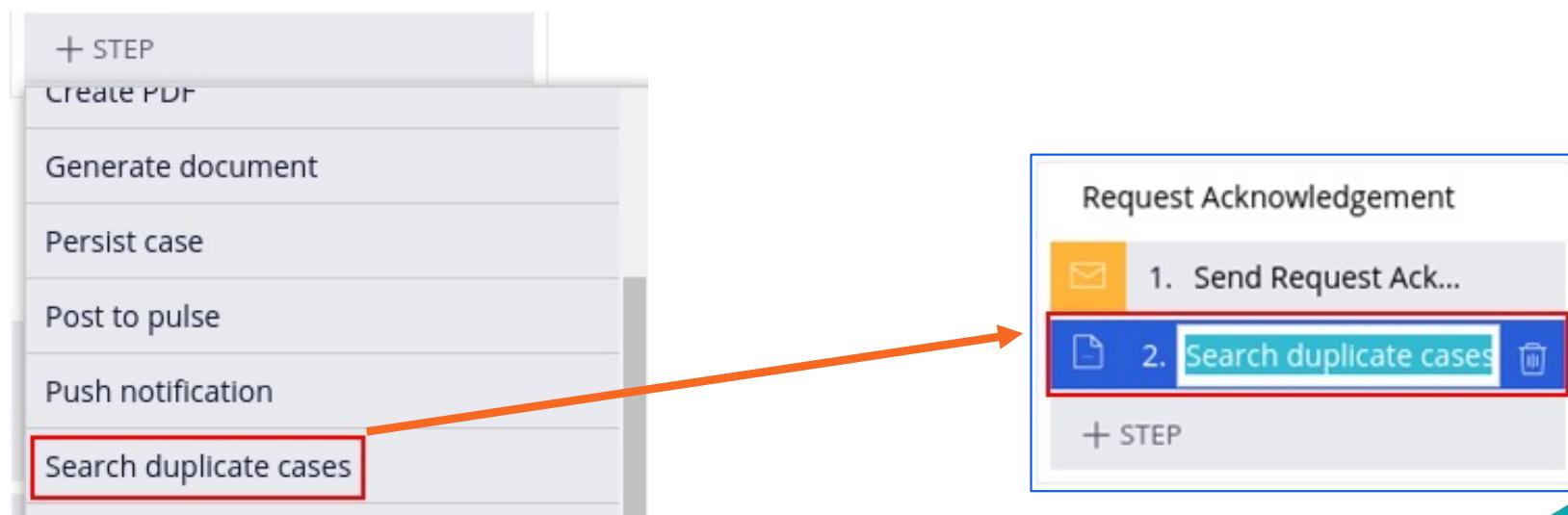


From Case Designer

Implementation

STEP ONE

- Add a Duplicate Search Cases step to the Case type.
 - Select **+ STEP > More > automations > Search duplicate cases**

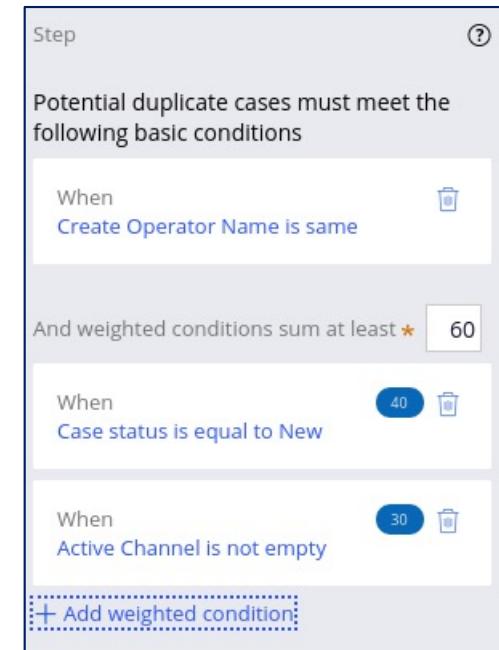


From Case Designer

Implementation

NEXT

- Configure basic and weighted conditions.
- Basic conditions specify criteria that duplicates must meet.
 - Useful for reducing the number of cases to be matched and improving performance
- The sum of the weighted conditions must be greater than a threshold.



Display Duplicate Cases

Implementation

RUN TIME

- When a potential duplicate is identified, the end user is notified of a potential duplicate and can choose to
 - Ignore and continue treating the cases as unique instances
 - Or close the case as a duplicate

The screenshot shows a Pega application window titled "Customer Onboarding (C-60) NEW". A green banner at the top indicates "3 days from now". Below it, a section titled "Display Duplicate Cases" (with a "DUE IN 6 DAYS FROM NOW" badge) displays a message: "The current case was identified as a potential duplicate of the following:". A table lists one potential duplicate case:

Case ID	Created on	Status	Why is this shown here?
C-58	Feb 1, 2019	New	Why is this shown here?

Below the table, a question "How do you want to resolve the event?" is followed by two buttons: "Close this case as duplicate" and "→ Ignore & continue". At the bottom are "Cancel", "Save", and "Submit" buttons.



Schematic

Design Layer

Case Designer
↓
Step - Duplicate Search Subprocess

Rules Layer

Flow → [Activity, Properties, Case Match] [Section, Flow Action]

Java HTML CSS XML Javascript JSP

11001010101110001

Skill Mastery

Understand:

- duplicate cases
- duplicate search
- basic and weighting conditions in search duplicates
- add search duplicate cases in case designer
- add search duplicate cases in the flow editor



Best Practices for using Search Duplicate Cases

- Use a basic condition to filter out too many duplicates and improve run-time performance.
- Use the clipboard tool to trouble shoot duplicate search issues.
- Investigate the Case Match rule created to help resolve problems during development.



SKILL
LESSON

Sending Correspondence



Overview

The Pega Platform can send automated correspondence on behalf of organizations that need to communicate with customers, end users and other internal employees, and depend on timely communication to establish a shared understanding of transactions or assignments.



Use Case

- During case processing, an end user may need to be notified of work assigned to them.
- Automatically send correspondence to customers to inform them of status updates throughout the processing of a case.
- Create timely communications to establish a shared understanding of transactions or assignments.



Correspondence

Definition

- An outgoing email, letter, fax, or text message that is produced by the system and its users.
 - The system can send automated email correspondence to all relevant parties.
- Typically associated with a work item and can include text and images.
- Useful to send a notification to one or more of the work parties in a case.
 - Work party is all the parties who are expected to work on cases of the associated case type.



Effective communication

Description

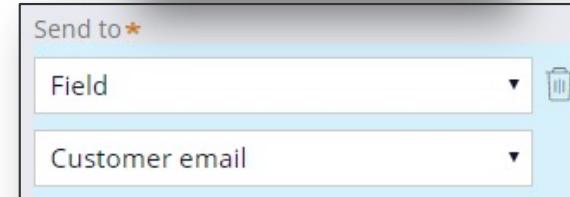
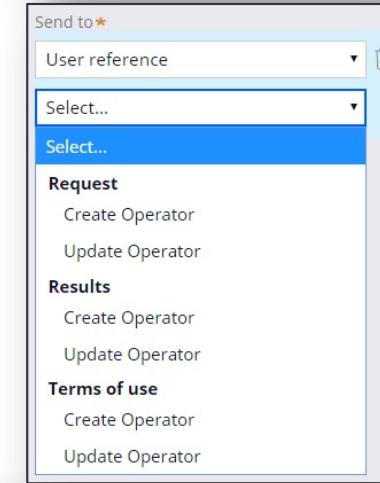
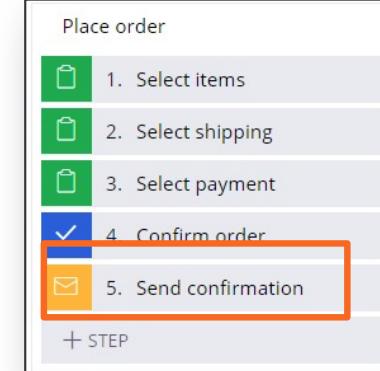
- To achieve effective communication, answer these three questions:
 - Who needs the communication?
 - How will the communication be composed?
 - When does the communication need to be sent?



Identify who

Definition

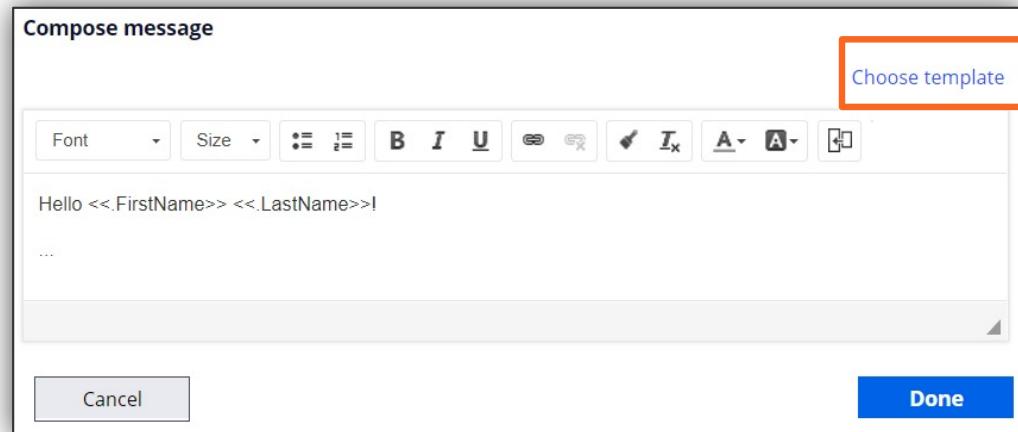
- When configuring correspondence, first determine, *With whom do I need to communicate?*
- Sending to:
 - **Email address**
 - Must update the application if email address changes.
 - **Field**
 - A property with email address as the value.
 - **User reference**
 - A list of existing users in an application.
 - **Participants**
 - people, businesses, and organizations that are involved in a case.



Identify how

Definition

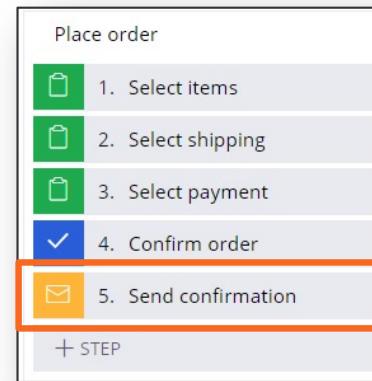
- Determine *How will the communication be composed?*
- Pega Platform provides a rich text editor to create formatted email correspondence.
- The editor allows reuse of case data in the email.
- Pega Platform provides out-of-the-box templates that use case processing data, such as the case ID and case status.
Templates such as:
 - Confirmation and thank you notices
 - Past deadline and goal notifications
 - Rejection and resolution details
 - Status updates



Identify when

Definition

- Determine *When does the communication need to be sent?*
- Pega Platform simplifies sending correspondence by allowing configuration of the **Send email automation step** to a case.
 - Sends an email automatically to the selected parties.
- Another option is to configure email notifications for case level notifications.
 - Automatically sends a notification when an assignment in the case is routed to a user worklist.
 - Automatically configured to send to the user associated with the worklist.



A screenshot of the "Notifications" configuration screen. It shows the "Email notifications" section with a checked checkbox labeled "Email user when assignment is routed to worklist". This checkbox is highlighted with a red border. Below it, there are options for "Email" (set to "Custom") and "Subject" (a text input field). At the bottom, there are "Message" and "Compose message" buttons, and a note stating "Message preview will appear here".

Correspondence types

Definition

- Pega Platform provides 4 correspondence types to send automated communication with users:
 - Email
 - Outgoing message may contain HTML, attachments or only text.
 - System must contain an email account data instance and connect to a mail server.
 - SMS phone text
 - Short outgoing text messages to be sent to digital cell phones.
 - Fax
 - Outgoing letters to be sent by fax transmission.
 - Printed letter
 - Outgoing postal letters to be printed.

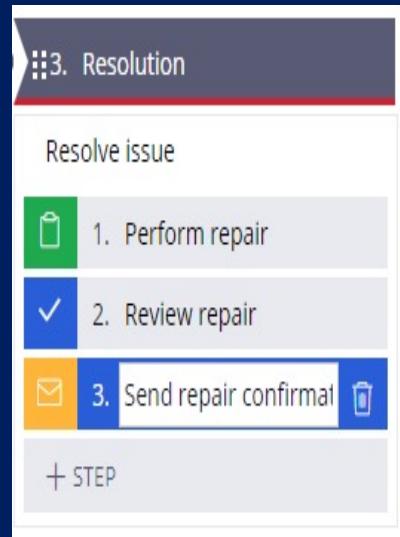


Configure send email step

Navigation

From App or Dev Studio:

1. Click +Step > More > Automation.
2. Then, select the Send Email step.
3. Configure the step by selecting the Send to option.
4. Enter a subject for the email.
5. Select an existing message or click Compose to create a new message.

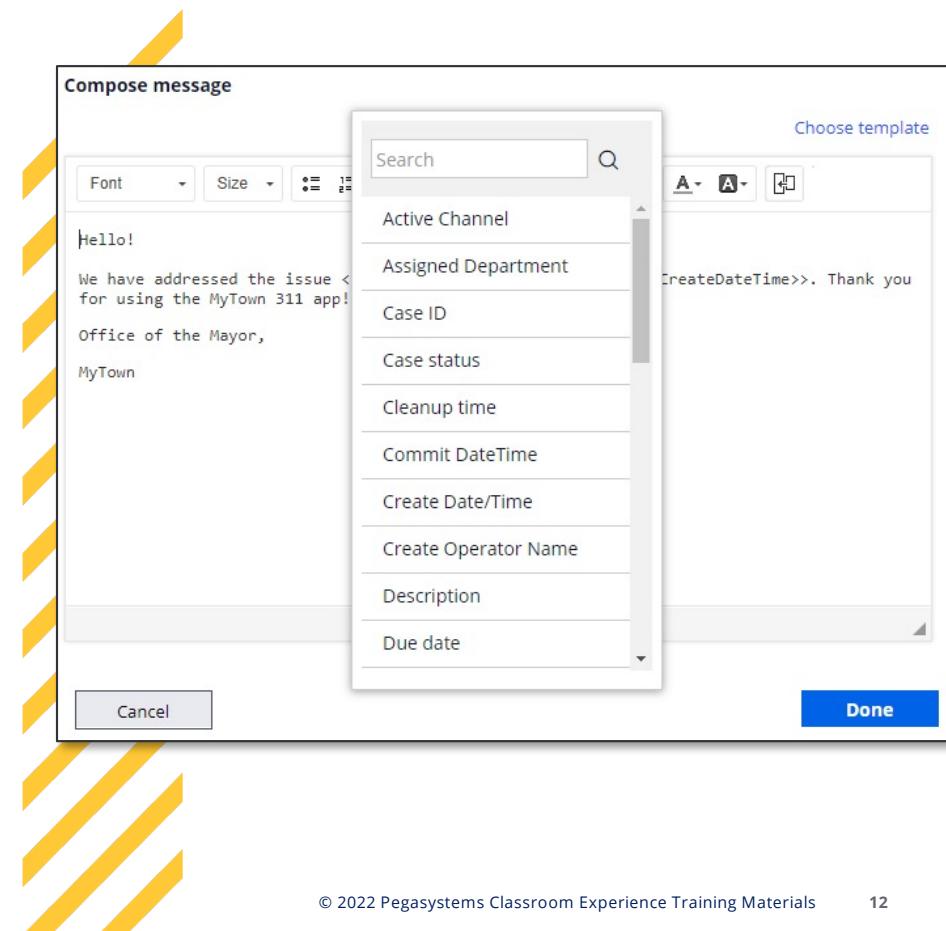


The screenshot shows the "Send to" configuration dialog. It includes fields for "Field" (set to "Requestor Email") and "Subject" ("Your service request has been completed!"). Below these, there is a "Compose" section with a preview message: "Hello! We have addressed the issue <<.pyID>> that you logged on <<.pxCreateDateTime>>. Thank you for using the MyTown 311 appl". The dialog also features a toolbar for font and size selection, and buttons for "Cancel" and "Done".

Compose message

Description

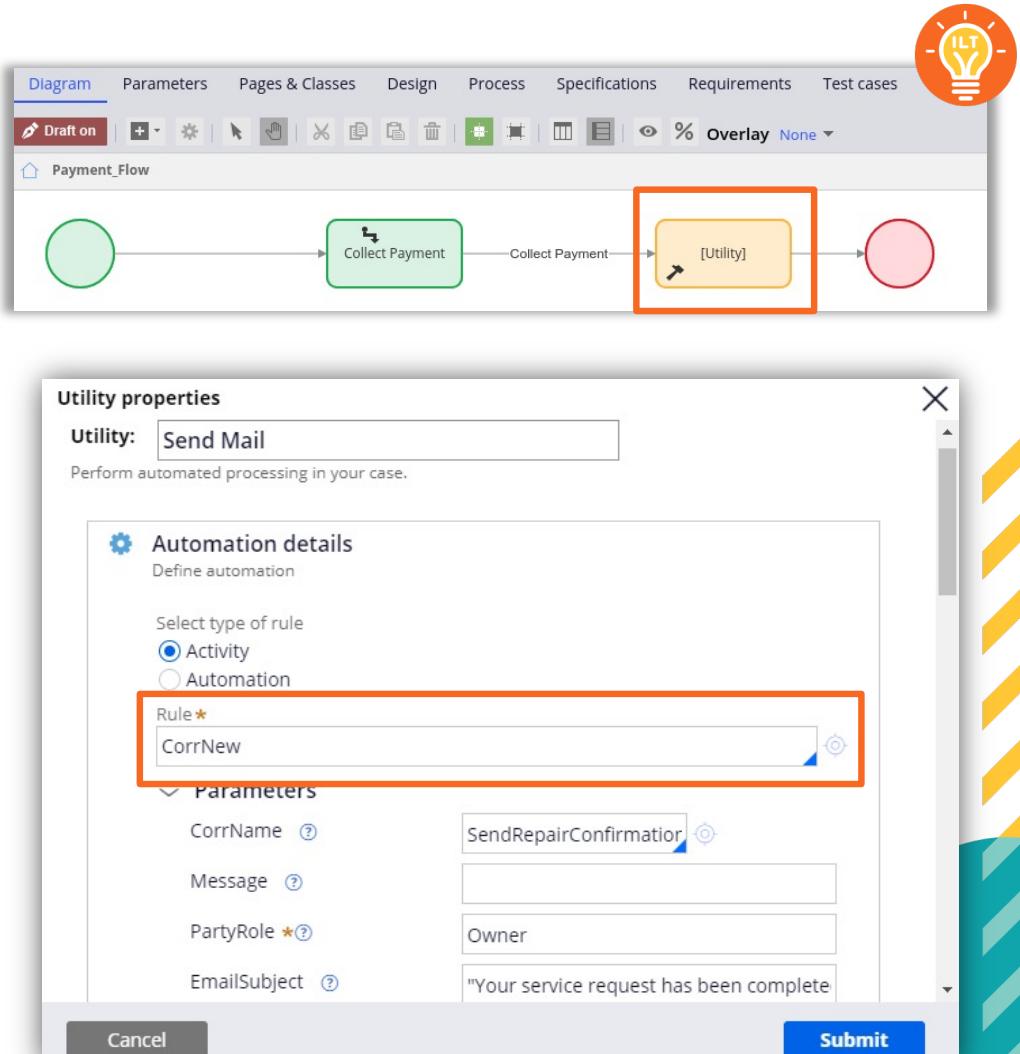
- Further configure the correspondence rule from Dev studio using the rich text editor.
 - Toolbar is available to apply formatting and other edits such as bold, underscore, and italics.
 - Allows adding sections (views), paragraphs, fragments (boilerplate HTML text), and other correspondence rules.
 - The editor also allows adding text directly or dynamic fields to generate customized messages.
 - Java Server tags also allow correspondence to incorporate property values and calculations.



Utility shape

Description

- Add a **utility** shape in the process flow to send other types of correspondence such as mail, fax, and text messages.
- Similar to the Send Email shape, use a Utility shape in a process flow to automatically send correspondence.
- A Utility shape configured with the **CorrNew** activity rule offers greater flexibility than the Send Email step.





Add the utility shape

Navigation

From Dev Studio:

1. Select the case type.
2. Open the process diagram.
3. Right-click and add the utility shape.
4. Double-click the shape.
5. Edit the utility properties.
6. Enter CorrNew in the Rule field.

Utility properties

Utility: Send Mail
Perform automated processing in your case.

Automation details
Define automation

Select type of rule
 Activity
 Automation

Rule *****
CorrNew

Parameters

CorrName ?	SendRepairConfirmation
Message ?	
PartyRole *?	Owner
EmailSubject ?	"Your service request has been complete"

Cancel **Submit**



Configure correspondence rule

Navigation

From Dev Studio:

1. Select App explorer.
2. Expand the class/case type.
3. Expand process.
4. Right-click and select Create.
5. Enter a name in the Label field.
6. Select the correspondence type from the autocomplete field.

Correspondence Record Configuration

Identifier Correspondence Edit

Label *

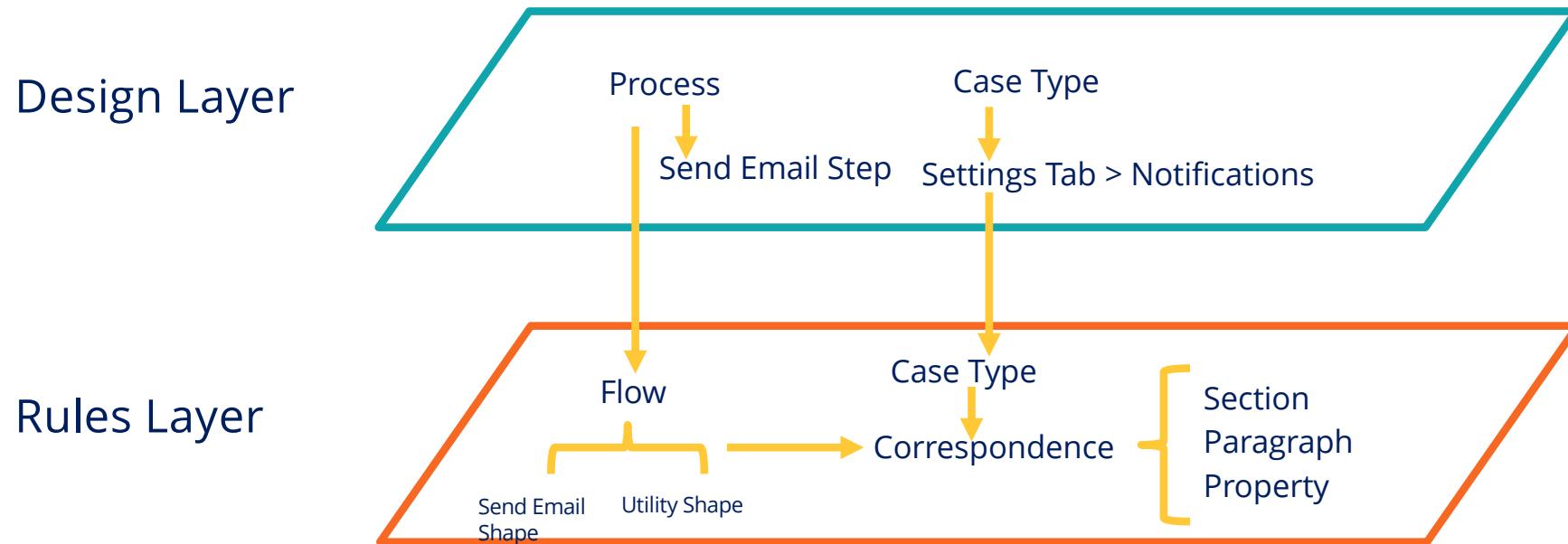
A short description or title for this record

Correspondence Type

Email
Fax
Mail
C...
PhoneText



Schematic



Java HTML CSS XML Javascript JSP

11001010101110001

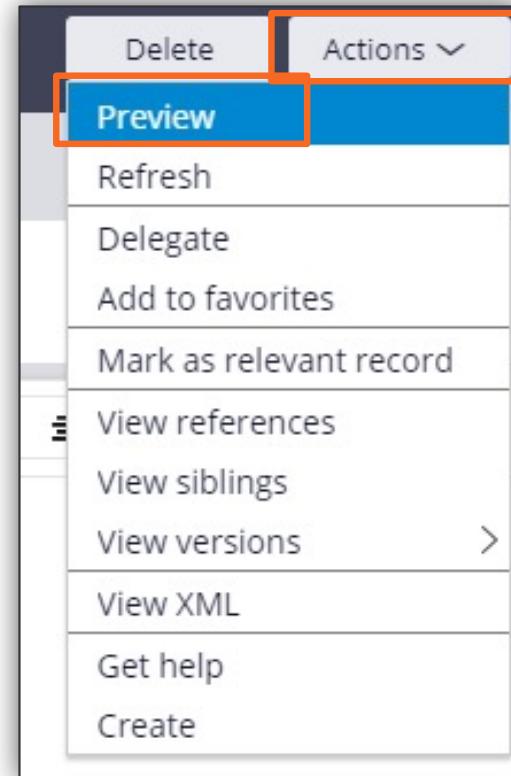


Correspondence

Unit Test and Debug

Select *Preview* to see a display of the email that will be sent.

1. Open the correspondence rule.
2. Select the Actions menu.
3. Click the Preview option.



Skill Mastery

Understand:

- what is correspondence
- how to achieve effective communication and the questions to ask
- what are the correspondence types
- how to compose email messages
- what is the utility shape and how to configure it
- how to configure a correspondence rule

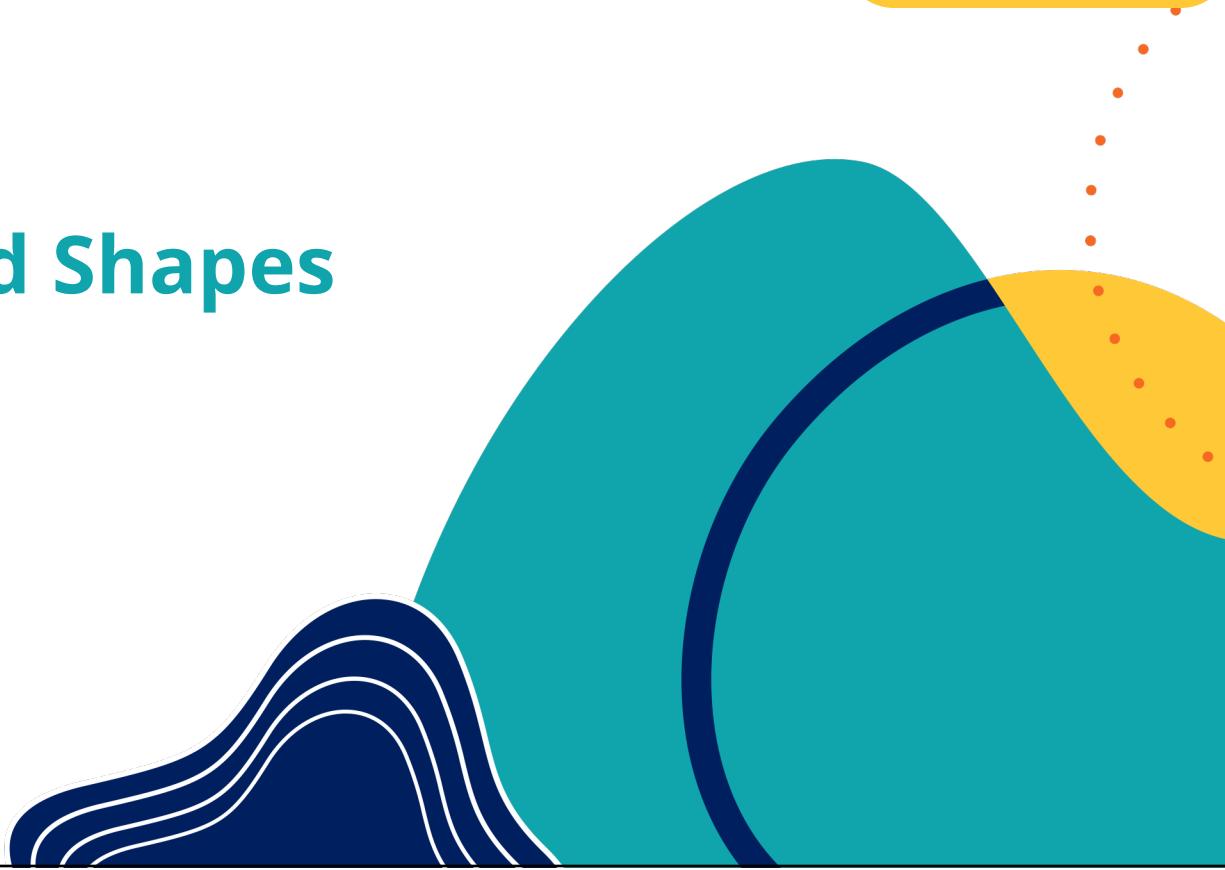
Best Practices

- Use the Send Email step to create quick and simple emails.
- Use a correspondence rule to create more complex notifications such as fax, SMS, email or printed letters.
- Reuse existing correspondence, paragraphs, sections, and fragments when possible.
- Configure a logical condition to include correspondence information where necessary.



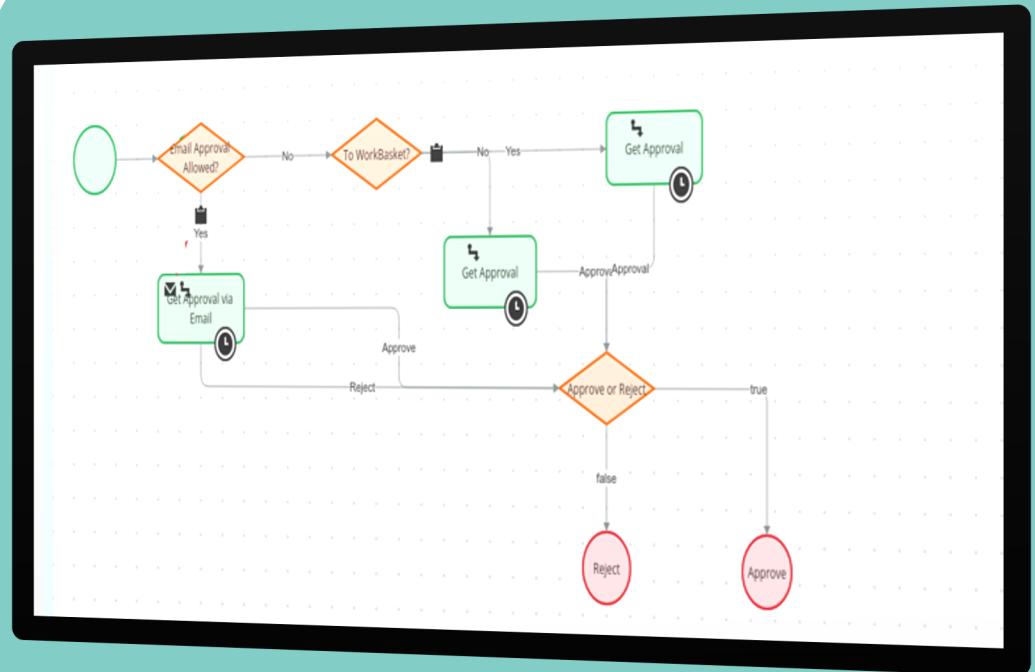
SKILL
LESSON

Decision Points and Shapes



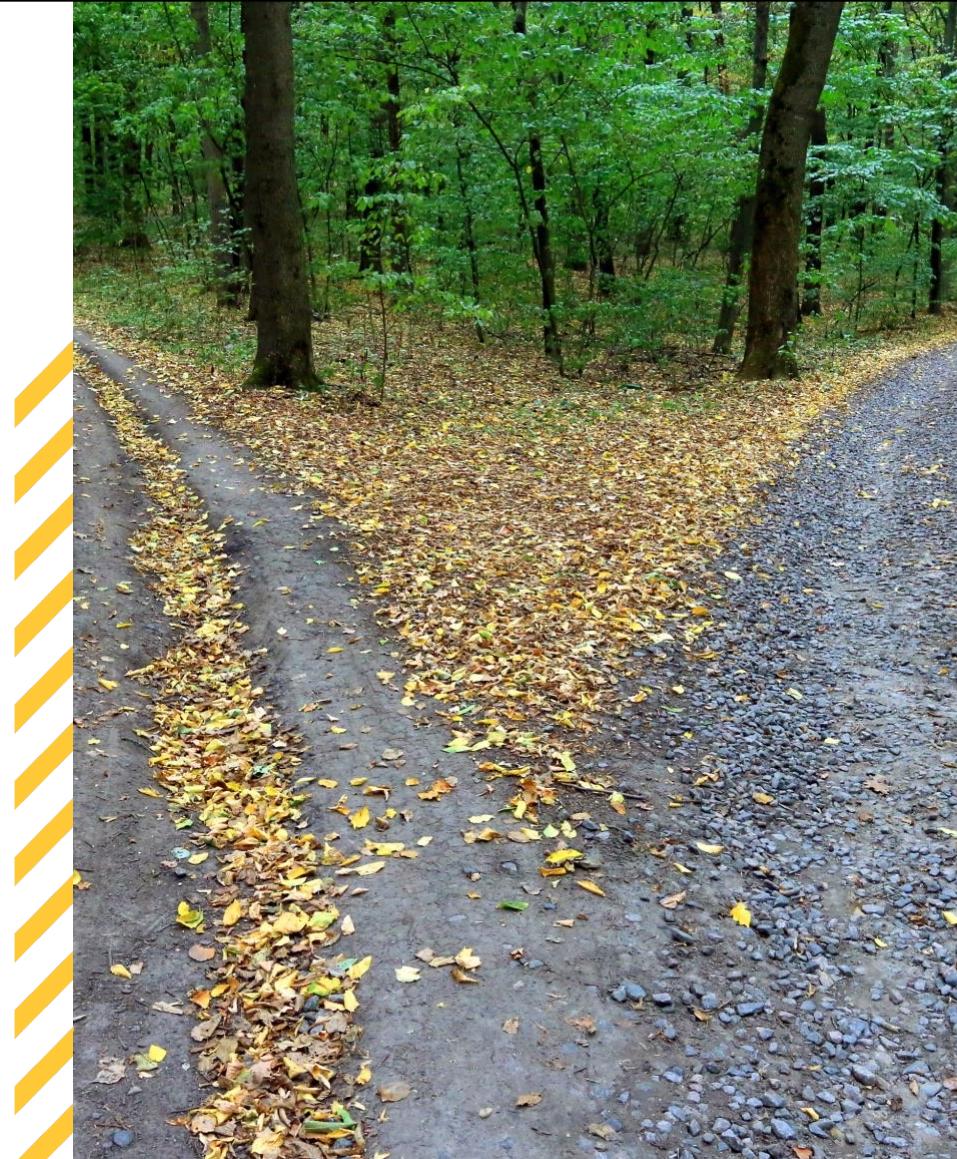
Overview

Decision points in a case type allows support for more than one outcome for a business process and adjusts the case to changing business needs and circumstances.



Use Case

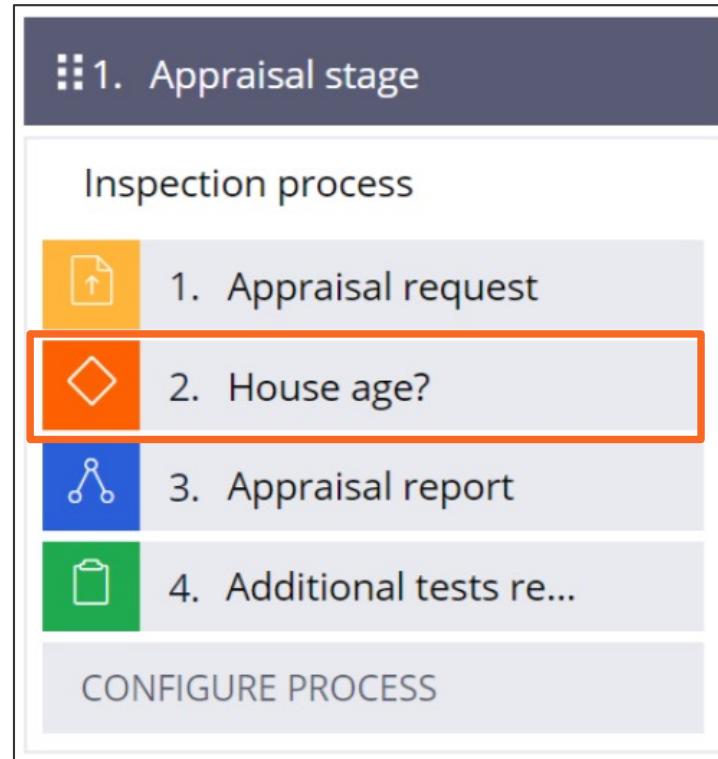
- Case processing requires support for several possible outcomes.
- Automate decisions to reduce processing time and ensure consistency between cases.
- Allow application users to focus on decisions that require human expertise.



Decision points

Definition

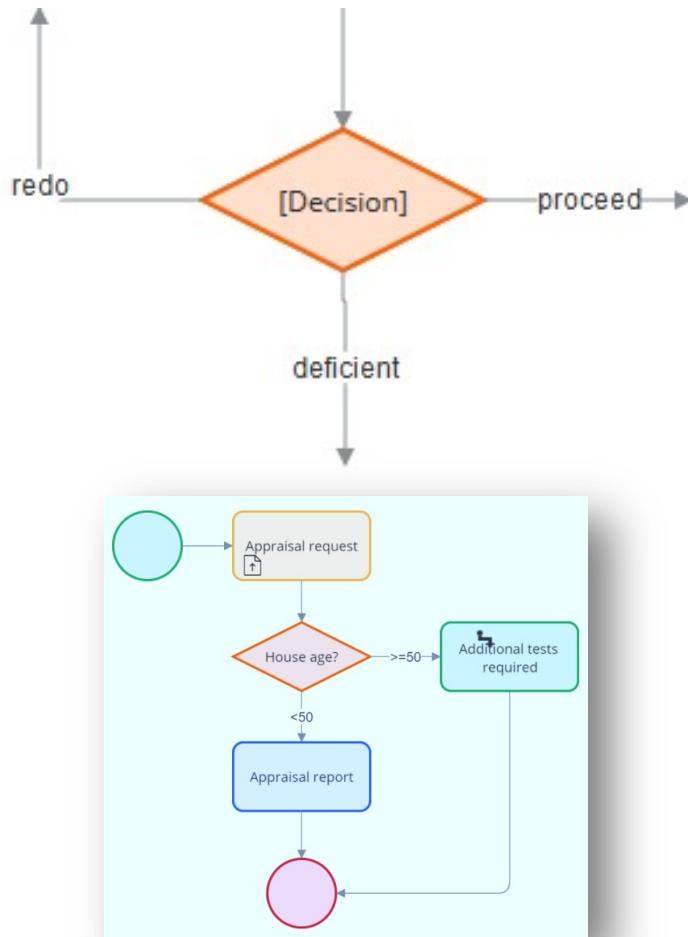
- Evaluates an expression or calls a rule to determine which step is next in the flow progression.
- Decision shapes create alternate paths in a flow because they can have more than one outgoing connector (points).
- Allows creation of different types of processes, or flows, based on the shapes or steps and how a process is integrated within the case life cycle.



Decision step/shape

Description

- Use a Decision shape to add a conditional path to a flow or model more complex use cases.
- Decision shapes are configured to advance a workflow automatically.
- Define conditions that cause the flow to follow different paths.
- Supports more than one outcome.





Types of decision logic

Description

- The decision shape supports multiple configurations and rules.
- Decision rules that can be used with the decision shape include:
 - Boolean expression
 - Fork
 - Map value
 - Predictive model
 - Scorecard
 - Decision table
 - Decision tree

Decision properties

Decision: Right Size
Automatically evaluate a business decision to determine how this case progresses.

Details
Provide a decision to be made in the current process

Type

Decision Table
Boolean Expression
Decision Table
Decision tree
Fork
Map Value
Predictive model
Scorecard Model

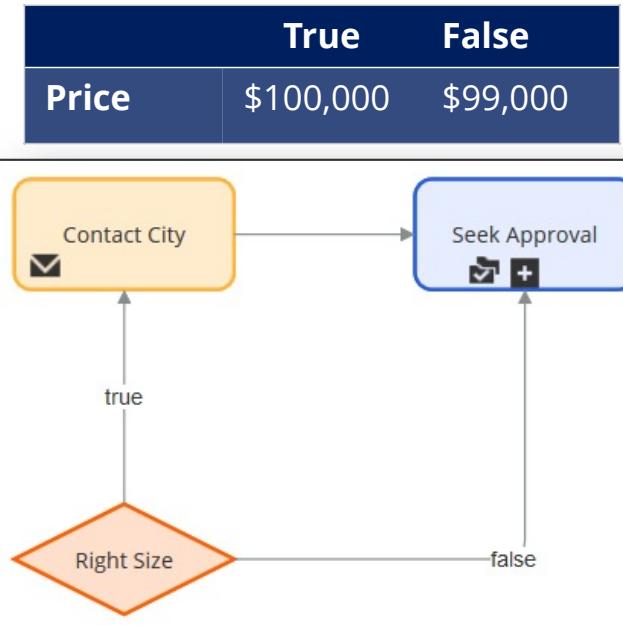
Cancel Submit



Boolean expression

Definition and description

- Returns a true or false value based on an evaluation of conditions and properties defined.
- The Boolean expression uses a *when* condition rule to evaluate comparisons among values of properties, and then returns true or false.
- At run-time, the decision will evaluate the expression and follow the path for the true or false connector.

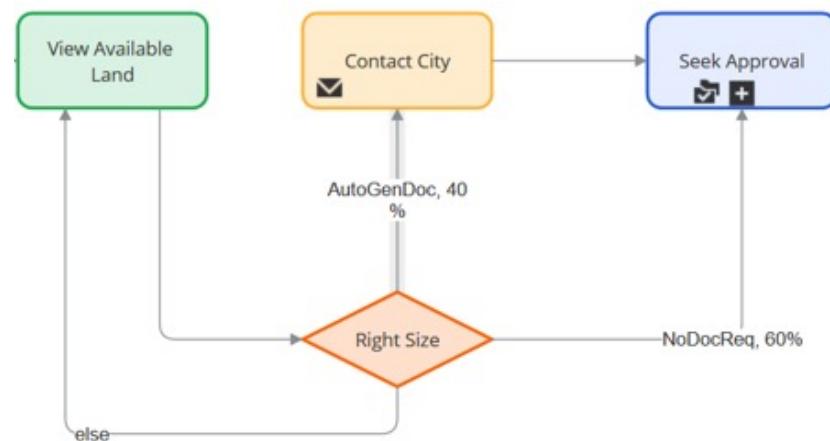




Fork

Definition and description

- Evaluates *when* conditions, expressions, and likelihoods defined to determine which connector advances the flow to the next shape in the sequence.
 - Likelihood as a percentage determines the default path.
 - The highest percentage is the screen displayed. Others are choices under *Actions* menu listed from highest to lowest.
- Connectors - executed in ascending order based on the likelihood value.
 - *Else* path is followed if all *when* rules evaluate as false.

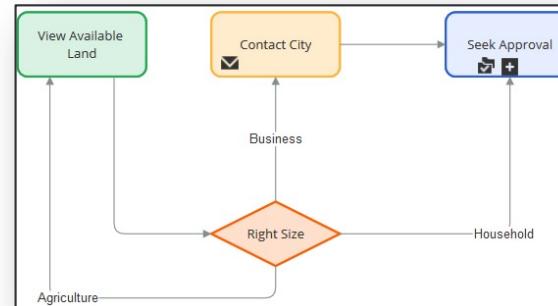




Map Value

Definition and description

- Converts one or two input values into a single-value result.
- Use a map value to create a table of number, text, or date ranges that converts one or two input values.
- Use a map value to record decisions based on one or two ranges of an input value.
- Map value rules can be updated as needed to reflect changing business conditions by someone other than a skilled developer.



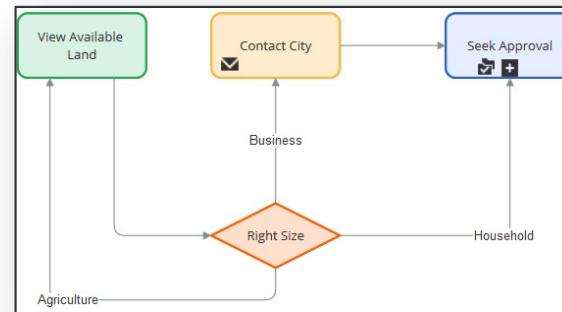
Land Type Row	=Commercial	=Farming	=Private
=Commercial	"Business"	"Agriculture"	"Business"
=Farming	"Agriculture"	"Agriculture"	"Agriculture"
=Private	"Business"	"Agriculture"	"Household"



Scorecard

Definition and description

- Use scorecards to derive decision results from several factors.
- A scorecard creates segmentation based on one or more conditions and a combining method.
- The output of a scorecard is a score and a segment defined by the results.
- Map the score-based segmentation to results by defining cutoff values to map a given score range to a result.



Scorecard [Results](#) Pages & Classes Specifications History

Map score range to segment results.

Refresh

Minimum score Maximum score
0.0 2000.0

[+](#)

Result*	Cutoff value	Interval
ResultLT80	0	< 0
ResultEQ20	20	[0;20)
Default	Otherwise	≥ 20



Predictive Model

Definition and description

- Uses customer data to develop models that can predict customer behavior.
 - Use Prediction studio to create a predictive model and then add the model to the Model tab of the rule form.
 - The model tab will display the output fields of the Predictive Model Markup Language (PMML) model
 - Third-party models in PMML format can also be imported.

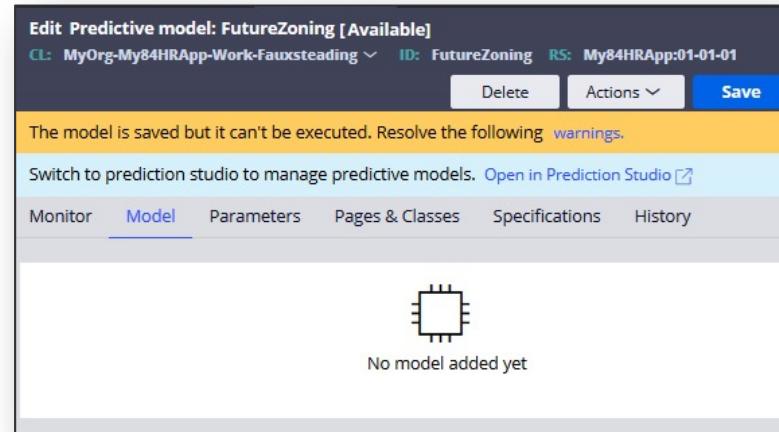
Edit Predictive model: FutureZoning [Available]
CL: MyOrg-My84HRApp-Work-Fauxsteading ID: FutureZoning RS: My84HRApp:01-01-01

The model is saved but it can't be executed. Resolve the following [warnings](#).

Switch to prediction studio to manage predictive models. [Open in Prediction Studio](#)

Monitor **Model** Parameters Pages & Classes Specifications History

No model added yet



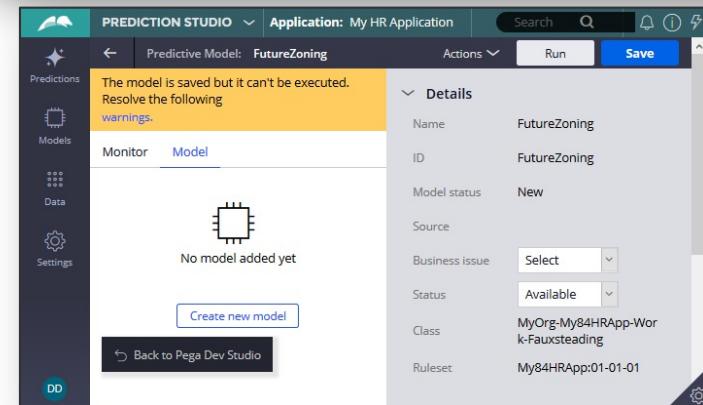
PREDICTION STUDIO Application: My HR Application

The model is saved but it can't be executed. Resolve the following [warnings](#).

Details

Name	FutureZoning
ID	FutureZoning
Model status	New
Source	Select
Status	Available
Class	MyOrg-My84HRApp-Wor k-Fauxsteading
Ruleset	My84HRApp:01-01-01

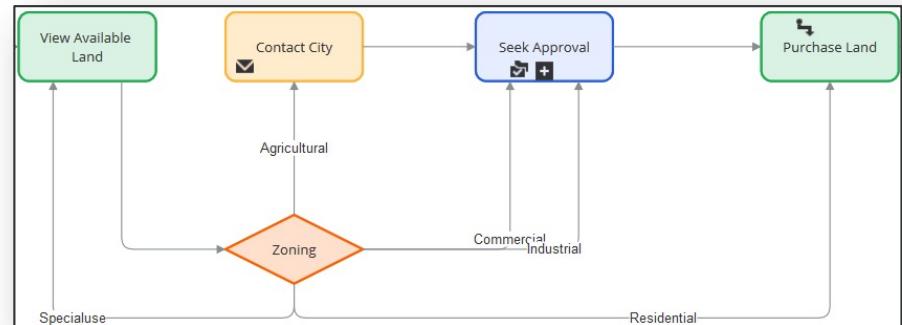
Back to Pega Dev Studio



Decision Table

Definition and description

- Use to derive a value that has one of a few possible outcomes, where each outcome can be detected by a test condition.
- The table consist of two or more rows, each containing test conditions, actions (optional), and a result.
- At run time, the system evaluates the rows starting at the topmost row:
 - Conditions in a row evaluate to false, processing continues with the next row, Actions and Return ignored.
 - All the conditions in a row evaluate to true, Actions and Return are processed.
- Evaluate All Rows
 - not selected, processing ends, returns the value in the Return column as the value of the entire rule.
 - is selected, processing continues for all remaining rows, performing the Actions and Return for any rows which are true.

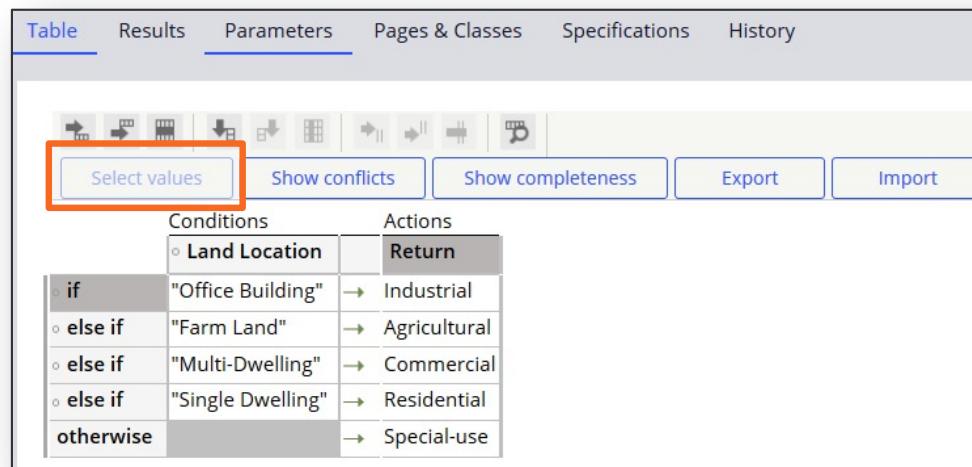


Conditions		Actions
if	"Office Building"	→ Industrial
else if	"Farm Land"	→ Agricultural
else if	"Multi-Dwelling"	→ Commercial
else if	"Single Dwelling"	→ Residential
otherwise		→ Special-use

Decision table toolbar

Implementation

- Decision tables resemble a spreadsheet with rows and columns where the buttons on the toolbar allow inserting and deleting rows and/or columns.
- Select values
 - Option is enabled to select one or more values for a property.
 - The list displays values that were entered for the property in a case.



The screenshot shows a decision table interface with a toolbar at the top. The toolbar includes buttons for Table, Results, Parameters (which is selected), Pages & Classes, Specifications, and History. Below the toolbar is a row of icons: a plus sign, a minus sign, a copy icon, a paste icon, a search icon, a magnifying glass, a refresh icon, a save icon, and a print icon. A red box highlights the 'Select values' button, which is located next to the search icon. To the right of the 'Select values' button are five other buttons: 'Show conflicts', 'Show completeness', 'Export', and 'Import'. The main area of the interface displays a decision table with two columns: 'Conditions' and 'Actions'. The 'Conditions' column contains radio buttons for 'if', 'else if', 'else if', 'else if', and 'otherwise'. The 'Actions' column contains corresponding values: 'Return', 'Industrial', 'Agricultural', 'Commercial', 'Residential', and 'Special-use'. The 'Return' value is highlighted with a gray background.

Delegation options

Implementation

Complete the fields on the Results tab to restrict the possible values returned by a decision table.

- Evaluate all rows
 - Select this option to process each row in the table.
 - The *if/else* statements change to *when* statements

When you delegate a decision table, the following options in the Results tab controls the options that are available to the process owner:

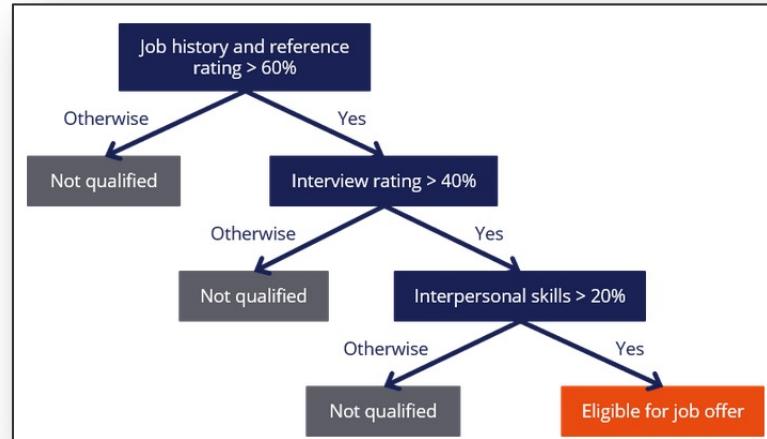
- Allowed to update row and column layout
 - Disable to prevent column manipulation.
- Allowed to change property sets
 - Disable to prevent cell value updates.
- Allowed to build expressions
 - Disable to hide the Expression Builder icon.
- Allowed to return values
 - Enable to indicate a return value can be assigned to a property.
 - Disabled when the Evaluate all rows is selected.

Table	Results	Parameters	Pages & Classes
Delegation options			
<input checked="" type="checkbox"/> Evaluate all rows			
<input checked="" type="checkbox"/> Allowed to update row layout			
<input checked="" type="checkbox"/> Allowed to update column layout			
<input checked="" type="checkbox"/> Allowed to change property sets			
<input checked="" type="checkbox"/> Allowed to build expressions			
<input checked="" type="checkbox"/> Allowed to return values			

Decision tree

Description

- Used to define comparisons by using a tree structure.
- Contains nested if... then... else conditions to specify a series of tests performed on property values to allow an automated decision.
 - To configure, in the **then** list, select *continue*.
 - Continue* causes the next row of the decision tree to be nested.
 - Each indent level supports comparisons against a single value.



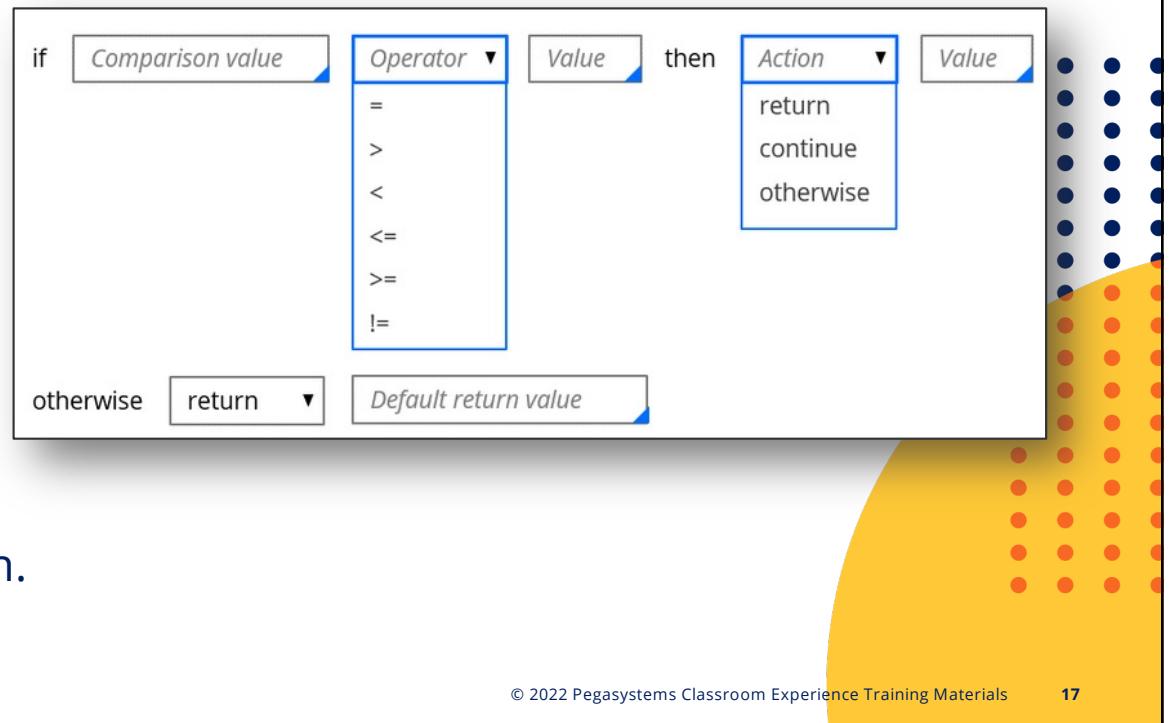
The screenshot shows the "Decision" tab of a Pega configuration interface. The interface includes the following elements:

- Header tabs: Decision, Configuration, Parameters, Pages & Classes, Test cases.
- Buttons: Show Conflicts, Show Completeness.
- Decision logic:
 - if Expected Price > 150000 then return "Too expensive"
 - if Expected Price < 99000 then return "Too cheap"
 - otherwise RETURN Great price

Tree conditions

Implementation

- Configure single *if* statements with return values.
- Configure nested *if* statements with *continue* statements.
- **Comparison value** is a property or expression.
- Comparison **operators** are conditional symbols.
 - (e.g. <,>, <=, >=, =, !=)
- **Value** is a fixed value, property or express
- **Action** can return a result, continue the evaluation, or stop the evaluation.



Decision rule conflicts

The decision table and decision tree rule forms include the ability to test for **conflicts**.

Show conflicts

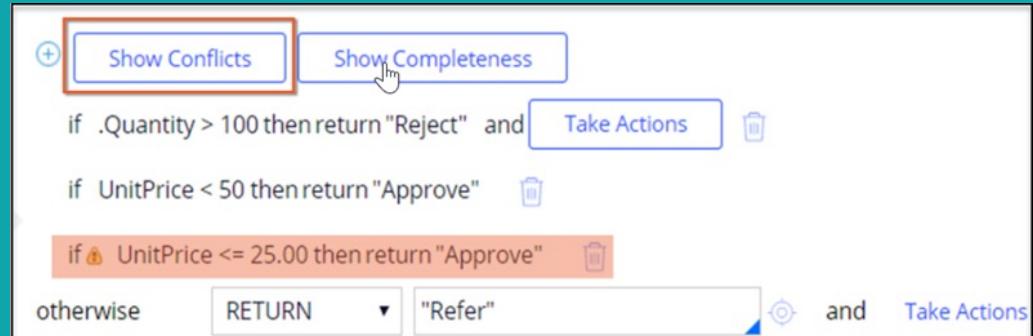
- Identifies potential gaps in the decision rule execution by identifying conditions that may not be tested during execution.
- A warning is displayed on the row, which causes the conflict to specify the condition that did not evaluate.

Decision Table conflicts

A screenshot of a software interface for managing decision tables. At the top, there's a toolbar with various icons. To the right of the toolbar is a button labeled "Select values" and another labeled "Show conflicts", which is highlighted with a red box. Below the toolbar is a table with three columns: "Conditions" and "Actions". The "Conditions" column contains radio buttons for "Credit Score >" and "Outstanding Balance <". The "Actions" column contains a "Return" action. There are four rows under these conditions:

- if 900 → Approval Level 1
- else if 1000 → Approval Level 2 (This row is highlighted with a red box)
- else if 500 → Approval Level 3
- otherwise → Reject

Decision Tree conflicts



Decision rule completeness

The decision table and decision tree rule forms include the ability to test for **completeness**.

Show Completeness

- Identifies a decision table that has missing conditions or a decision tree that has missing branches.
- Automatically adds rows that cover additional cases, rows can be altered or eliminated.
- You can add returned results as additional rows if the decision rule needs a more detailed evaluation of the values.

The screenshot displays two PEGA rule forms illustrating the 'Show Completeness' feature.

Decision table completeness: This table shows conditions and actions. A row for 'CustomerLevel = Bronze' with a condition ' $<=10000$ ' is highlighted with a cursor, indicating it is being edited. The 'Show completeness' button in the top right is highlighted with a red box.

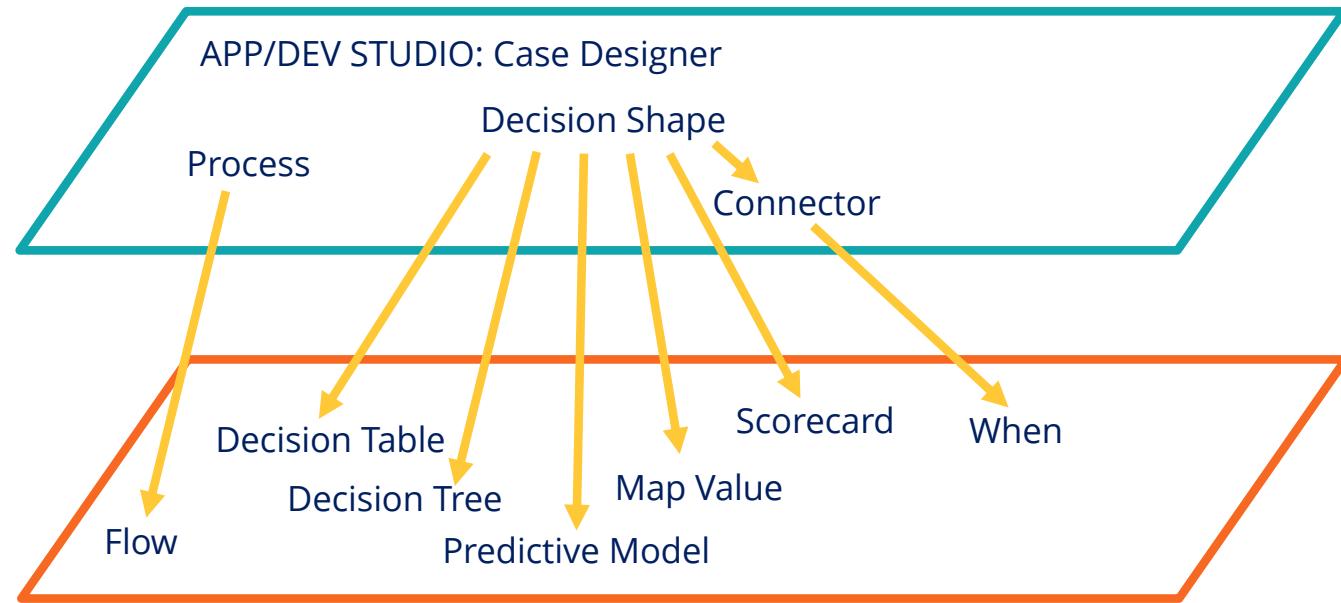
Conditions		Actions	
if	Bronze	100000	→ .03
else if	Bronze	50000	→ .02
else if	Bronze	10000	→ .01
else if	Bronze	$<=10000$	→
else if	Gold	$<=10000$	→
else if	Gold	$<=50000$	→
else if	Gold	$<=100000$	→
else if	Gold	>100000	→
else if	Silver	$<=10000$	→
else if	Silver	$<=50000$	→
else if	Silver	$<=100000$	→
else if	Silver	>100000	→
otherwise			→ 0

Decision tree completeness: This tree diagram shows a root node 'Customer Level = "Bronze" then continue'. Below it, nodes branch into 'Customer Revenue > 100000 then return .03', 'Customer Revenue >= 50000 then return .02', and 'Customer Revenue > 10000 then return .01'. Further branches lead to 'Customer Level = Gold then continue', 'Customer Level = Silver then continue', and finally an 'otherwise RETURN 0' node. The 'Show Completeness' button in the top left is highlighted with a red box.



Schematic

Design Layer



Rules Layer

Java HTML CSS XML Javascript JSP

11001010101110001

Best Practices

- Use the case designer to configure the case life cycle, then work with the shapes and connectors in the flow diagram when a process requires advanced functionality.
- There is no limit for the number of rows in a decision table/tree; however, to avoid slow performance while updating the rule form and to avoid Java errors, limit the decision tables/trees to no more than 300 to 500 rows.
- For decision tables, list the more likely outcomes in rows above the rows for outcomes that are less likely.
- Use the When rule or Boolean configuration for simple logic evaluation.
- Use the Decision Table to compare different values for the same property.
- Use the Decision Tree rule to compare different properties and values.

Skill Mastery

Understand:

- Decision points
- How to configure a decision step/shape
- Types of decision logic
 - Boolean expression (when rule)
 - Fork (when rule)
 - Decision Table
 - Decision Tree
 - Map Value
 - Score card
 - Predictive Model

User Interface



SKILL
LESSON

Introduction to UI

The View presented to the end user to interact with the application



Overview

The User Interface (UI) is the View presented to the end user for interaction with the application.

The View is composed of UI rules configured manually or with visual tools.

The UI and elements represent a view for the end user to perform a task.

This module covers:

- Intro to UI concepts
- UI Structure
- UI Controls
- UI Gallery



Use Case

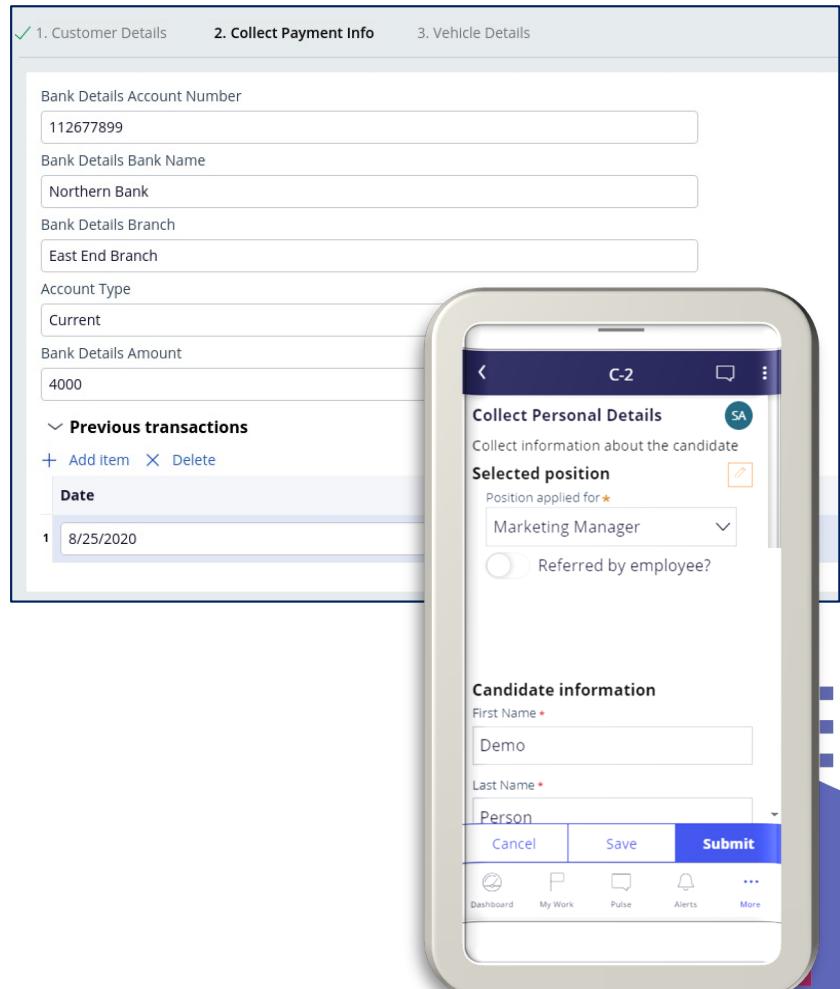
- Use the UI when the Pega application requires human interaction during the lifecycle of a case.
- For example, an individual must login to an application, a sign in screen (UI) is displayed for username and password to access the application features.



User Interface Concepts

Description

- The application presents a User Interface or View to the user to interact with the application
 - Entering values into fields
 - Selecting a value from a dropdown
 - Clicking on a control, a link or image
- Careful consideration of design is important
 - How will the user interact with the UI?
- Presentation consistency
 - What devices will be used by the user?
 - Are dynamic features required
 - Are Responsive UI features required
- Users of Pega applications can interact with the application through a web browser, tablet or mobile application.



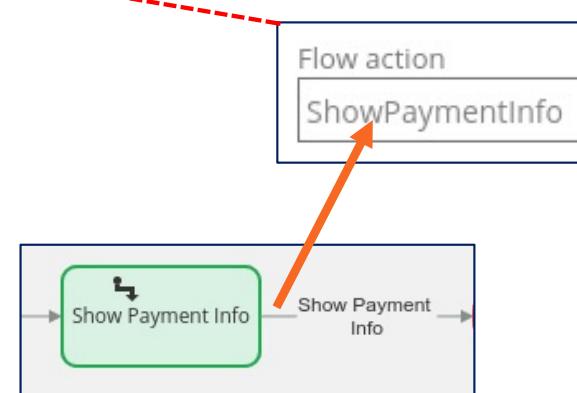
View – User Interface

Definition

- A View is associated with a collect information step (assignment shape) in the flow rule of a case
- A Flow Action is associated with the Connector leaving the shape in the Main or Sub flow
- A Flow Action is associated with the Assignment shape in a Multi-Step Form
- A View may be constructed using Case Designer or by configuring the rules directly

The screenshot shows a 'Multi-Step Form' titled '2. Collect Payment Info'. It contains several input fields for bank details: 'Bank Details Account Number' (112677899), 'Bank Details Bank Name' (Northern Bank), 'Bank Details Branch' (East End Branch), and 'Account Type' (Current). There is also a field for 'Bank Details Amount' (4000). Below these, there is a section titled 'Previous transactions' with a table:

Date	Amount
1 8/25/2020	500





View – User Interface

Description

- The View itself is composed of a hierarchy of rules
- These UI rules are organized into a nested structure
 - Portal rules
 - Harness
 - Section
- The rules are accessible using Live UI

1. Customer Details 2. Collect Payment Info 3. Vehicle Details

Bank Details Account Number
112677899

Bank Details Bank Name
Northern Bank

Bank Details Branch
East End Branch

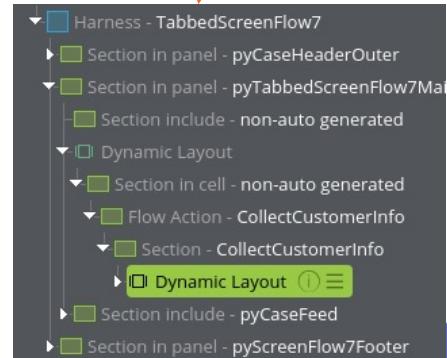
Account Type
Current

Bank Details Amount
4000

▼ Previous transactions

+ Add Item X Delete

Date	Amount
1 8/25/2020	500





Live UI

Definition

- Live UI is a developer tool to examine and edit the rule structure of your user interface designs
 - Quickly identify and open a UI rule
 - Change the presentation of a layout or a field
 - Add or delete elements
 - Move elements within the structure
- Available from Dev and App Studio

The screenshot shows the Pega Live UI interface. On the left is a form titled "Evaluate and Sell a Vehicle (E-5)" with fields for "Account Number", "Bank Name", "Branch", "Account Type", and "Amount". Below the form is a navigation bar with steps: REQUEST, MAN..., EXPENSE, DEAL, and HAN... On the right is a tree view of the rule structure, starting with "Skin - pyEndUser" and "dynamic Layout". A red arrow points from the "Live UI" tab at the bottom to the tree view, indicating that the tree view is a feature of the Live UI tool.

Dev Studio

Navigation



The diagram illustrates the navigation flow through various Pega Dev Studio components:

1. Connector
2. Flow Action Rule Form
3. Flow Action Rule Form
4. Section Rule
5. Section Rule
6. Section Rule Form
7. Cell with UI Control
8. Property Rule

Left Panel (Flow Designer): Shows a process flow starting with a green initial state, followed by a green "Select services" connector, and ending with a red final state.

Middle Panel (Flow Actions): Displays the "Flow" section of the navigation tree, expanded to show "PerformService", "pyStartCase", "Flow Action", "SelectServices", and "Section". The "SelectServices" item is highlighted in blue.

Bottom Left Panel (Section Configuration): Shows the configuration for the "SelectServices" section, including "Page context" (Use current page context), "Applies to" (GoGoCM-GoGoRoadCM-Work-Service), and "Section" (SelectServices).

Bottom Center Panel (Section Rule Editor): Displays the "Edit Section: Services [Available]" screen. It shows a table structure with columns: Service, Unit cost, Quantity, and Line total. A red box highlights the "Unit cost" column header. The "Cell Properties" panel on the right is open, showing settings for the "UnitCost" property, including "Default value" set to ".UnitCost" and "Visibility" set to "Always".

App Studio

Navigation

The screenshot shows the Pega App Studio interface for creating a service workflow. On the left, the navigation bar includes icons for Overview, Case types, Data, Interfaces, and Pages. The main area is titled "APP STUDIO" and "Service". The "Workflow" tab is selected, showing a "Case life cycle" with a single step labeled "1. Service". This step has a "Perform service" action with a sub-step "1. Select services". A red arrow points from the "Select services" button in the workflow step to a modal window titled "Select services". The modal contains a search bar and three categories: Fields, Views, and Data types. To the right of these categories are two columns of fields and their types:

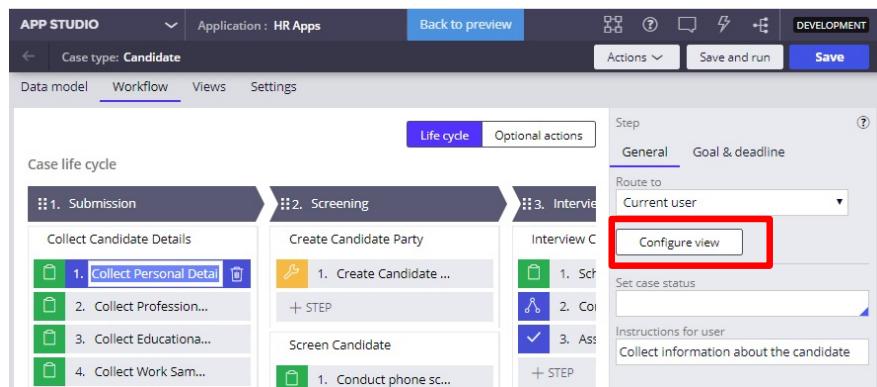
Field	Type
Services	Field group (list)
Service	Text (single line)
Unit cost	Currency
Quantity	Integer
Line total	Currency

Design wizard (1 of 3)

Implementation

Click Configure view:

- To access the design wizard and create fields/properties under the Data model tab
- Fields/properties can be associated with UI controls such as text fields, dropdown list, buttons, etc.



Edit case type: UI_Controls

Data model Workflow Views Settings Test cases

Show: System fields Reusable fields [?](#)

Fields Validations

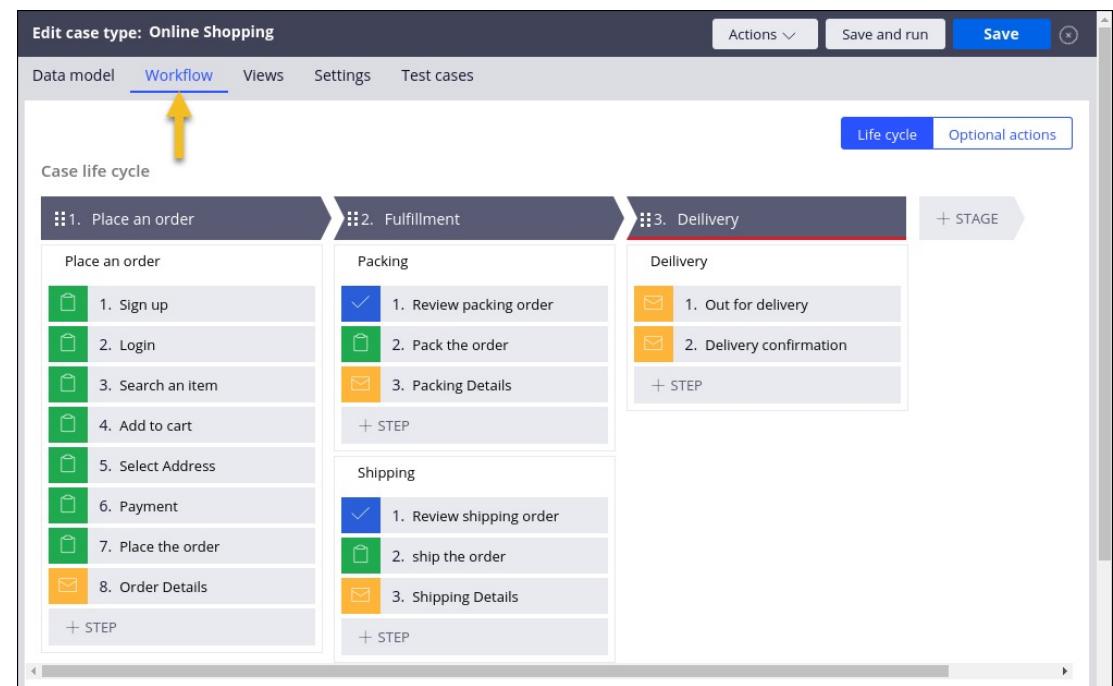
Search [Search](#)

Name	ID	Type	Options
First Name	FirstName	Text (single line)	⚙️ trash
Last Name	LastName	Text (single line)	⚙️ trash
Gender	Gender	Picklist	Radio buttons ⚙️ trash
Date of Birth	DateOfBirth	Date only	⚙️ trash
Contact Number	ContactNumber	Phone	⚙️ trash
Email address	EmailAddress	Email	⚙️ trash
Communication Address	CommunicationAddress	Text (paragraph) Plain text	⚙️ trash
Father Name	FatherName	Text (single line)	⚙️ trash
Mother Name	MotherName	Text (single line)	⚙️ trash
Required Course for Admission	RequiredCourseForAdmission	Picklist	Drop-down list ⚙️ trash
Instructions for user		Collect information about the candidate	
+ Add field			

Design wizard (2 of 3)

Implementation

- The case life cycle can be designed by clicking on the Workflow tab.
- Assignment steps (collect information) represent a View or UI.



Design wizard (3 of 3)

Implementation

- By using the design wizard, you can view the default views and views created by the developer.
- From the Views tab, you can create a new view.

The screenshot shows the Pega Platform interface for editing a case type named "Online Shopping". The top navigation bar includes tabs for "Data model", "Workflow", "Views" (which is highlighted in blue), "Settings", and "Test cases". On the far right of the top bar are "Actions", "Save and run", and a "Save" button. Below the tabs is a toolbar with icons for search, calendar, and other functions. The main area displays a grid of 12 cards representing different views or steps:

View Type	Description	Last Updated
Create new view	(Placeholder)	-
Cancel request	(Placeholder)	-
Case wide Optional Process Step	1 month 7 days ago	-
Case wide Optional User Action	1 month 7 days ago	-
Create	C	-
Edit	C	-
Login	1 month 3 days ago	-
Review	C	-
Search an item	1 month 4 days ago	-
(Placeholder)	-	-
(Placeholder)	-	-

A yellow arrow points to the "Views" tab in the top navigation bar. The bottom right corner features the PEGA logo and a decorative graphic.

Cosmos Design System

Description

- Provides a library of reusable UI components based on 35+ years of business application experience.
- Intuitively presents data and actions to business users.
- Extensible by front-end developers: Publish new components by using the open-source React UI.
- Ensure that your application includes a modern, responsive, and consistent UI by using the out-of-the-box Theme-Cosmos in your application stack

The image contains two side-by-side screenshots of a Pega application management interface. Both screenshots show the 'Edit Application' screen for different applications, illustrating the use of the Theme-Cosmos UI.

Screenshot 1: Edit Application: Hire Me

This screenshot shows the 'Edit Application: Hire Me' screen. The top bar displays the application ID as 'ID: ApplicantMgr * 01.01.02' and the rule set as 'RS: ApplicantMgr [Edit]'. A message at the top right states 'This record has 4 info warnings'. The navigation tabs include 'Definition', 'Cases & data', 'Application wizard', 'Documentation', 'Integration', and 'Security'. Below the tabs, a section titled 'Built on applications' shows a table with one entry:

Name	Version
1 Theme-Cosmos	03.01

Screenshot 2: Edit Application: Cosmos Rules

This screenshot shows the 'Edit Application: Cosmos Rules' screen. The top bar displays the application ID as 'ID: Theme-Cosmos * 03.01' and the rule set as 'RS: Pega-ProcessCommander [Edit]'. A message at the top right states 'This record has 1 severe or moderate warning and 4 info warnings'. The navigation tabs are identical to the first screenshot. Below the tabs, a section titled 'Built on applications' shows a table with one entry:

Name	Version
1 PegaRULES	8

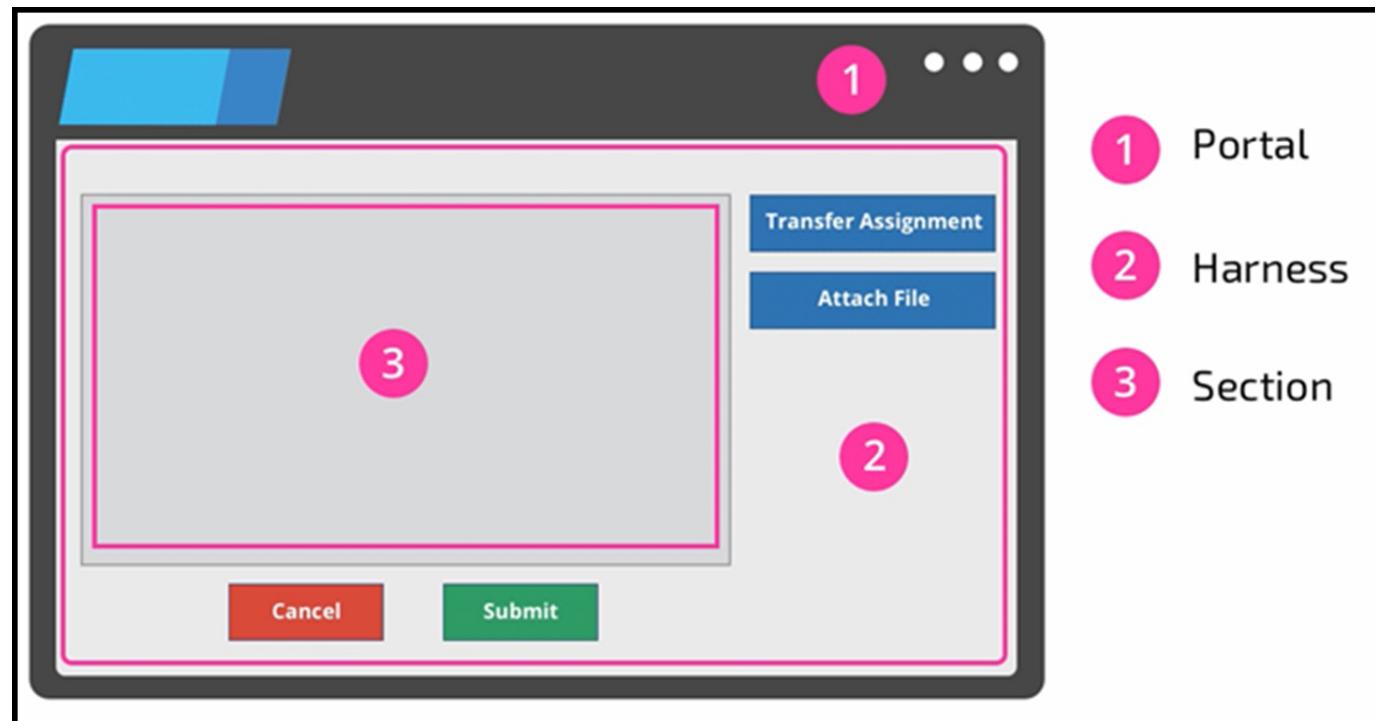
On the right side of the second screenshot, there are additional sections for 'Development branches' (which is empty) and 'Application rulesets' (which lists '1 Theme-Cosmos:03-01').



UI Rule Structure

Definition

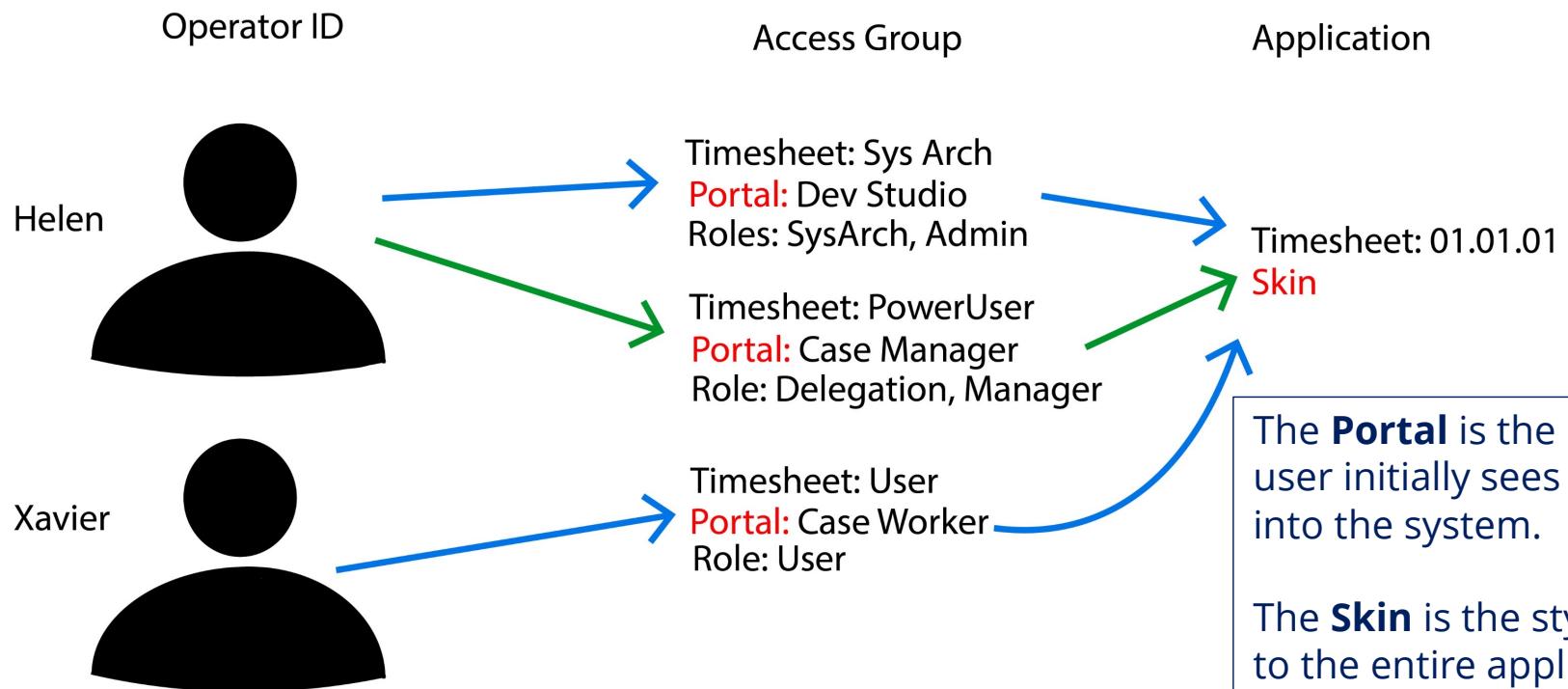
The View is composed of a hierarchy of rules, the Portal, Harness and Section





User Portals

Description



The **Portal** is the landing page a user initially sees when they log into the system.

The **Skin** is the styling applied to the entire application.



Portal Rules

Description

- The UI is built inside a **portal rule** which sets up workspaces for users
- Pega provides standard portals rules tailored to specific users
- **App Studio** and **Dev Studio** — Citizen developers and System architects develop applications
- **Case Worker** and **Case Manager** — Business users who create, update, route, and resolve cases

The screenshot displays the Pega Platform interface with three main windows:

- DEV STUDIO:** Shows the "Evaluate and Sell a Vehicle" case type selected. The left sidebar includes sections for Case types, Data types, Records, and Favorites.
- APP STUDIO:** Displays the "Evaluate and Sell a Vehicle" application workspace. It shows the "Properties" tab with details like "DEVELOPMENT http://localhost:9080/pweb/development", "Version: Pega platform 8.4.1", and "Personas: Manage (0)". The "New features include" section highlights "One journey at a time".
- PEGA:** Shows the "Case Manager" workspace. It includes a "Summary for Properties" section with a welcome message, a "Case stages for RV Calculation" section with a "New (0)" button, and a "My Worklist (4)" section.



Harness Rules

Description

- Portal rules reference **harness rules** for content
- **Harness** - frames the work areas in which users process cases and provide tools that let users manage the assignment process
- A harness gives users the ability to cancel, save, or submit their work
- While working in the assignment, users can transfer their assignments to other users, attach files to the case, or send email correspondence



Flow Action Rules

Description

- When a case reaches an assignment, the harness presents a **flow action** that allows users to perform a task defined for the assignment
- The flow action references a **section rule** to display the form contents
- The user form (e.g. view/UI) is rendered by a section rule

Flow Action: Show Payment Info [Available]
CL: WIND-Auto-Work-EvaluateAndSellAVehicle ID: ShowPaymentInfo R:

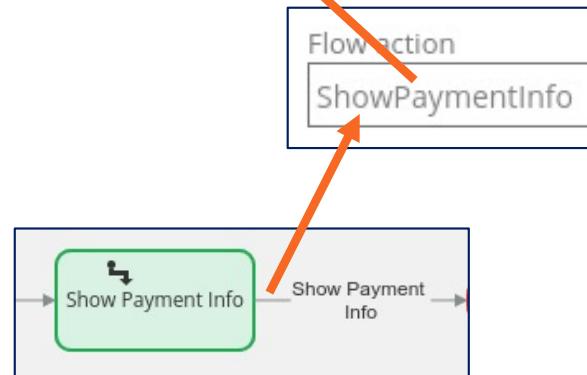
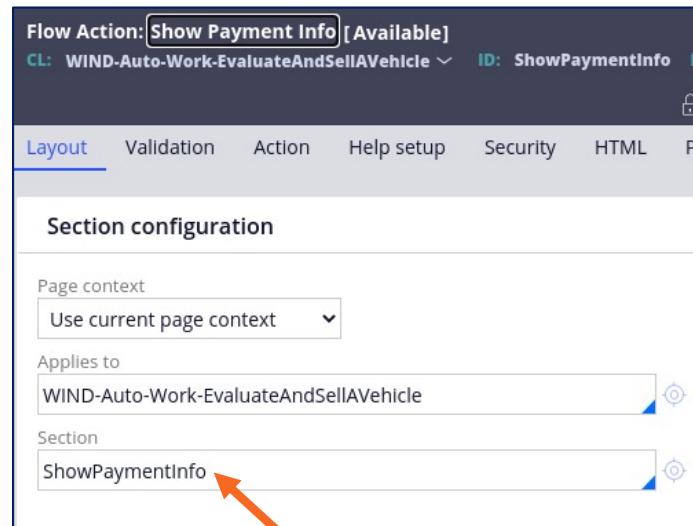
Layout Validation Action Help setup Security HTML Page

Section configuration

Page context: Use current page context

Applies to: WIND-Auto-Work-EvaluateAndSellAVehicle

Section: ShowPaymentInfo



Section Rules

Description

- Section - renders the view for displayed
- A Section rule is composed of layouts
 - Layouts arrange the controls
 - Pre-defined layout templates are available
 - Possible to define your own templates
- Section editor allows UI design without templates

This section is configured using the View editor. To access full design capabilities [convert to full section editor](#)

Design Settings Parameters Pages & Classes Specifications History

627px

Template

A

1 Column

Change

Account Number Text Input

Bank Name Text Input

Branch Text Input

Dropdown Currency

Bank Details Account Number
lorem ipsum

Bank Details Bank Name
lorem ipsum

Bank Details Branch
lorem ipsum

Account Type
Select...

Bank Details Amount
123456

Select a template

1 Column

1 Column with 100% width content

2 Column

2 Column (30 | 70)

2 Column (70 | 30)

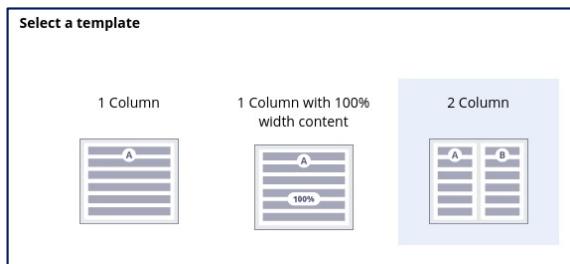
2 Column (Inline Wrapping)

3 Column

3 Column (Inline Wrapping)

Changing a Template and Section

- Click Change and choose the template to be used
- Drag the controls to the column
- Save the changes



- For specialized configuration of a Section click the convert to full section editor
- Commonly used layouts include
 - Dynamic
 - Table
- Select **Structural > Layout**

To access full design capabilities [convert to full section editor](#)

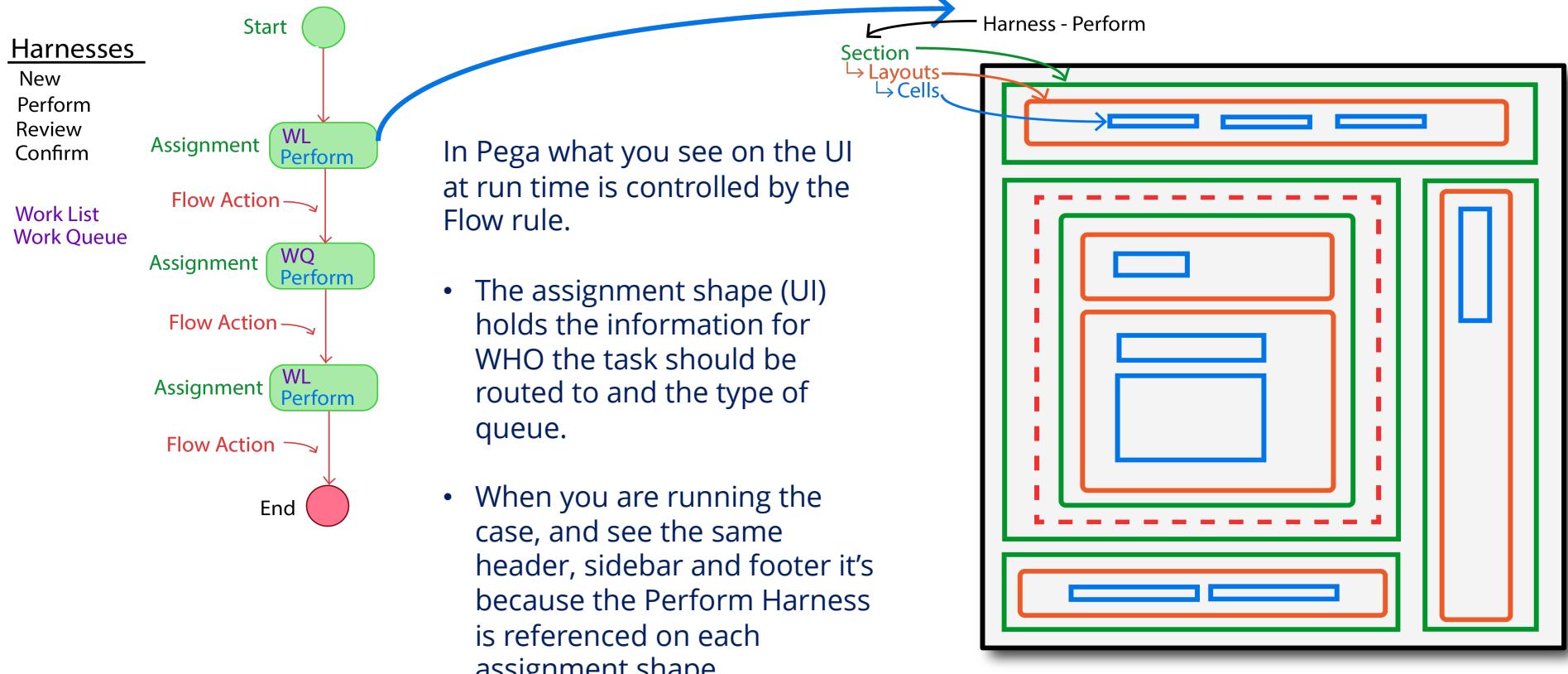
The interface shows a form with fields: "Bank Details Account Number" (Text Input), "Bank Details Bank Name" (Text Input), "Bank Details Branch" (Text Input), "Account Type" (Dropdown), and "Bank Details Amount" (Text Input). There are two sections labeled A and B. Section A contains "Account Number" (Text Input) and "Bank Name" (Text Input). Section B contains "Branch" (Text Input), "Account Type" (Dropdown), "Amount" (Currency), and a "Button". A "Change" button is located next to the template preview. The template preview shows a 2-column layout with columns A and B.

The interface shows a "Dynamic Layout (Stacked) - 1" section. It contains fields: "Account Number" (Text Input), "Bank Name" (Text Input), "Branch" (Text Input), "Account Type" (Dropdown), "Current" (Text Input), and "Amount" (Text Input). Below these fields is a "Button". The top navigation bar includes "Design", "Settings", "Parameters", "Pages & Classes", "HTML", "Specifications", and "History". The toolbar includes icons for copy, paste, delete, and layout options. A message at the top says "To access full design capabilities [convert to full section editor](#)".



Hierarchy of UI rules (1 of 2)

Description





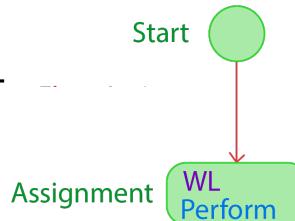
Hierarchy of UI rules (2 of 2)

Description

Harnesses

New
Perform
Review
Confirm

Work List
Work Queue



Configure View

Validate Rule

Flow Action
↳ Section
↳ Layouts
↳ Cells

pyCaseActionArea

Harness - Perform

Section
↳ Layouts
↳ Cells

Controls

Controls

Section
↳ Layouts
↳ Cells

Button
↳ Action Set

Submit

pxTextInput

pxDropDown

Property

Text
Decimal
Integer

Edit Validate
Pattern

###-##-####

XXXX@XX.XX

Overview

- Controls are configured in user interface restricting the end users to give input only in an expected format.
- Controls have properties to be configured as per the business requirement.
- Controls can be customized for input under editable settings.
- Controls can be configured with customized styles as per the brand requirement.
- Also known as UI elements configured in a section rule by using dynamic layouts.



Common Controls

Description

Four types of Controls

- **Data capture** - text input, text area, date only
- **Pickers** - dropdown, checkbox
- **Action** - buttons, links
- **Data display** – charts, **video**, image

▼ Data capture

-  Text input
-  Text area
-  Date time
-  Currency

▼ Pickers

-  Autocomplete
-  Multi-select list
-  Checkbox
-  Dropdown

▼ Action

-  Button
-  Link
-  Menu
-  Attach content

▼ Data display

-  Embedded section
-  Text
-  Label
-  Paragraph



Property types - Default controls

Description

- There are many predefined property types associated with default controls.
 - As soon as you create a property with a specific type, the associated default control will be used for data input at run time automatically.

Edit case type: UI_Controls

Actions Save and run Save

Data model Workflow Views Settings Test cases

Show: System fields Reusable fields ?

Property type Fields Validations

Name	ID	Type	Options
First Name	FirstName	Text (single line)	 
Last Name	LastName	Text	"pxRadioButtons/pxDropdown" control  
Gender	Gender	Picklist	Radio buttons  
Date of Birth	DateOfBirth	Date only	"pxDateTime" control  
Contact Number	ContactNumber	Phone	"pxPhone" control  
Email address	EmailAddress	Email	"pxEmail" control  
Communication Address	CommunicationAddress	Text (paragraph)	Plain text  
Father Name	FatherName	Text (single line)	 
Mother Name	MotherName	Text (single line)	 
Required Course for Admission	RequiredCourseForAdmission	Picklist	Drop-down list  

 Add field

         PEGA

Control Type and features

Description

Use case	Control type	Control features
Users must enter a date that includes day, month, and year.	Calendar	Selecting a date from a calendar icon helps ensure that users enter a date in a valid format.
Users must select one of three possible loan types. Users must see all types on the form.	Radio buttons	Restrict choices to a set of valid values and allows users to select only one value. You can use radio buttons when only a small number of options (for example, fewer than five) is available.
Users must select one of 10 types of office chairs from a list. The options do not need to be displayed on the form.	Dropdown	Restricts valid values to the ones that appear in the list. A drop-down list presents the options only when users click the control. This helps reduce the clutter on the form. Unlike radio buttons, you can configure the drop-down control so that users can select multiple values.
Users must select the country in which they reside from a list. Users can enter text in the control to help find the right country.	Autocomplete	When users enter one or more values in the control, the control filters the available options. This helps users find an option in a list if there is a large number (for example, more than 20) of possible options.
Users select an option to purchase extra travel insurance.	Checkbox	User can select the check box or leave it blank. This ensures that a true/false property is either true (selected) or false (unselected).

Control Type and features

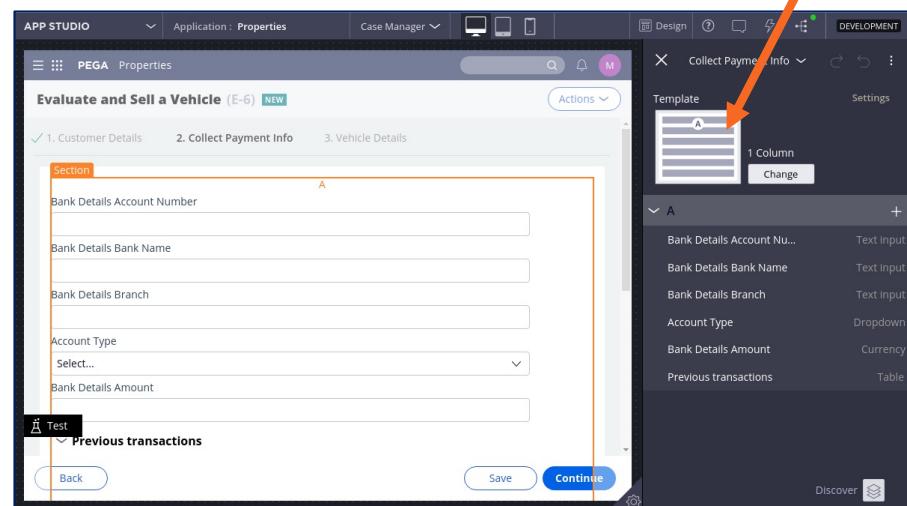
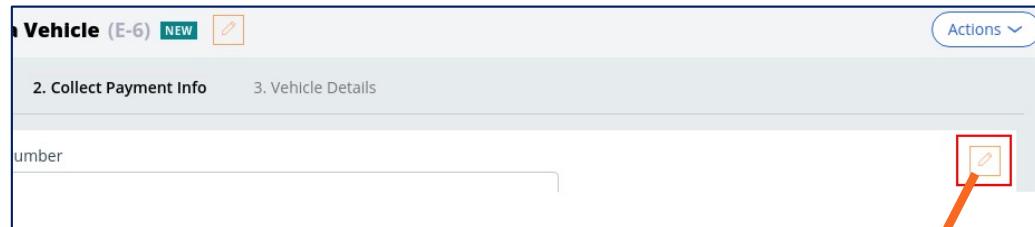
Description

Use case	Control type	Control features
To capture single line of text for any requirement like first name, last name, user name.	Textbox	Allows all types of characters as input in a single line.
To capture multiple lines of text from end users at run time like address, comments, description.	Text Area	Allows all types of characters as input in multiple lines.
To provide any website links, page links, social media links.	Link	Any clickable link can be configured.
Transport related businesses requires location tracking by using maps	Map	Allows to show a location on a map with address details
Online shopping applications requires to take signature of the customer at the time of shipment delivery.	Signature capture	Allows to capture the digital signature from the end users.

Advanced Design

Description

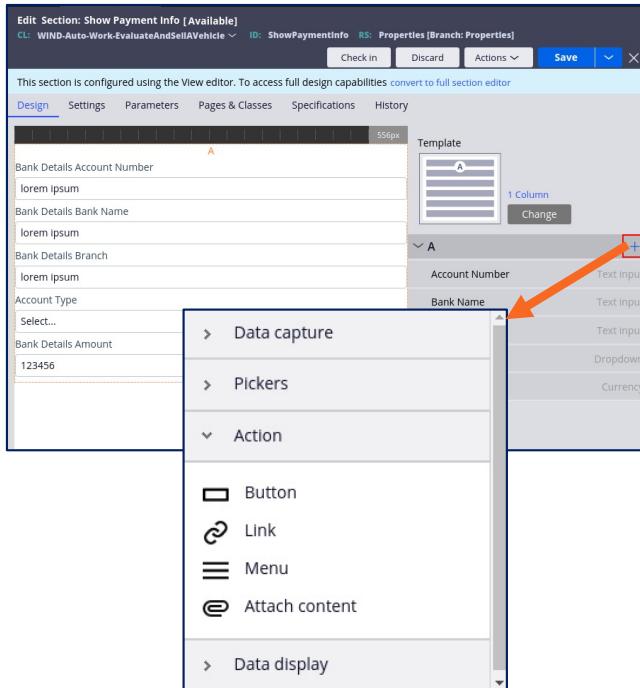
- For more advanced design and development of the UI use the Edit form feature from App Studio
- From the run-time view click the pencil icon to access the template view of the section
- Re-order fields by dragging them
- Click + to add new fields



Add a Control

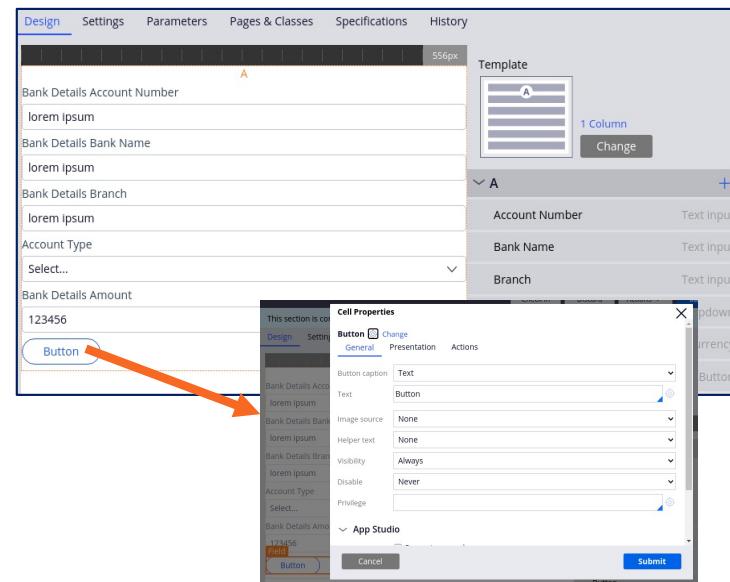
Implementation

- A section must contain one or more layouts
- Open the section
- Click + > **Category**> Control



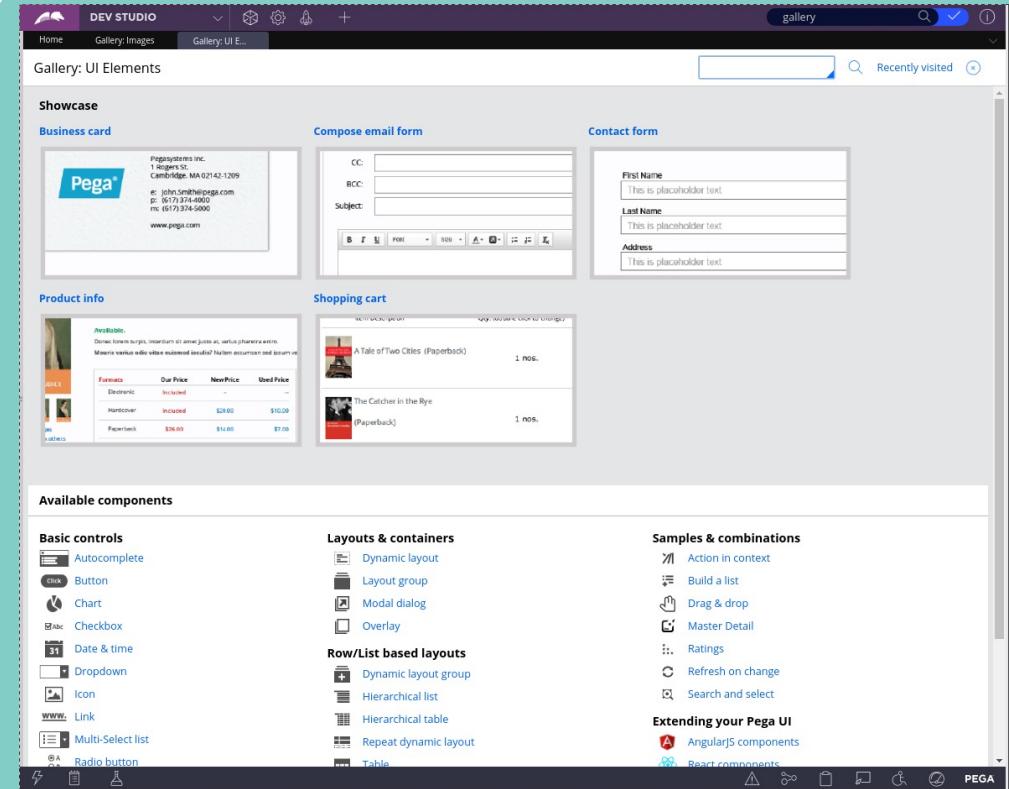
Configure a Control

- Select the control and configure its properties
- Submit and save the changes



Overview

- The UI Gallery provides sample, guardrail-compliant design patterns that follow best practices for delivering valuable user experience.
- You can use these components as a reference when you create your own user interface or copy them directly to your application for immediate reuse.
- For example, if you change the data source for the shopping cart design, you can use it in your application without further changes.

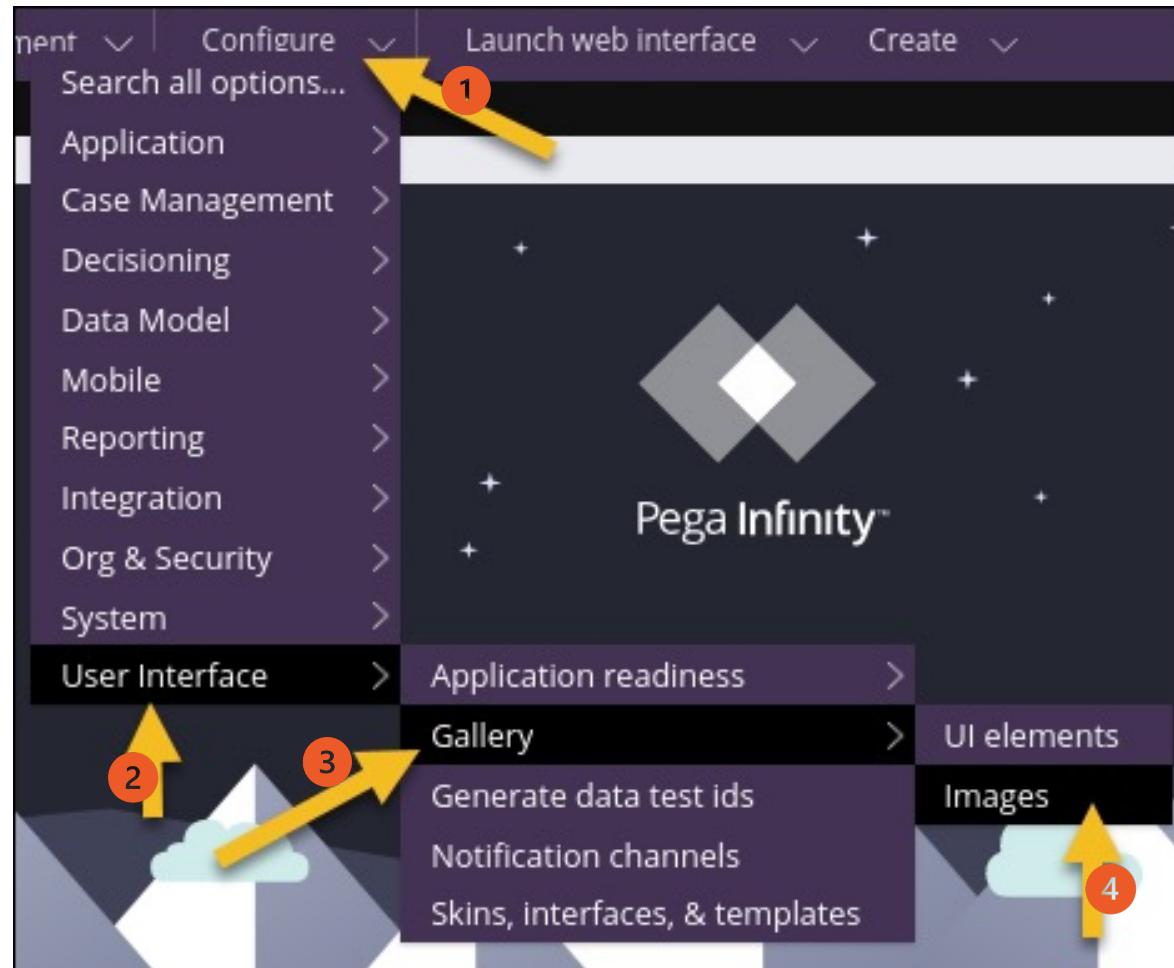




UI Gallery

Definition

- The UI Gallery provides an extensive selection of examples that demonstrate Pega user interface (UI) features and UI design best practices.
- Review these examples to learn PegaUI configuration techniques, and even incorporate them into your own applications.
- You can also add your own samples to the UI Gallery.





UI Gallery

Navigation

Dev Studio → Configure → User Interface →
Gallery → Images → UI Elements

Images

The screenshot shows the 'Gallery: Images' section of the Pega Dev Studio. At the top, there's a search bar with the word 'gallery'. Below it is a table with several columns of icons representing different file types like documents, folders, and images. Each icon has a corresponding file name below it. There are also some larger preview images. At the bottom, there's a navigation bar with icons for file operations like 'New', 'Open', 'Save', etc., and a 'PEGA' logo.

Elements

The screenshot shows the 'Gallery: UI Elements' section of the Pega Dev Studio. It features a 'Showcase' area with examples like 'Business card', 'Compose email form', 'Contact form', 'Product info', and 'Shopping cart'. Below this, there's a 'Available components' section with categories like 'Basic controls', 'Layouts & containers', and 'Samples & combinations'. Each category lists various UI components with small icons next to them. A sidebar on the right lists 'Samples & combinations' such as 'Action in context', 'Build a list', 'Drag & drop', etc.

Video Demo – UI Gallery

The screenshot shows the Pega Dev Studio interface with the title bar "DEV STUDIO" and "Application: GoGoRoad - Case Management". The main area displays a dashboard with a night sky background featuring stars, a crescent moon, and a diamond-shaped logo labeled "Pega Infinity". A banner at the top left says "Hide this until the next release" and one on the right says "Read more on the Pega Community". Below the banner is a section titled "Guardrail warnings (last 7 days)" with a table showing counts for Severe, Moderate, and Informational warnings across two categories: "Introduced by you" and "introduced by team". The "Informational" row for "Introduced by you" shows a count of 12. The "Informational" row for "introduced by team" also shows a count of 12. To the right is a "Security status" section showing "0 out of 33" security tasks completed. A sidebar on the right lists "Pega Community News" with links to "Visualize application layers for case processes and sections (8.2)", "Manage and communicate more easily in IVA for Email (8.2)", and "Meet deadlines more efficiently with Pulse tasks (8.2)". The bottom navigation bar includes links for Agile Workbench, Current work, Scenario Testing, Issues, Tracer, Clipboard, Live UI, Accessibility, Performance, and a PEGA logo.

	Severe	Moderate	Informational
Introduced by you	0	0	12
introduced by team	0	0	12

Guardrail warnings (last 7 days)

View all warnings Refresh

Security status

Refresh

0 out of 33

security tasks completed Review the Application Security Checklist at: Resources > Application Guides

Pega Community News Refresh

Visualize application layers for case processes and sections (8.2)

In App Studio, you can now easily visualize the application layers for case processes and sections, including work areas and views. When you click the Discover gadget for a case process or section, you are shown a visual representation of the application layers. You can open a layer in Dev Studio to see the rulesets and other attributes of an application. The following image shows the Discover gadget.

Manage and communicate more easily in IVA for Email (8.2)

To resolve email triage cases more quickly and efficiently, communication in Pega Intelligent Virtual Assistant™ (IVA) for Email is enhanced. You can reply to any email address, instead of only to the email address of an IVA triage case operator. Furthermore, response options are expanded with ability to forward, reply all, carbon copy (cc), and blind carbon-copy (bcc). You can send Pulse messages to discuss case details without exposing the information to external users. You are automatically notified in the Email Manager and Case Manager portals if you receive Pulse messages.

Meet deadlines more efficiently with Pulse tasks (8.2)

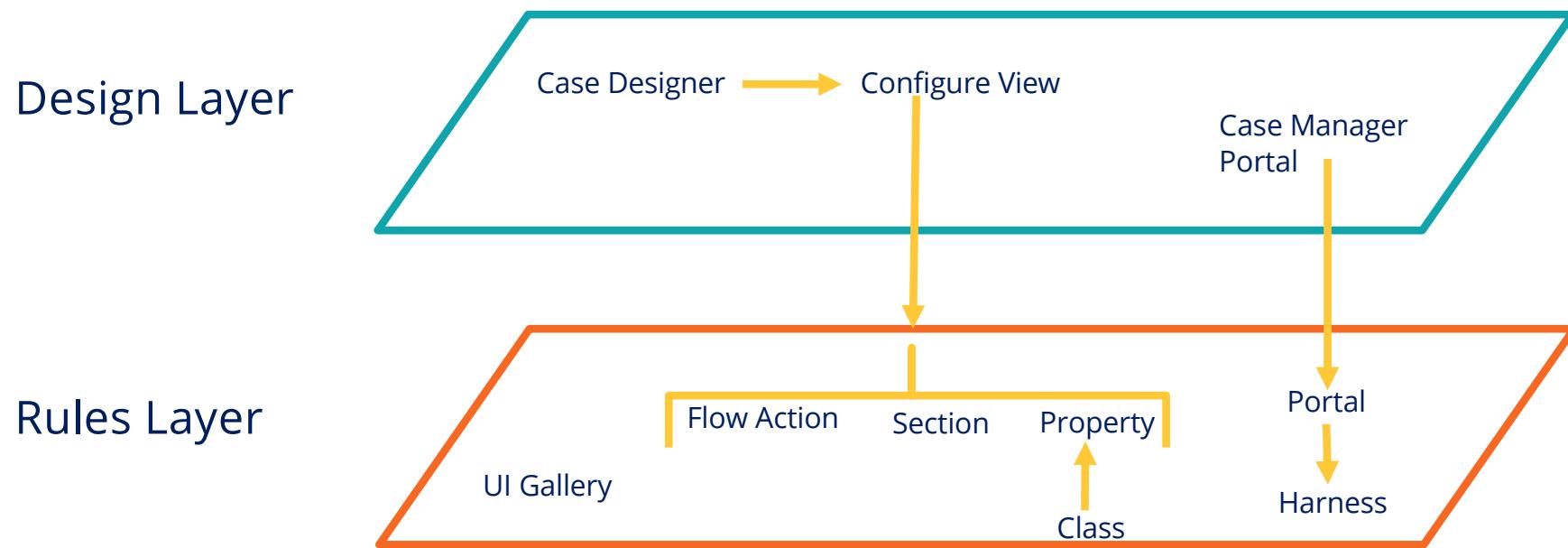
Organize your assignments and work to case deadlines more effectively by defining tasks in Pulse. To help complete case work within a specific time frame, create tasks and assign them to other users or to yourself. For example, you can assign a task to yourself as a reminder to upload address documents to a Car Loans case by the end of the day.

Agile Workbench Current work Scenario Testing Issues Tracer Clipboard Live UI Accessibility Performance PEGA 43

Pega Dev Studio - Google C...



Course Schematic



Java HTML CSS XML Javascript JSP

1 1 0 0 1 0 1 0 1 0 1 1 1 0 0 1

Skill Mastery

- Cosmos and User Interface
- Simple field types from out-of-the-box
- Fancy field types from out-of-the-box
- Design templates from section rule
- Controls mapped to default field types
- Understand the UI Hierarchy
- UI Kit
- UI Gallery



Best Practice

Points to consider while configuring user views:

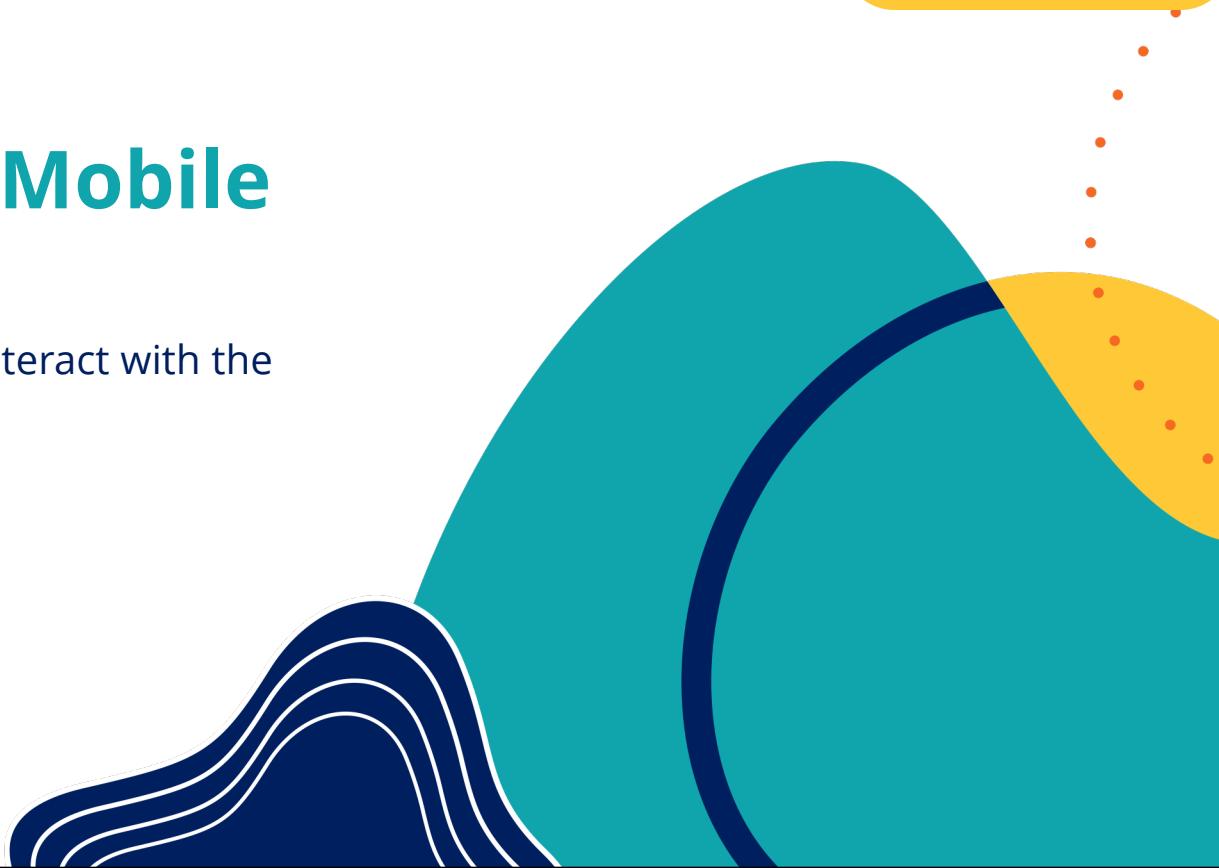
- Provide users with information when they need it
- Align the user view with the case life cycle to ensure the user interface is model-driven
- A model-driven approach results in faster application development, and a contextually sensitive UI
- A model-driven approach enables you to quickly modify the UI when business rules change
- Understand the user's role and what the user needs to accomplish
- Use common UI labels and elements
- Minimize data elements to an optimal set that is critical to the task
- Focus on what users are trying to accomplish
- Be aware of where users are in the life cycle and the next best action



SKILL
LESSON

Responsive UI and Mobile Experience

The view presented to the end user to interact with the application from a mobile device

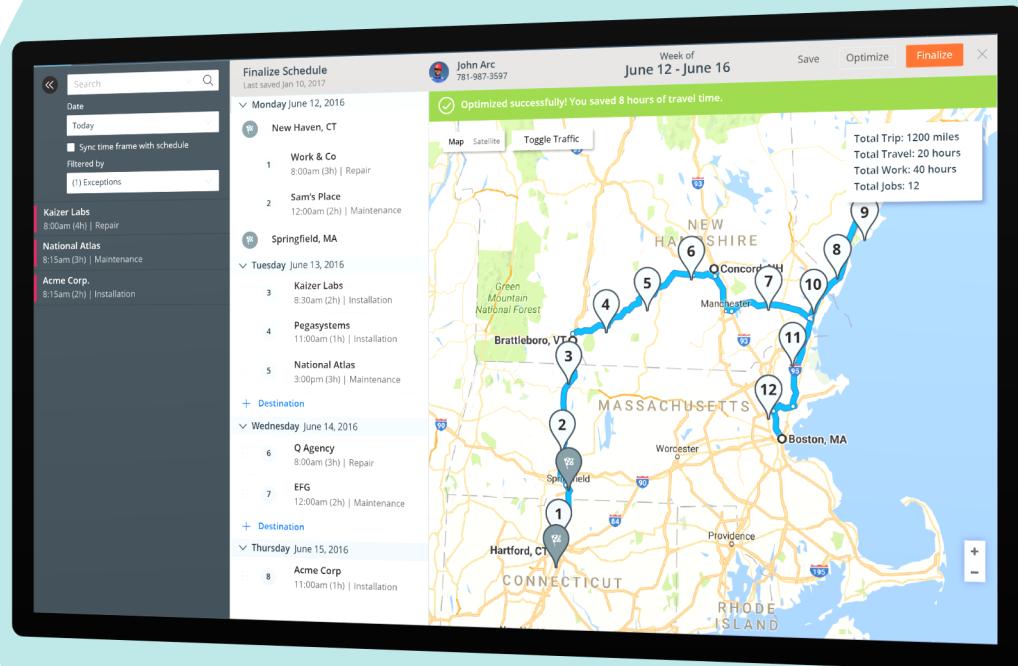


Overview

Leverage responsive behavior, such as that found in layout templates, to create an optimal user experience regardless of which device users prefer. Mobile app experiences built with Pega Platform™ can be configured for each user role or persona.

This module covers:

- Designing UI for multiple device types
- Designing a mobile app experience

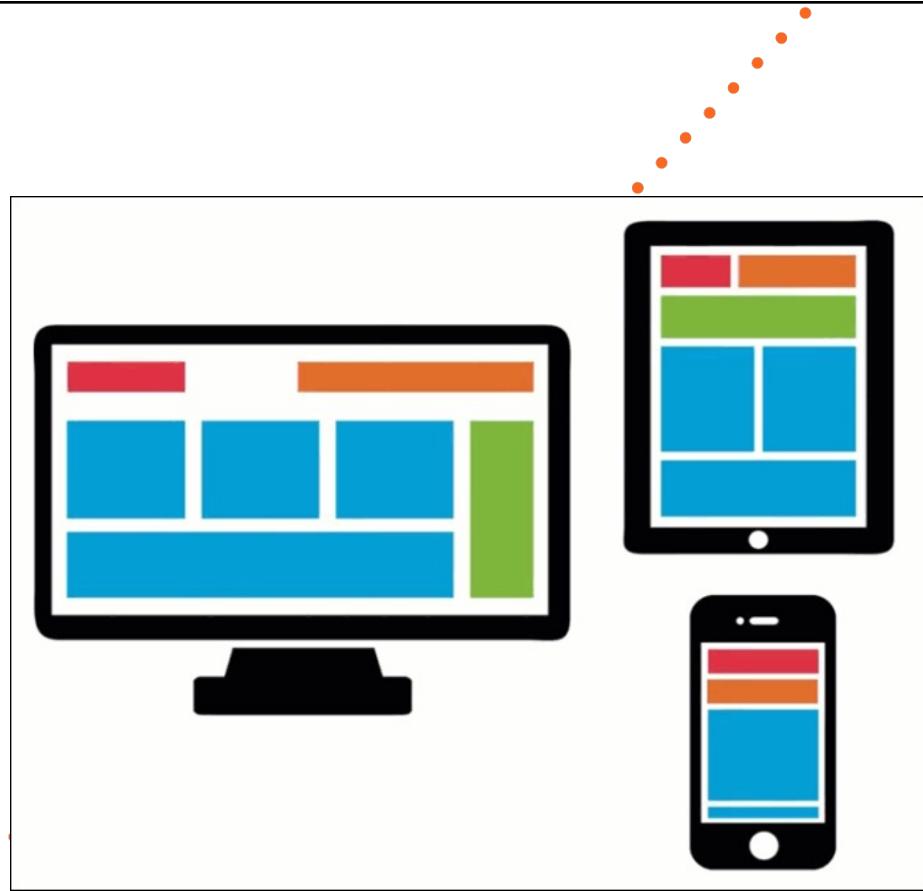


Responsive Behavior

Definition

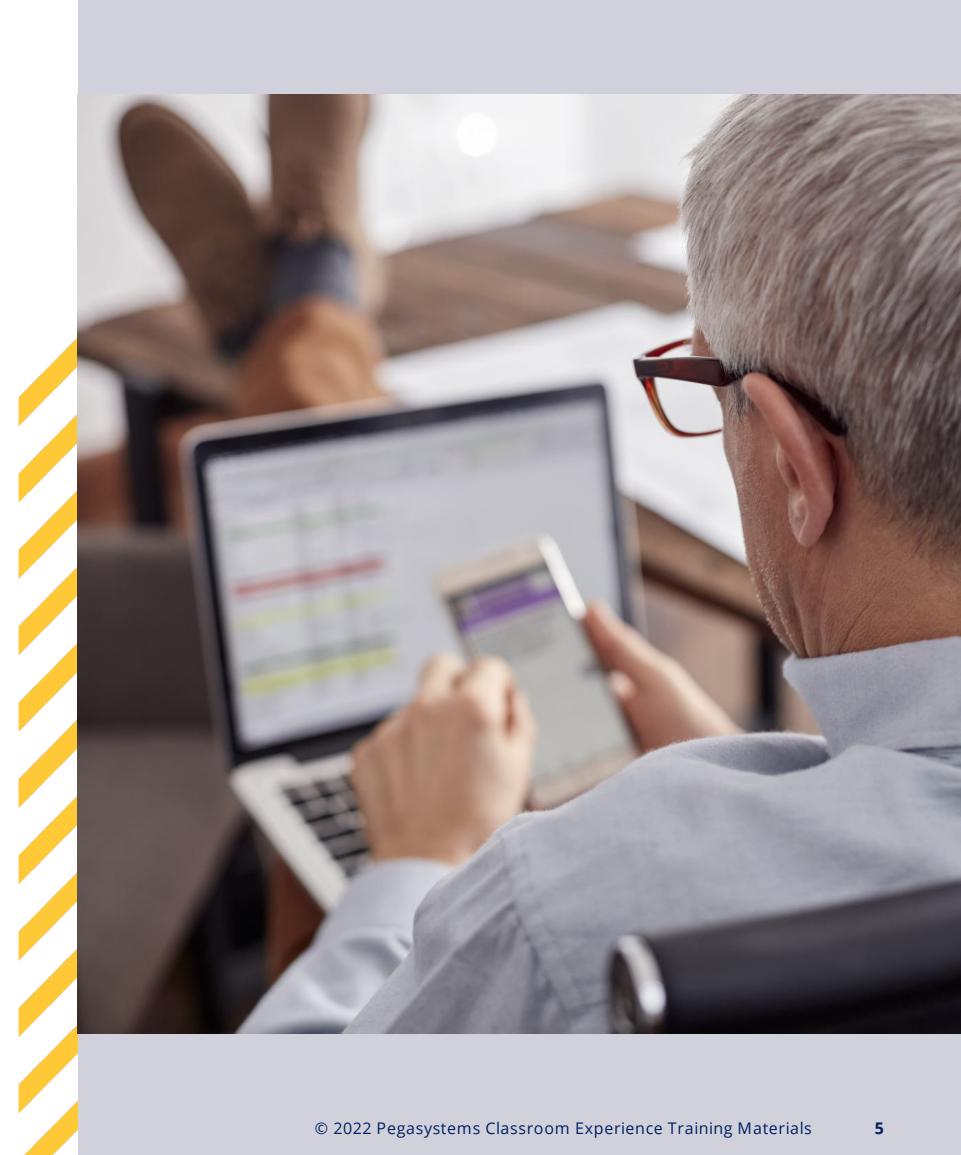
Responsive behavior provides an optimal user experience of the elements in a view regardless of screen size — minimizing horizontal scrolling and maximizing data presentation in the available display space.

Responsive behavior options include the design layout of the form and the column importance in tables.



Use Case

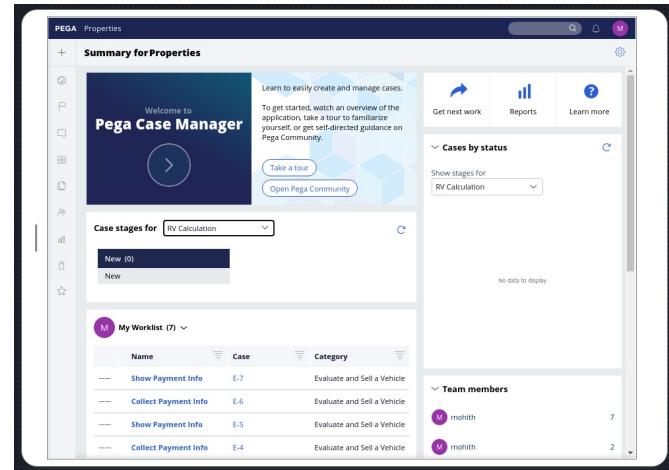
- Support for responsive behaviour is a best practice.
- Used whenever applications are expected to be run from different types of devices like a desktop, laptop, tablet or, smart phone each with varied screen sizes



Responsive breakpoints

Definition

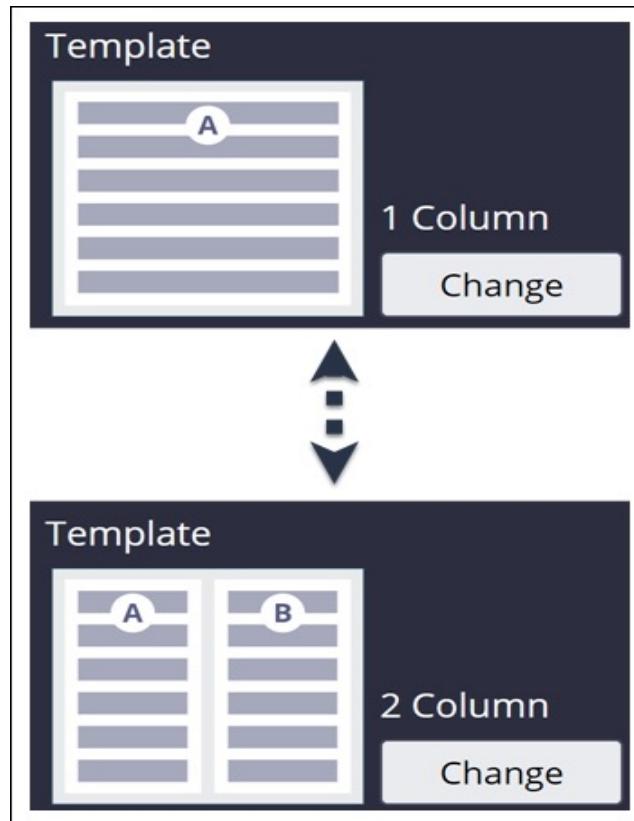
- A responsive breakpoint defines a threshold at which the UI reorganizes its content.
- Pega responsive breakpoints are set automatically by default.
- When the display width crosses a breakpoint, the responsive behavior is applied to the layout.
- With a dynamic layout, responsive breakpoint behavior changes the layout of the fields based on the width of the display area.
- For example, the default breakpoints automatically display four columns as two columns on a tablet, and as a single column on a mobile phone — making the form readable on smaller screen widths and eliminating the need for horizontal scrolling.



Layout Templates

Description

- The layout of the fields is also affected by design templates.
- The design template determines how the user interface changes.
- For example, you can configure a view to use a 2 Column design template then as the window size changes, the content will change accordingly.



Responsive UI and tabular data

Description

In Pega Platform™, a table automatically displays data from a field group list. You can configure responsiveness in tables to limit the information that is displayed when the display size changes.

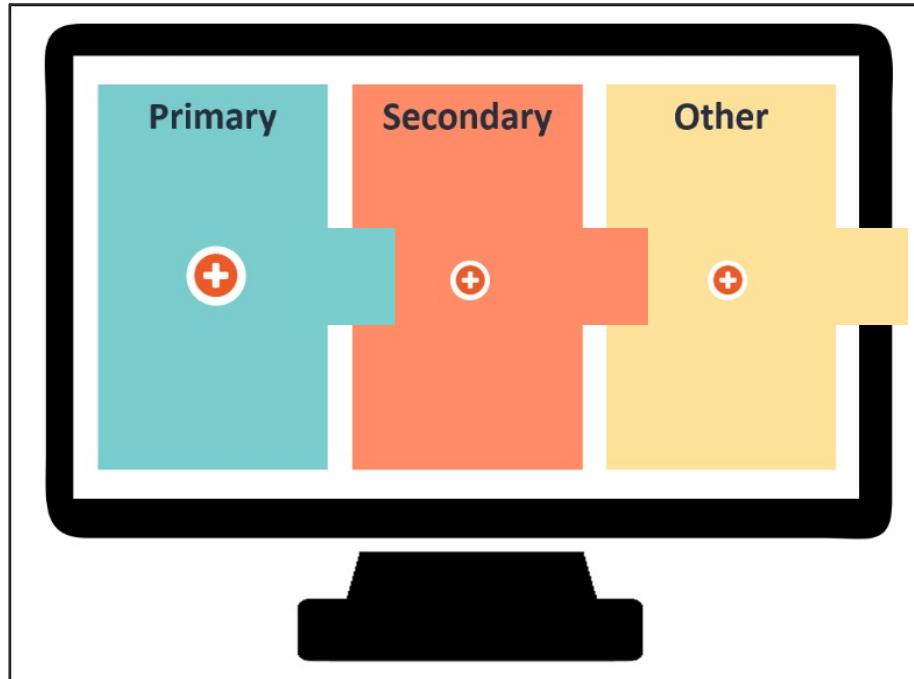
- Responsive table design minimizes horizontal scrolling by removing the less important information from the user interface.
- When designing responsive behavior, it is a best practice that the displayed information aligns with the business needs.
- The order in which information is presented to end users affects the user experience. For example, in the following table, the Part number column and Name column should always be displayed, and the Line total column should always be presented at the end of the displayed information.

Column name:	Part number	Unit cost	Name	Description	Quantity	Line total
	333-88	1.99	InkPen	2-pack ballpoint black	100	199.00

Column importance

Description

- A table has one or more columns, each with a configured importance setting.
- Column importance defines how Pega Platform presents columns as the display size changes.
- The column importance options are **Primary**, **Secondary**, and **Other**; by default, the left-most column is set to Primary.
- You configure column importance in **Dev Studio**.
- In **App Studio** you can set or change a column to primary importance while editing the form configuration at runtime.
- You can set a column's importance to primary by moving that column to the leftmost place in the table.
- When you change the order of columns in a table, Pega Platform automatically sets the leftmost column to primary importance and the remaining columns are set to secondary importance.

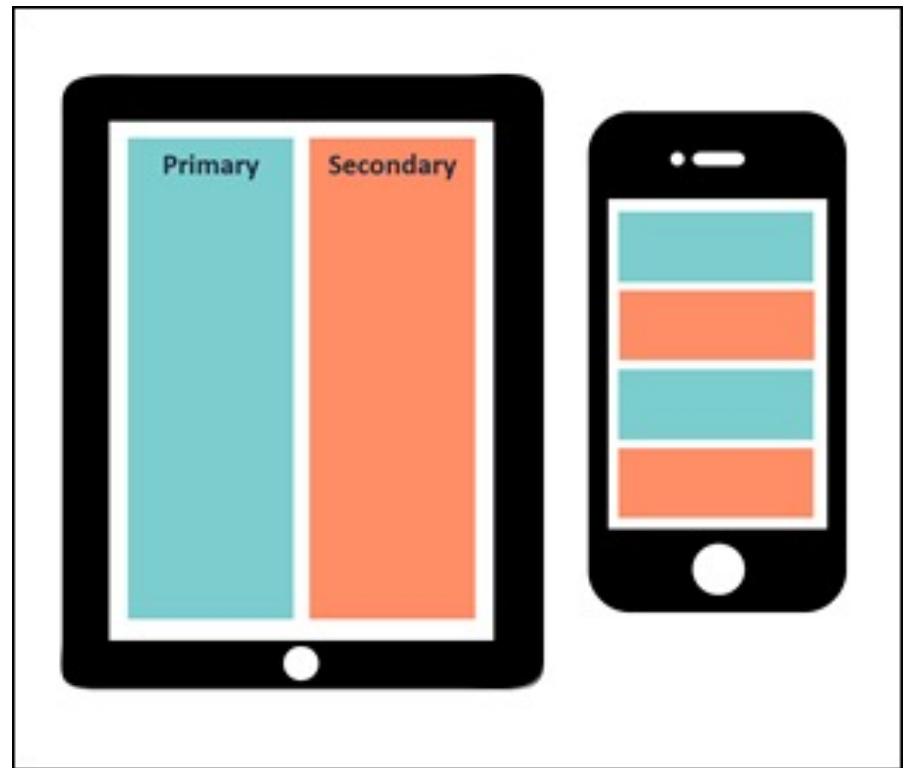


Responsive table default breakpoint settings

Description

Two responsive breakpoints are set for tables.

- At the first, wider breakpoint, columns with importance set to Other are not displayed.
- At the second, narrower breakpoint, the table is displayed as a list.
- For each row in the table, the contents of each column stack vertically, eliminating the need for horizontal scrolling.



Responsive Breakpoints

Navigation

Responsive breakpoints are configured in the skin rule.

Open the Skin rule and select **Component styles > Inherited > Responsive breakpoints**

Breakpoint behaviors:

- Hide this component
- Transform to other format
- Transform to list
- Drop columns with importance

Other

The screenshot shows the 'Edit Skin' interface for 'Cust_Onboard'. The 'Component styles' tab is selected. Under 'DYNAMIC LAYOUTS', the 'Inherited' tab is chosen. A list of formats includes 'Default', 'Stacked', 'Inline', 'Inline grid double', 'Inline grid triple', 'Action area', and 'Grid filter panel'. The 'Responsive breakpoints' section is expanded, showing a checked checkbox for 'Enable support for responsive breakpoints'. Below it, 'Breakpoint1' is defined with a 'Remove' link. Two options are available: 'Transform to other format' (selected) and 'Hide this component'. A note states: 'At the dimensions provided below, this dynamic layout will use the following format:'. It includes dropdowns for 'Action area' (set to 'Screen') and 'Media type' (set to 'Screen'), and input fields for 'max-width' (768 px) and 'min-width' (empty). A note says: 'Leave min-width empty when a range is not desired'. At the bottom is a blue '+Add breakpoint' button. To the right of the main panel, there are sections for 'Label settings' (vertical alignment top, label position top, spacing 5px), 'Item spacing' (top margin 0px, right margin 0px, bottom margin 14px, left margin 0px), and another 'Responsive breakpoints' section with an unchecked checkbox for 'Enable support for responsive breakpoints'. Yellow arrows point from the text labels in the first paragraph to the corresponding UI elements: one arrow points to the '+Add breakpoint' button, and another points to the 'Responsive breakpoints' section in the sidebar.

Different Devices

Implementation

Same application on different devices:

The screenshot shows the Pega App Studio interface with the 'Customer On-boarding' application open. The main form is titled 'Customer On-boarding (C-9002)' and contains fields for 'Customer Registration' such as First Name, Last Name, Company Name, Official Email Id, Personal Email Id, Mobile Number, Correspondence Address, Preferred Mode of Contact (Email, Mobile, In Person), and Services Interested (Email, Credit Cards, Etc.). It also includes sections for 'Case details', 'Open assignments', 'Recent followers', 'Recent content', 'Participants', and 'Related cases'. A sidebar on the left shows the application's navigation structure.

The screenshot shows the same 'Customer On-boarding' application running on a tablet device. The interface is identical to the desktop version, displaying the customer registration form and various data sections.

The screenshot shows the same 'Customer On-boarding' application running on a mobile phone device. The interface is identical to the desktop and tablet versions, maintaining a consistent look and feel across all devices.

Configuration

Implementation

Configuring responsive behavior on a table:

The screenshot shows the 'Edit Section: Product Details [Available]' interface in Pega Studio. The section header includes CL: WIND-Cust_Onboard-Work-CustomerOnboarding, ID: ProductDetails, and RS: Cust_Onboard:01-01-01. The toolbar has Save and other actions. A message at the top says: 'This section does not follow section best practices of using a design template. Convert section to use a design template'. Below is a navigation bar with Design, Settings, Parameters, Pages & Classes, HTML, Specifications, and History.

The main area displays a 'Table - 1' component. The table has columns: Select, Product Id, Product Name, Category, Credit Cards, and Charges. The 'Category' column is highlighted with a red circle and a yellow arrow pointing to it from the right. Another yellow arrow points from the left towards the 'Importance' dropdown in a modal dialog.

A modal dialog titled 'Edit Section: Product Details' is open, showing 'Column Properties'. It includes fields for Width (101), Inline style (checkbox checked), Enable sorting (checkbox checked), and Importance (dropdown menu showing Primary selected). A yellow arrow points from the left towards the 'Importance' dropdown. The 'Submit' button is visible at the bottom of the dialog.

Mobile Channel

Navigation

App Studio > Channels

Definition

- Convenient access channel to Pega applications to support employees in the field.
- Use existing channel
- Create new channels

The screenshot shows the Pega App Studio interface for creating a new channel interface. The left sidebar has 'Channels' selected. The main area shows a grid of channel icons: Portal, Web mashup, Mobile (which is highlighted with a red box), Unified Messaging, Email, Web Chatbot, and Alexa. Below this is a section titled 'Current channel interfaces' with cards for 'User Portal' (Employee-facing portal), 'User Mobile App' (Pega Mobile Client), and 'uPlus Expense' (Pega Mobile Client). A red arrow points from the 'Mobile' icon in the grid to the 'uPlus Expense' card. A modal window titled 'New mobile interface' is open, showing fields for 'Name*' (uPlus Expense) and 'Description' (A mobile app for submitting and managing expense reports). A 'Submit' button is at the bottom right of the modal.

Mobile Channel

Content Configuration Layout Manage

Description

- Content
 - Controls the content available to users in the mobile app. Customize app navigation and functionality by:
 - Adding reordering, and removing the default mobile app pages, such as search functionality and notification lists
 - Creating customized app content using mobile list pages
 - Defining swipe actions for list items to provide users with quick access to common tasks
- Configuration
 - Controls basic app functionality, such as
 - The app name as displayed on the mobile device
 - The role assigned to users of the mobile app
 - Whether the app supports offline processing
 - App security, including user authentication and session management
- Layout
 - Controls app branding and styling options, including the launch page and UI element colors. You can change the default icon, using native icons or custom files for the icon to personalize your app.
- Manage
 - Controls administrative functions such as log access and administrative push notifications.
 - Note:** Administrative push notifications, which are sent from the console on this tab, differ from the app-based push notifications sent during case processing. For example, you use the administrative push notification console to inform users to upgrade to a newer version of the mobile app.

Mobile Channel Security

Implementation

- Define authentication method
- Enable lock and set timeouts
- Preview and test the behavior

The screenshot shows the Pega Configuration interface with the 'Configuration' tab selected. On the left, a sidebar lists 'Content', 'Configuration' (selected), 'Layout', and 'Manage'. Under 'Configuration', the 'Security' section is highlighted. The main panel displays the 'Authentication and Security' configuration. It includes options for 'Authentication method' (set to 'Pega Platform account'), 'Enable application lock' (checked), and 'Unlock with' (set to 'Biometrics and device lock'). Two timeout options are shown: 'Lock app after 30 minutes of inactivity' (checked) and 'Lock app after 2 minutes of session duration' (unchecked). A red box highlights the 'Enable application lock' and 'Unlock with' sections. To the right, a mobile phone screen displays a lock screen titled 'User Mobile App' with 'Enter PIN'. The screen shows a numeric keypad from 0 to 9, a 'Touch ID' button, and a 'Cancel' button. The Pega logo is visible in the top right corner of the phone screen.

Mobile app branding and theme

- Customize interface to match your branding requirements
- Customize icons
 - Upload image
 - Select from list
 - Default text-based icons
- Define application wide theme

Realtime preview!

The image displays two screenshots of a mobile application's branding and theme configuration interface. The top screenshot shows the 'Branding' section, which includes a 'Launch screen' and an 'App icon'. The bottom screenshot shows the 'Theme' section, which includes settings for 'Bottom menu style' (set to 'Icons and text'), 'Floating action button position' (set to 'Left/right aligned'), and 'Mobile specific colors' for various UI elements like headers, footers, and buttons. Both screenshots show a preview of the mobile application's interface on the right, featuring a green header, a green body with text, and a green footer with navigation links.

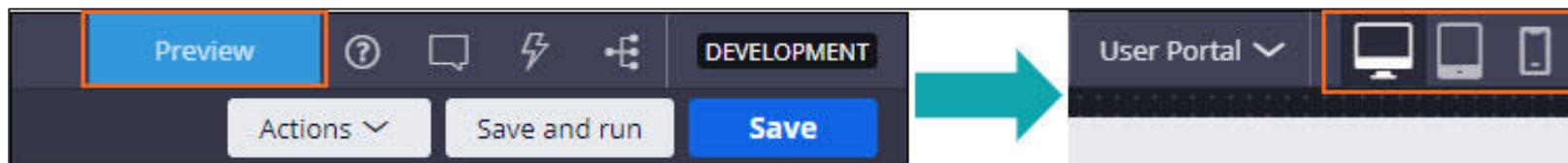
Skill Mastery

- Skin rule
- Responsive UI configuration
- Responsive table configuration
- Previewing application in large screen, Tablet, Phone

Unit Testing and Debugging Guidance

Preview for multiple devices

In App Studio, you can preview your application to see how it looks on various devices using the **Preview** feature.



Mobile design best practices (1 of 3)

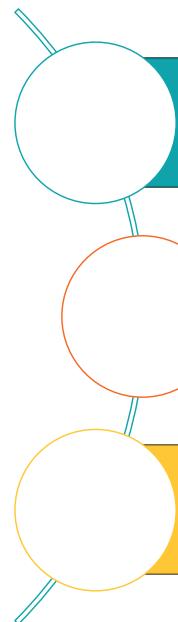
Use case

- Offline use
 - Device characteristics
- A mortgage underwriter frequently visits offices in the locations where the mortgage company does business. While traveling on a plane, the mortgage underwriter reviews loan requests. The mortgage app must be available when the mobile device is in airplane mode.
 - In a course selection desktop application, a hover action displays more information when a pointer hovers over a course name. The same hover action is useless on a mobile app because mobile devices do not support hover events.

Best practice is to design to be compatible across device types.

Mobile design best practices (2 of 3)

Offline



Enable client-side decisions and validations

Use data pages as data sources

Use layout groups to take advantage of responsive UI

When designing an app to support offline use, minimize server access in the UI design.

Mobile design best practices (3 of 3)

Device characteristics



Create a simple user interface



Use auto-generated controls



Incorporate native features



Design for finger taps



Test the application



Use relative positioning

Best practice is to design to be compatible across device types.

Supplementary information

Certification exam content

- **Authoring mobile list pages:**

<https://community.pega.com/knowledgebase/articles/mobile/86/authoring-mobile-list-pages>

- **Customizing image assets of mobile apps:**

<https://community.pega.com/knowledgebase/articles/mobile/86/customizing-image-assets-mobile-apps>

- **Previewing mobile apps:**

<https://community.pega.com/knowledgebase/articles/mobile/86/previewing-mobile-apps>



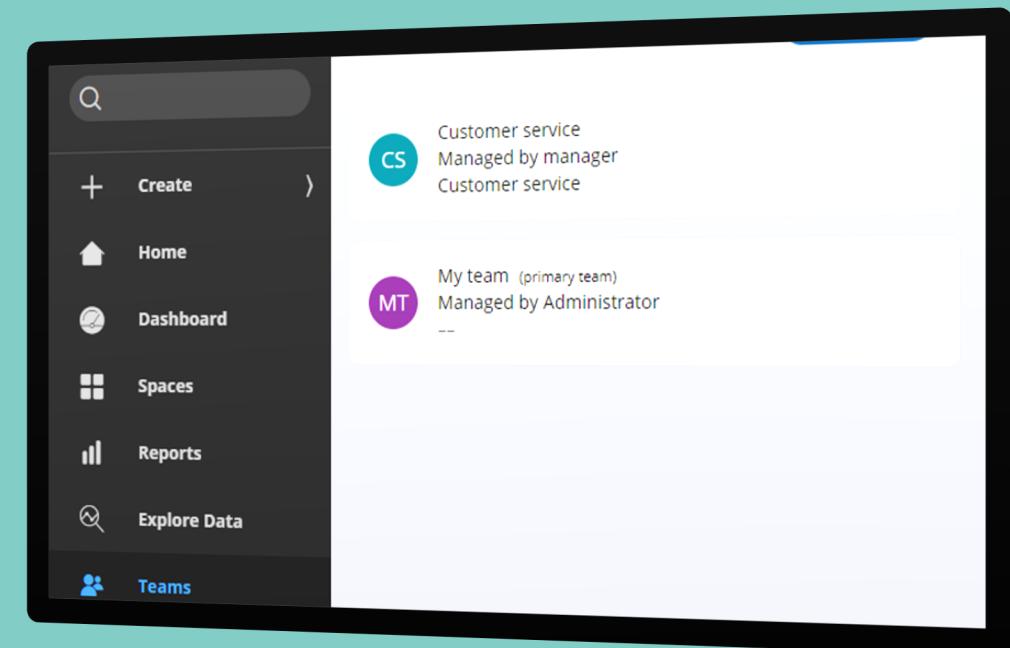
SKILL
LESSON

Portals and Dashboards



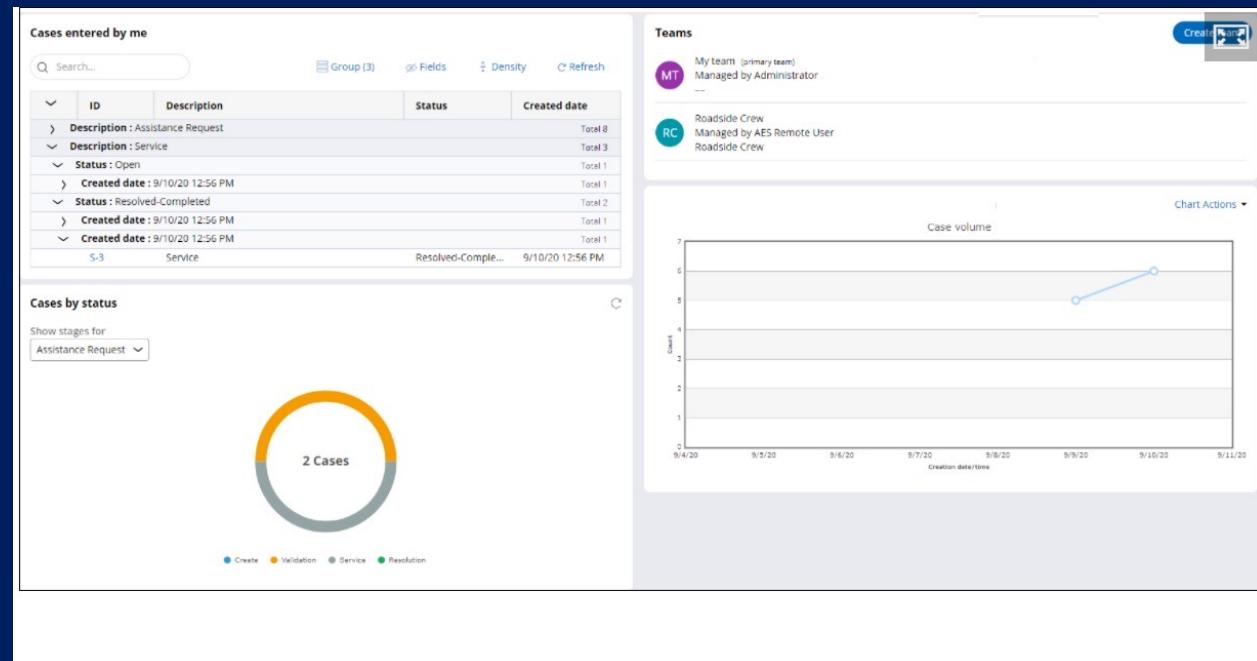
Overview

Users interact with Pega via the Portals.
Portals provide a digital customer experience.



Use Case

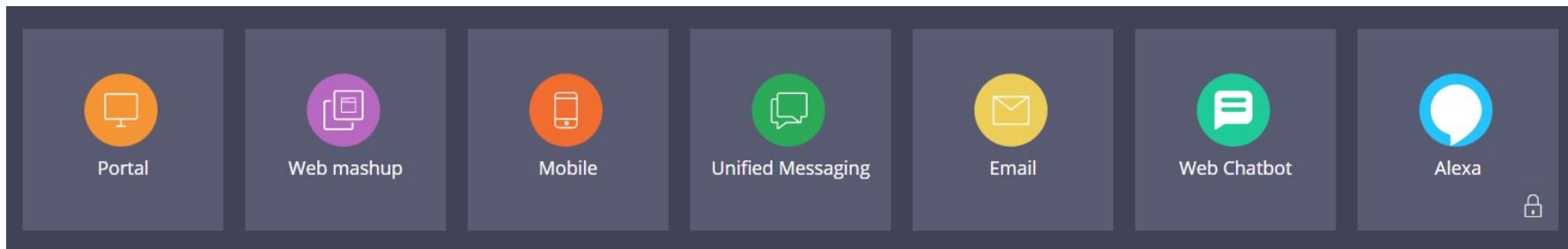
- End users interact with Pega via portals.
- Information can be organized in a dashboard.
- A loan agent reviews a new Loan Request that is assigned to her. Progress of all loans can be seen in the dashboard.



Channel

Definition

- A messaging service, voice service, web portal or mobile portal.
- Customers interact with organizations through a variety of channels.
- Created from templates that include predefined layouts and navigation for use in an application.
- Provide ways for users to interact with your application by using **Pega Intelligent Virtual Assistant™** and **Pega Email Bot**.
- The **Channels** landing page allows you to create, view, and edit all types of channel interfaces.
- The following Channels can be added:

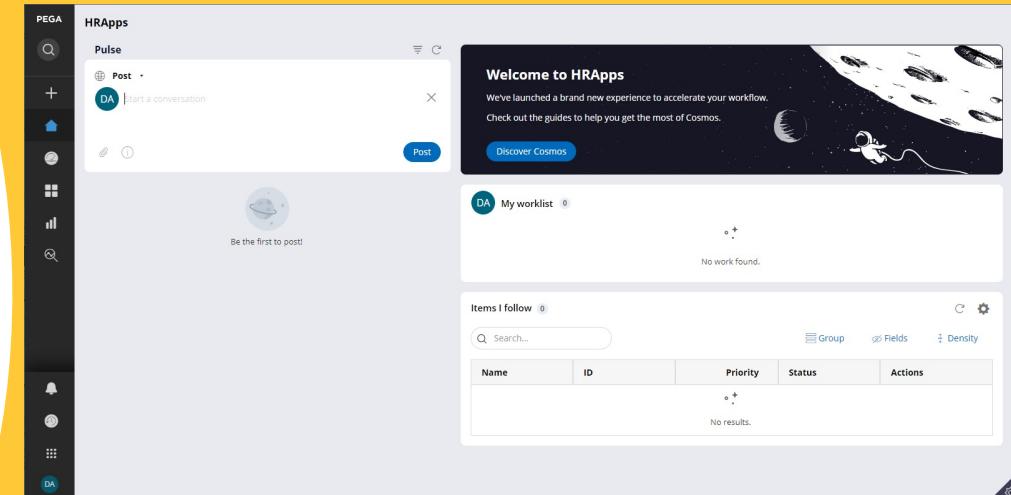


- New components can be downloaded from Pega Marketplace.
- The Lock icon on a channel indicates that the channel component is not yet added to the application.

Portals

Definition

- Users experience the portal through a browser, regardless of device type.
- Easily configured by choosing from predefined portal templates that define screen layout and required features.
- **User** portal provides a standard user interface for working on cases. It is intended to be used by end users on desktop and mobile devices.



Pages

Definition

- **Page Menu**

- Create, configure, and add pages to menus within an application to display specific information.
- Added to a portal and automatically added to the portal's navigation menu.
- Managing the list of pages, customizes the primary navigation menu, improving navigation and user experience.

- **Page Types**

- **Landing Pages** are created to build an application that matches the needs of your users and consists of fields, control and resources presented as images and text.
- **Custom pages** are Pega's out-of-the-box landing pages that are built as part of the application and are not customizable by end users.

The screenshot shows the PEGA GoGoRoad application. On the left, a dark sidebar displays the 'Page Menu' with options: Create, Home, Dashboard, Spaces, Reports, Explore Data, and Teams. The 'Teams' option is highlighted with a blue underline. To the right, a light-colored panel titled 'My Teams' lists two teams: 'Customer service' (Managed by manager, Customer service) and 'My team (primary team)' (Managed by Administrator). A 'Create team' button is located at the top right of the 'My Teams' section. The bottom right corner features a yellow decorative graphic with a grid of red dots.

Dashboard

Description

- Dashboards are customizable by end users and are made up of widgets that consolidate summary information.
- Displays operational information about an application and key performance indicators from different sources.

The screenshot shows a Pega dashboard interface with the following components:

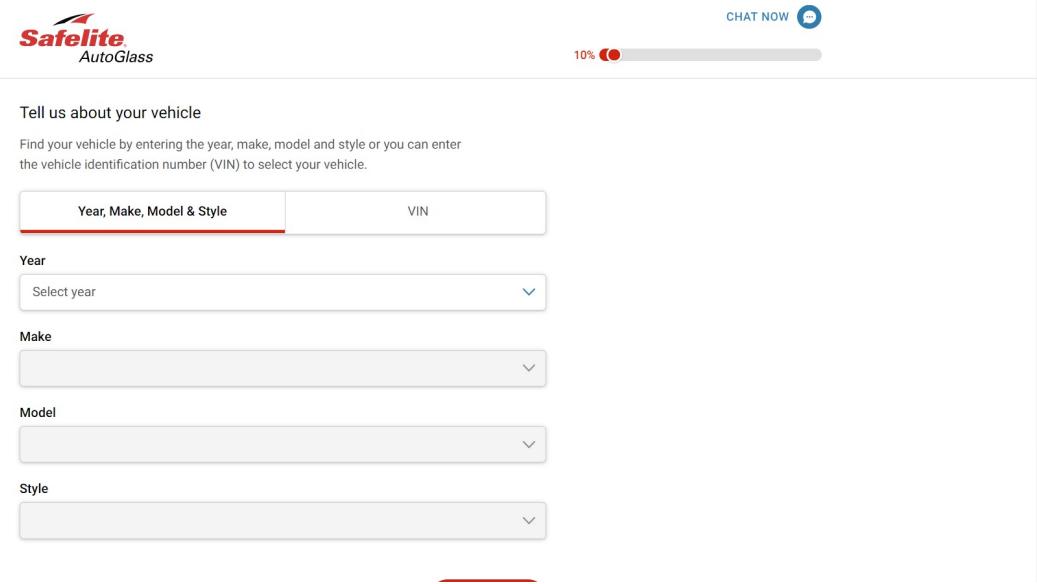
- Left Sidebar:** Includes links for '+ New', 'Dashboard', 'My Work', 'Pulse', 'Spaces', 'Documents', 'My Teams', 'Reports', 'Tags', 'Following', and 'Recents' (with entries 'Case C-05' and 'Case C-04').
- Main Content Area:** Titled 'Summary for Your Application'. It contains several widgets:
 - Case stages for case type:** A process flow diagram showing four stages: Stage 1 (Process), Stage 2 (Process), Stage 3 (Process), and Stage 4 (Process).
 - Operator:** Shows 5 OPEN cases and 2 OVERDUE cases. A table lists cases assigned to an operator:

Name	Case	Category
-2d Assignment name	C-05	Case name
-1d Assignment name	C-04	Case name
-- Assignment name	C-03	Case name
-- Assignment name	C-02	Case name
-- Assignment name	C-01	Case name
 - Cases by status:** A donut chart showing the distribution of cases by status: Stage 1 (dark blue), Stage 2 (orange), and Stage 3 (yellow). The legend indicates: Stage 1, Stage 2, Stage 3.
 - Team members:** A list of team members with icons:
 - O Operator
 - A Administrator
 - U User

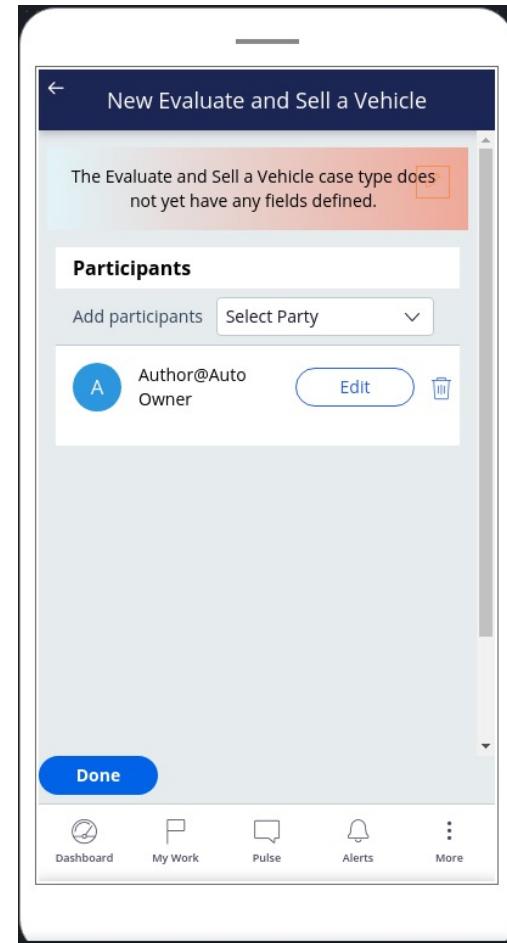
Portals

Navigation

Portals are accessed via a URL, web address or interface.



The screenshot shows a web-based portal for Safelite AutoGlass. At the top left is the Safelite AutoGlass logo. To its right is a "CHAT NOW" button with a progress bar indicating "10%". Below the logo is a section titled "Tell us about your vehicle". It contains two input fields: "Year, Make, Model & Style" and "VIN". Underneath these are dropdown menus for "Year", "Make", "Model", and "Style". The "Year" dropdown has "Select year" as its placeholder. The "VIN" field is currently empty. The entire form is set against a white background with a vertical scroll bar on the right side.



App Studio: Accessing Channels (1 of 2)

Navigation

Click on **Channels** to access the channels.

The screenshot shows the Pega App Studio interface with the following details:

- Header:** APP STUDIO, Application: Auto, Preview, DEVELOPMENT.
- Left Sidebar:** Overview, Case types, Data, **Channels** (highlighted with a red box), Users, Settings.
- Top Bar:** Create new channel Interface, with icons for Portal, Web mashup, Mobile, Digital Messaging, Email, Legacy Webchat, and Alexa.
- Current channel interfaces:**
 - User Portal:** Default employee-facing portal for Cosmos Rules applications. Use the Interfaces landing page to edit this portal and add/remove pages.
 - User Mobile App:** Pega Mobile Client.
 - API:** APIs are a set of REST services exposed by the application, including the Pega API - a set of built-in REST services for Pega applications.

App Studio: Accessing Channels (2 of 2)

Navigation

Click on a Channel Type to create a new Channel.

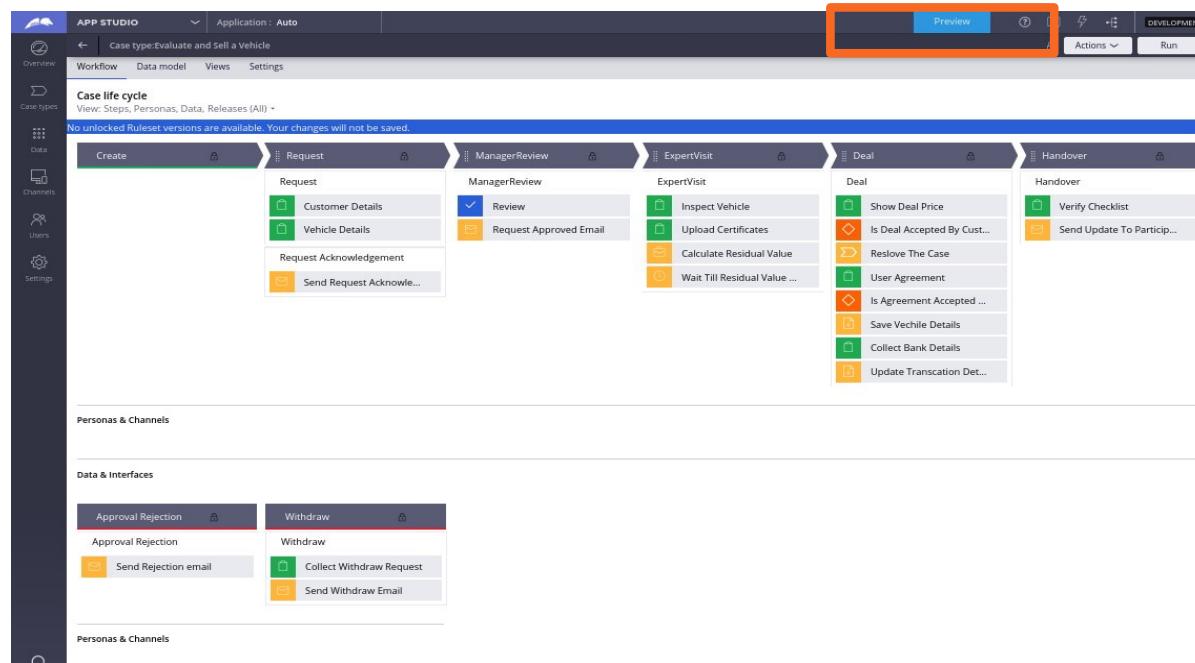
The screenshot shows the Pega App Studio interface. On the left, a vertical navigation bar includes links for Overview, Case types, Data, Channels (which is selected), Users, and Settings. The main area has a header "APP STUDIO" with "Application : Auto". Below the header, there's a "Create new channel interface" section with five icons: Portal (orange), Web mashup (purple, highlighted with an orange box), Mobile (red), Digital Messaging (green), and Email (yellow). To the right, a "Current channel interfaces" section lists "User Portal" (orange icon) and "User Mobile App" (red icon). A modal window titled "New Web mashup interface" is open, showing "Basic options" like Name and URL, and "Configuration" settings for Action, Case type, Thread name, and Skin. The URL field contains "http://localhost:9080/prweb/app/Auto_3465/". The configuration section also includes "Allow passing dynamic parameters" and "Initial skeleton" settings. A note at the bottom of the modal states: "Note: While using mashup, 'Enable same-site cookie attribute' option should be disabled and in case 'Enable CSRF token check' is enabled, 'Enable referrer check' should be enabled with valid referral list under the System->Settings->Cross-Site Request Forgery landing page."

App Studio: Building Portals & Pages (1 of 3)

Navigation

You build portals in App Studio by picking a template, populating it with pages and reusable widgets, and then setting up the primary menu.

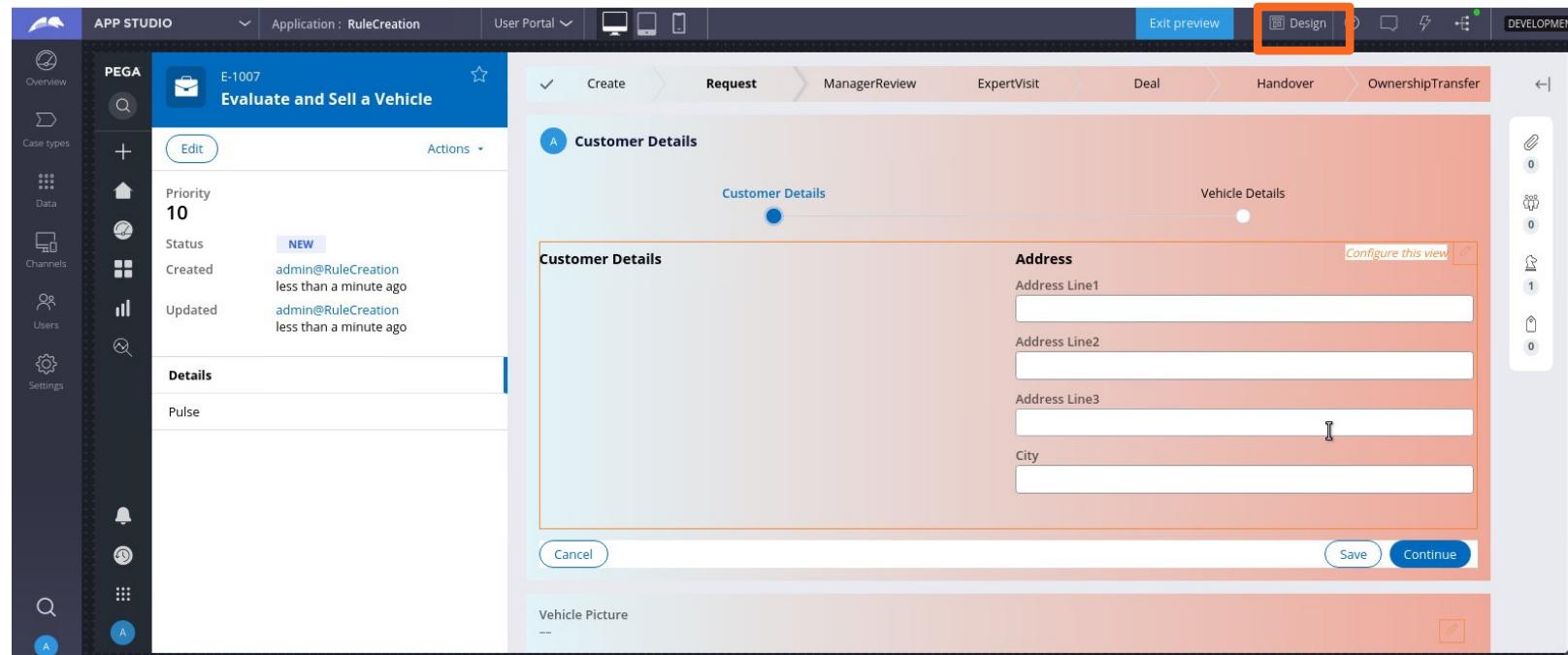
App Studio > click on **Preview**



App Studio: Building Portals & Pages (2 of 3)

Navigation

Click Design

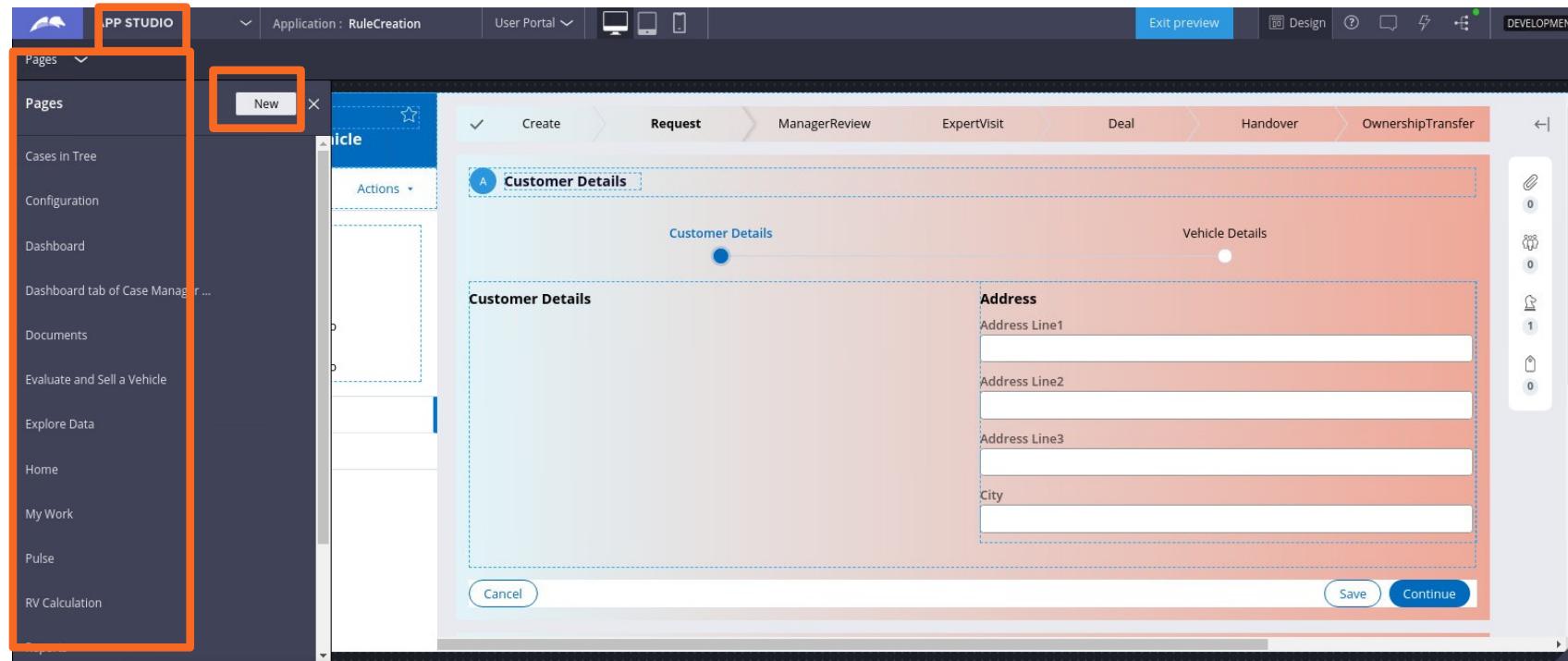


App Studio: Building Portals & Pages (3 of 3)

Navigation

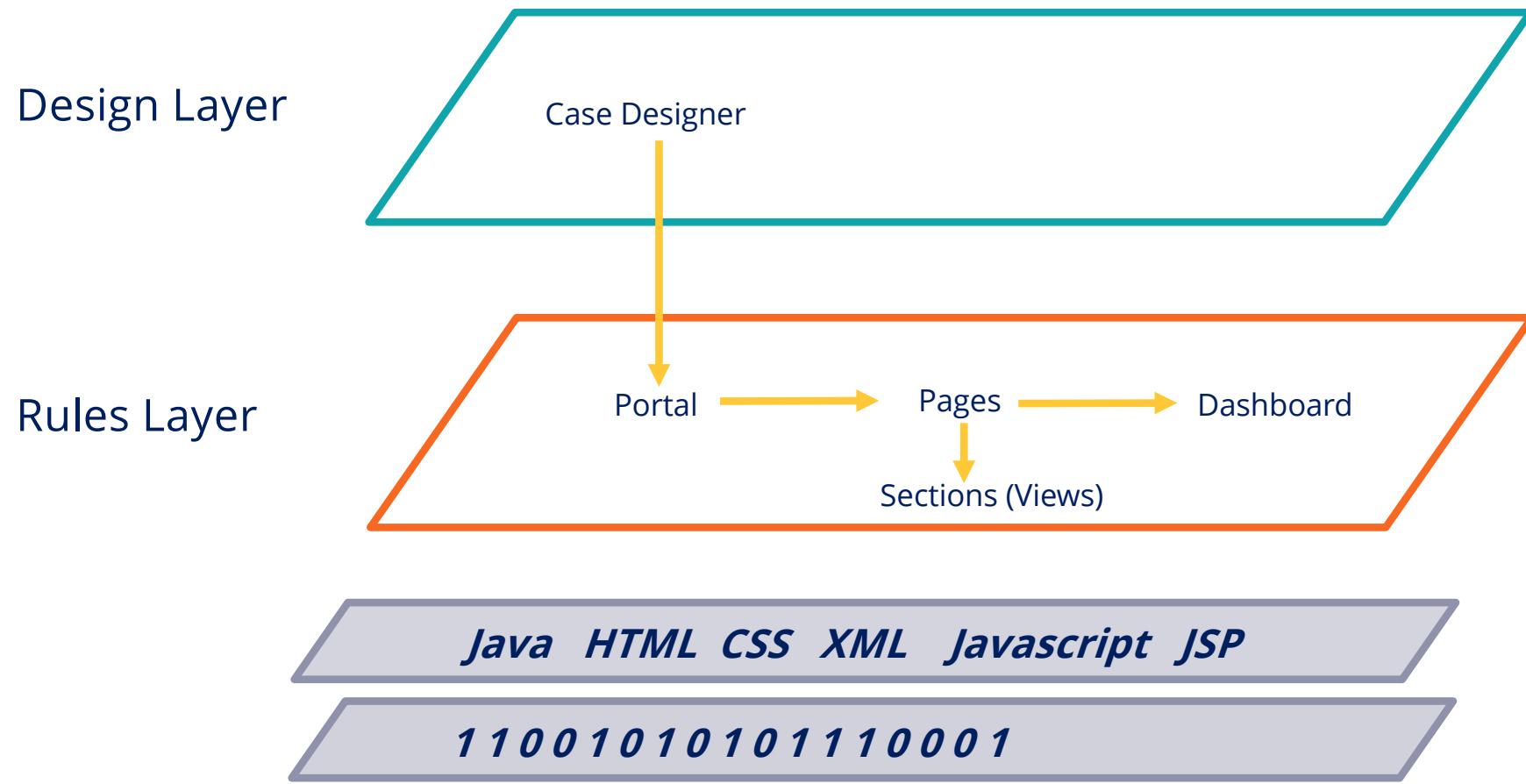
Expand the **Pages** menu and click **New**.

In the Create New window enter a page name and click Submit.





Schematic



Skill Mastery

- Understand
 - Channels
 - Portals
 - Dashboard



Best Practices

- Use a **noun** or **noun phrase** to describe the context of the section you are in.
- As much as possible, try to use no more than two words.
- Use names that are more meaningful and relevant to the business users.
- Build a UI that supports a more inclusive approach to users with disabilities.

Reporting



SKILL
LESSON

Reporting



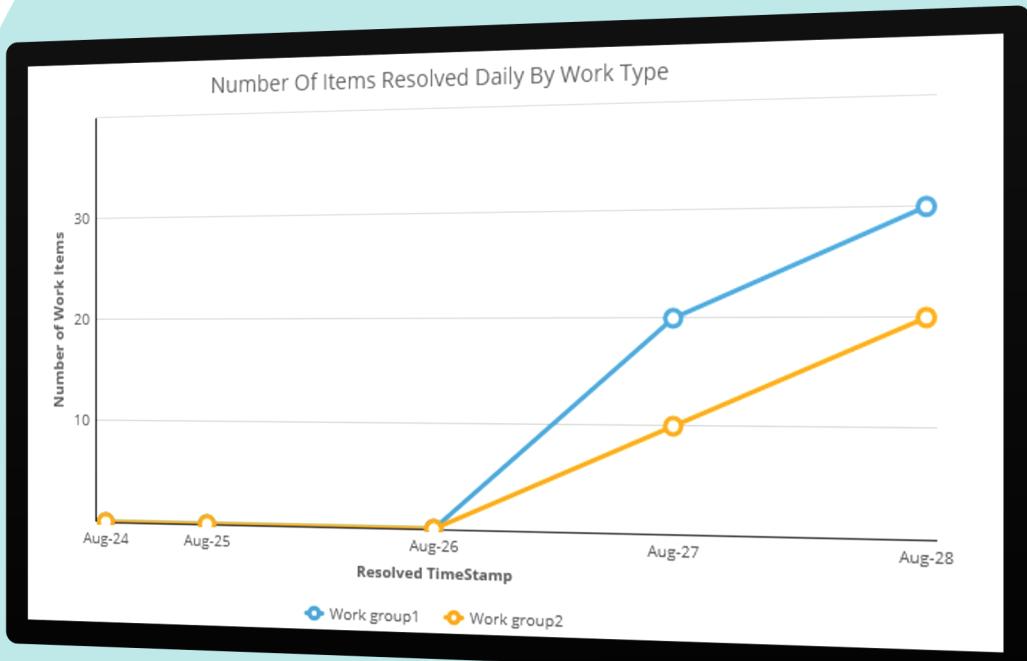
Overview

When designing reports, knowing what information the user needs and how it will be used is important.

Pega Platform provides reporting features for developers, business analysts, and managers so that you can create simple or complex reports to satisfy your business needs.

This module covers:

- Business vs Process Metrics
- The Report Browser
- Report Definitions
- Optimizing Reports
- Insights Dashboard



Use Case

- Business applications often target performance gains in time and efficiency to reduce costs.
 - Where are bottlenecks ?
 - Where there are opportunities to improve response time?
 - Emerging trends?
- Data generated by an application can show what is happening, what has happened over time, and whether these results match or differ from your plan.



Business Metrics

Description

What is the question?	The data indicates	What is the business decision?
What is the average profit margin for all automobile sales last year?	The average margin is below the target percentage.	The sales department decides to train its sales staff on promoting cars and options that have the highest margins.
What is the number of auto loans made in a month as compared to personal loans for the same period?	The number of personal loans is significantly lower than the number of auto loans.	The goal is to have the numbers approximately equal. The marketing department increases marketing resources for personal loans.
What is the number of office desks shipped each week for the past month, and how many are now in inventory?	The number of orders shows an upward trend.	As a result, inventory levels are unacceptably low. The purchasing department decides to restock more desks on a weekly basis.

Process Metrics

Description

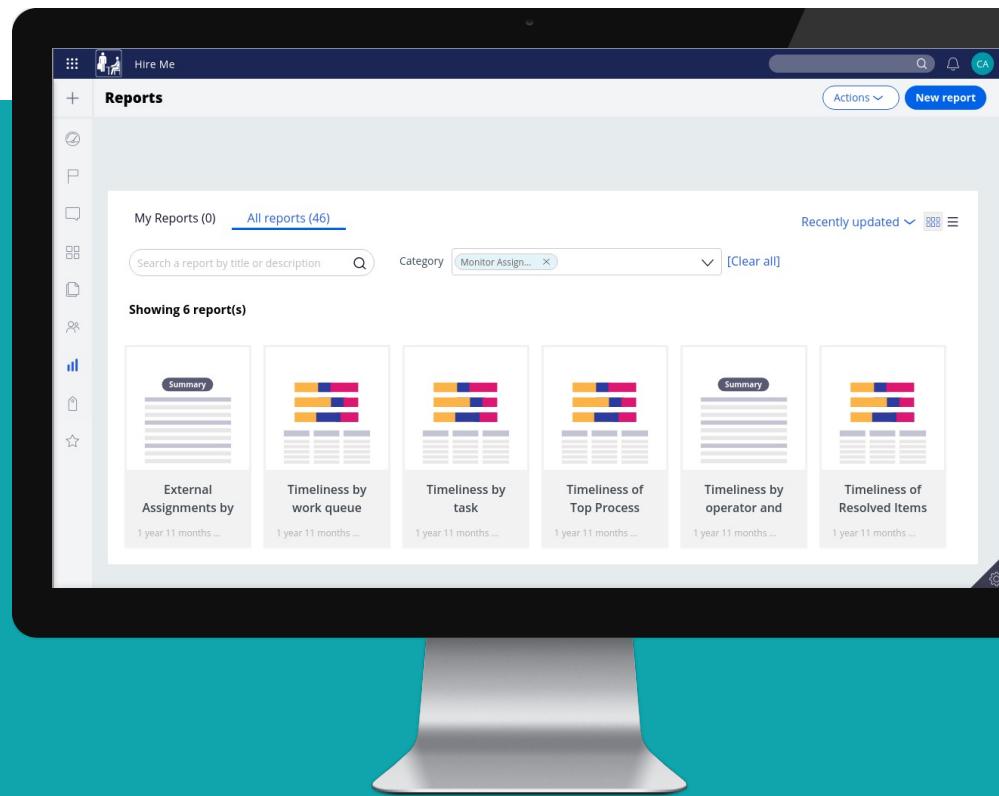
What is the question?	The data indicates	What is the business decision?
Which open loan application cases exceed the standard three-day service level deadline?	Most of the open cases are for loan amounts greater than USD 300,000.	Loan requests that exceed this amount must go through an additional review step, which accounts for the delay. The department manager decides to increase the service level deadline for loans exceeding 300,000 USD from 3 to 4 days.
What is the average duration of assignments by type and action?	This report identifies which user actions take the longest to complete.	Spend time on improving the efficiency of those assignments taking the most amount of time to complete.

Report Browser

Definition

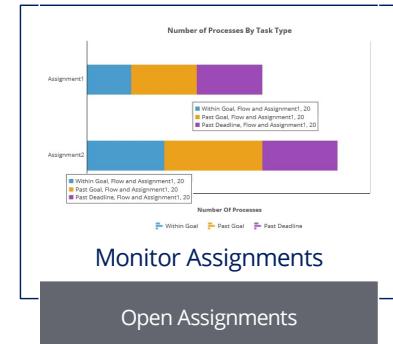
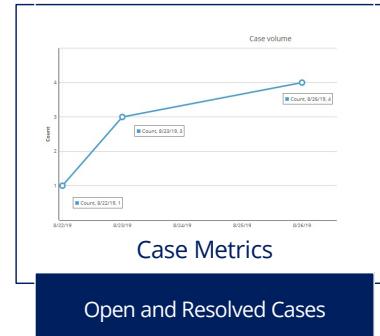
Used to create new reports, organize, run, and share reports, and manage report categories.

- **Private categories** provide shortcuts to reports that you create.
- **Public categories** provide shortcuts to standard Pega out-of-the-box reports that provide deep insight around the volume, process, and efficiency for case resolution as well as reports that users made available to others.



Standard Pega Public Report Categories

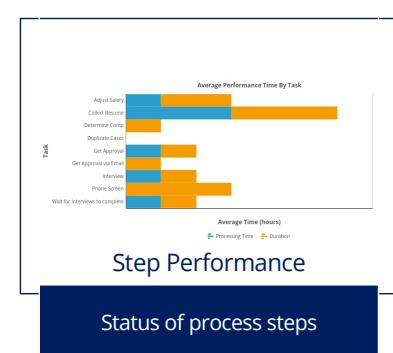
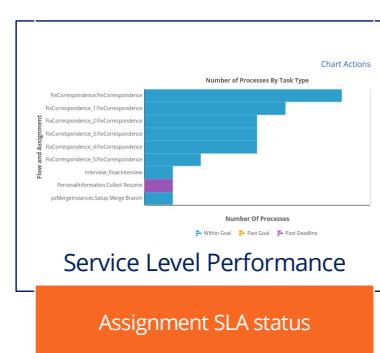
Description



Filtered by: Resolved By Me and Resolved Anytime and Any Resolution Timestamp				
ID	Descriptions	Status	Create Time	Resolved Time
S-1	Service Request	Resolved/Completed	8/19/19 4:34 PM	8/19/19 6:41 PM
S-2	Service Request	Resolved/Completed	8/20/19 1:06 PM	8/20/19 1:15 PM
S-3	Service Request	Resolved/Completed	8/21/19 1:58 PM	8/21/19 1:59 PM
S-4	Service Request	Resolved/Completed	8/21/19 4:35 PM	8/21/19 4:36 PM
S-6	Service Request	Resolved/Completed	8/23/19 4:36 PM	8/23/19 4:36 PM
S-7	Service Request	Resolved/Completed	8/23/19 4:44 PM	8/23/19 4:45 PM
S-5	Service Request	Resolved/Completed	8/26/19 4:22 PM	8/26/19 4:50 PM
S-10	Service Request	Resolved/Completed	8/26/19 5:26 PM	8/26/19 6:31 PM
S-11	Service Request	Resolved/Completed	8/26/19 5:28 PM	8/26/19 6:44 PM

Monitor Processes

Unresolved work items



Overview

When you create a report, you design the report in the Case Manager portal by creating the report definition rule and adding columns.

You can also use the Case Manager portal to add filters and make the report available in the Report Browser.

The screenshot shows the 'Edit columns' section of a report definition titled 'New Applicants [Available]'. The report is defined under the class 'HireMe-ApplicantMgr-Work-JobApplicant' with ID 'NewApplicants' and rule set 'ApplicantMgr-01-01-04'. The interface includes tabs for Query, Chart, Report Viewer, Data Access, Parameters, Pages & Classes, Test cases, Specifications, and History. The 'Query' tab is selected. The 'Edit columns' table lists the following columns:

Column source	Column name	Summarize	Sort type	Sort order
:: .pyStatusWork	Work Status	<blank>	<blank>	<blank>
:: .FullName	Full Name	<blank>	<blank>	<blank>
:: .Email	Email	<blank>	<blank>	<blank>
:: .pyID	Case ID	<blank>	<blank>	<blank>
:: .pxCurrentStageLabel	Current stage	<blank>	<blank>	<blank>
:: JobPosting.Title	Title	<blank>	<blank>	<blank>

Below the table are buttons for 'Add column' and 'Edit filters'. The 'Edit filters' section contains a placeholder 'Filter conditions to apply'.

Report Columns

Description

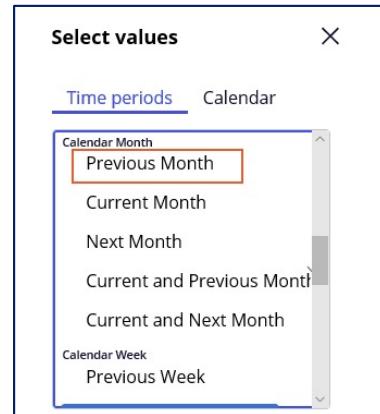
- **Columns** define the content of the report.
- Defining columns in reports is critical to designing reports users need.
- Each column corresponds to a single data element.
- The value in the column can be a value field or property such as a case ID, last modified date, or work status.
- You can format the data value in various ways, such as text, currency, or date.
- You can use calculations in columns to make reports more useful.
- Calculations, also known as functions, enable you to calculate results derived from data in the database.

Case ID	Employee	Hire Date	Location
O-101	James Martin	2/21/16	Atlanta
O-104	Anne Walker	2/4/16	Boston
O-746	Julia Phagan	1/12/16	Atlanta
O-983	William Kirk	9/8/16	Atlanta
O-171	Leonard Kelley	9/5/16	Boston
O-623	Kate Picardo	2/25/16	Atlanta
O-421	Robert Wang	2/1/16	Boston

Report Filters

Description

- By default, report queries return all the records that contain data from all the columns.
- A **report filter** compares a data value in the record against a defined condition.
 - Filters limit results to relevant records.
 - Records that fail the filter comparison are omitted.
 - Symbolic dates let you select time periods or dates without having to build functions.



Case ID	Employee	Hire Date	Location
O-101	James Martin	2/21/16	Atlanta
O-104	Anne Walker	2/4/16	Boston
O-746	Julia Phagan	1/12/16	Atlanta
O-983	William Kirk	9/8/16	Atlanta
O-171	Leonard Kelley	9/5/16	Boston
O-623	Kate Picardo	2/25/16	Atlanta
O-421	Robert Wang	2/1/16	Boston

A diagram illustrates the filtering process. It shows a wide blue oval at the top, with several curved lines entering it from the left. A dark blue funnel shape narrows the flow downwards. An arrow points from the top of the funnel to a second, narrower blue oval at the bottom. This visual metaphor represents how a filter (the funnel) narrows down the scope of the data (the ovals) from a general input to a specific output.

Case ID	Employee	Hire Date	Location
O-983	William Kirk	9/8/16	Atlanta
O-101	James Martin	2/21/16	Atlanta
O-746	Julia Phagan	1/12/16	Atlanta
O-623	Kate Picardo	2/25/16	Atlanta

Reporting Data

Description

Summary reports analyze detail data

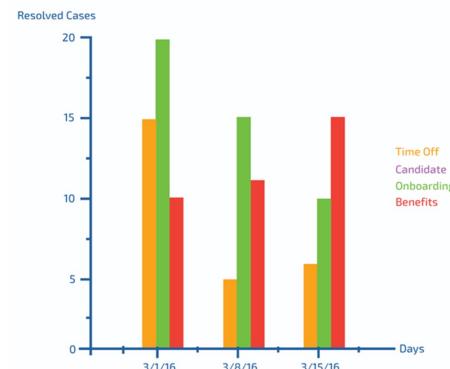
- Summary reports allow users to quickly identify key statistics.
- Users can “drill down” to another report with additional details.

Manager	Case ID	Office
Harry	O-101	Atlanta
Fred	O-83	Boston
Fred	O-99	Boston
Harry	O-64	Atlanta
Harry	O-171	Atlanta
Harry	O-623	Boston

Manager	Case ID	Office
Harry	4	Atlanta
Fred	2	Boston

Charts change information visually

- Help users quickly review and analyze data.
- Add charts to summary reports.
- One aggregate property must be present in order to add a chart to the report.



Sort report to organize

- Sort column values in ascending or descending order.
- Specify a sort order to sort multiple columns.

Employee	Start Date	Location
Kate Picardo	3/3/16	Atlanta
James Martin	2/9/16	Atlanta
Anne Walker	4/12/16	Boston
Robert Wang	2/9/16	Boston
Julia Phagan	3/8/16	Atlanta
William Kirk	2/8/16	Atlanta
Kelly Smith	4/5/16	Boston

Employee	Start Date	Location
Anne Walker	4/12/16	Boston
Kelly Smith	4/5/16	Boston
Julia Phagan	3/8/16	Atlanta
Kate Picardo	3/3/16	Atlanta
James Martin	2/9/16	Atlanta
Robert Wang	2/9/16	Boston
William Kirk	2/8/16	Atlanta

Data Explorer and Insights

Description

- The **Explore Data** landing page allows you to explore and analyze data in your application quickly.
- On the Explore Data landing page, you can query your data, and then sort, filter, and group the results based on your business requirements.
- Insights** are rules that Pega Platform™ uses to transform data queries into tables or visualizations that you can then share between users.
- You can use insights to retrieve specific data and present the data as a list or an interactive chart.
- For example, in an insurance application, you can visualize data for claims that are pending approval, and then analyze the type of claim and the amount submitted by users.

The screenshot shows the Pega Explore Data interface. On the left is a dark sidebar with icons for search, plus, up, down, clock, envelope, and a bell with a red notification badge containing the number '3'. The main area has a header 'Explore Data' with a magnifying glass icon. Below it is a dropdown menu 'What would you like to explore?' set to 'Case Type', with a red circle labeled '1' next to it. The main content area is titled 'Insights' and shows a table with 5 results. The table has columns: Name, Update Time (sorted descending), Insight Type, and Visibility. The first row under 'Name' is expanded to show 'Class (4) Auto insurance claim'. The table contains the following data:

Name	Update Time	Insight Type	Visibility
Class (4) Auto insurance claim			
Auto insurance claims by amount	February 15, 2021 10:00 AM	exploration	SHARED
Auto insurance claims by stage	February 13, 2021 3:30 PM	exploration	SHARED
Auto insurance claims by status	February 12, 2021 1:05 PM	chart	SHARED
Auto insurance claims by date	February 12, 2021 9:15 AM	chart	PRIVATE
Class (1) Homeowners insurance claim			
Homeowners insurance claim by date	February 15, 2021 11:00 AM	exploration	PRIVATE

At the top right of the table are buttons for 'Filter', 'Sort', 'Group', and more. A red circle labeled '2' is over the 'Name' column header, and a red circle labeled '3' is over the 'Filter' button.

View or Modify an Insight

Description

1. Save an insight.
2. Perform an action on an insight.
3. Access application data.
4. View data based on business needs.
5. Customize columns.

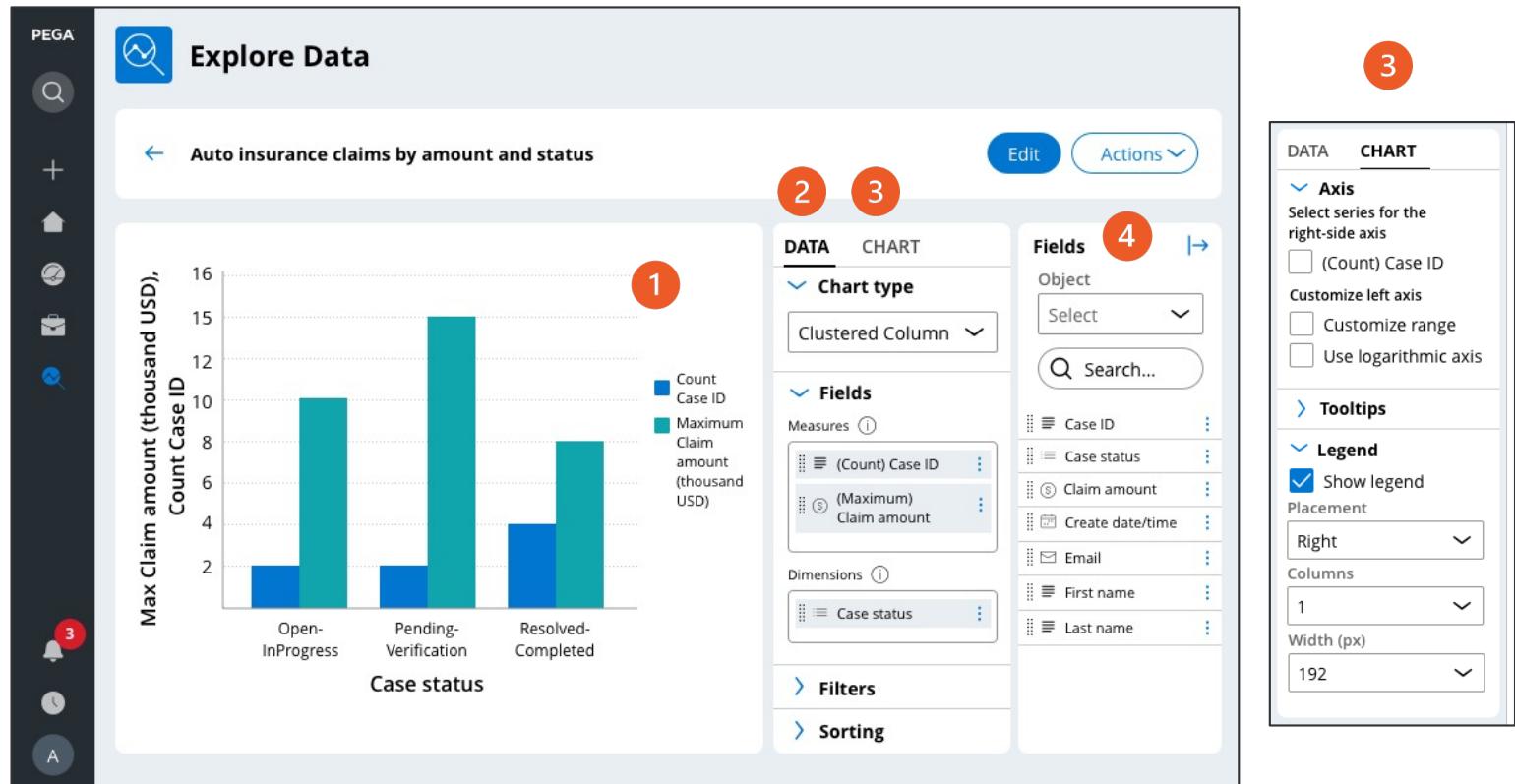
The screenshot shows the Pega Explore Data interface. On the left is a dark sidebar with the PEGA logo at the top and several icons: a magnifying glass (Search), a plus sign (New), a house (Home), a gear (Settings), a briefcase (Data), and a magnifying glass (Explore). A red circle with the number 3 is positioned over the explore icon. To the right is a light-colored main area with a header titled "Explore Data". Below the header is a search bar with the placeholder "What would you like to explore?" and a dropdown menu set to "Select". In the top right corner, there are three buttons: "Save as insight" (red circle 1), "Actions" (blue button), and another "Actions" button (red circle 2). Below the search bar is a search input field with the placeholder "Search..." and a result count of "8 Results". To the right of the search input are five icons: Filter (red circle 4), Sort (blue icon), Group (green icon), and a more options menu (red circle 5). The main content area is a table with the following data:

Name	Create date	Status	Claim amount
AI-1008	February 15, 2021	RESOLVED-COMPLETED	USD8000
AI-1007	February 15, 2021	OPEN-INPROGRESS	USD5000
AI-1006	February 13, 2021	OPEN-INPROGRESS	USD10000
AI-1005	February 12, 2021	PENDING-VERIFICATION	USD7500
AI-1004	February 12, 2021	RESOLVED-COMPLETED	USD2000
AI-1003	February 11, 2021	RESOLVED-COMPLETED	USD4500
AI-1002	February 10, 2021	PENDING-VERIFICATION	USD15000
AI-1001	February 10, 2021	RESOLVED-COMPLETED	USD3000

Create a chart from an insight

Description

1. Chart Preview
2. Data tab
3. Chart tab
4. Fields pane





Allowing Managers to create new reports

Description

- To enable Managers to create and modify Report Definitions, report shortcuts, and report categories using the Report Browser, you must create a dedicated ruleset version for manager reports.
- Ruleset** - One unlocked ruleset version of the security type Standard. **Use check out** is disabled.
- Operators** - Verify **Allow rule check-out** is enabled on the security tab
- Application rule** - Ruleset is identified as a **production** ruleset
- Access Group** - Advanced tab identifies the ruleset as a **production ruleset** and the **Default destination ruleset**

A screenshot of a web-based application interface, likely the Pega Report Browser. In the center, there is a card for a report titled "Pega Robot Yearly Report" which was created "6 days ago". To the right of the card is a context menu with several options: "Open", "Edit", "Move", "Copy", "Remove", "Delete", and "Schedule". The "Edit" option is highlighted with a red rectangular box. The background of the browser shows a grid of other report cards, and the overall theme includes yellow diagonal stripes on the left side.



Report Definitions

Definition

- Report definition is used to build a report.
- The report definition retrieves records from a database and returns the results.
- **Report columns** define the report's contents.
 - Each column corresponds to a single data element.
 - The column references a value property such as a case ID or work status.
 - You can format the data value in various ways, such as text, currency, or date.
- **Functions** in columns make reports more useful.
 - They allow users to calculate results derived from data in the database.
 - Pega provides many standard functions to use in the report definition.

Employee	Hire Date	Location
O-101	James Martin	2/21/16
O-104	Anne Walker	2/4/16
O-746	Julia Phagan	1/12/16
O-983	William Kirk	9/8/16
O-171	Leonard Kelley	9/5/16
O-623	Kate Picardo	2/25/16
O-421	Robert Wang	2/1/16

Start Date	Employee
Location: Boston	
4/12/16	Anne Walker
4/5/16	Kelly Smith
2/9/16	Robert Wang
Location: Atlanta	
3/3/16	Kate Picardo
2/9/16	James Martin
3/8/16	Julia Phagan
2/8/16	William Kirk

Group results to organize the data in large list reports.



Creating a Report

Description

- Dev Studio > App explorer > {class} > Reports > Report definition
 - Create a report definition rule
 - Add columns to define the data returned
 - Add a filter to limit the records returned
 - Make the report available in the Report Browser

The screenshot shows the Pega Dev Studio interface with the following details:

- Left Sidebar:** Shows the application structure with "2 applications". The current class selected is "GoGoR-GoGoRoadR-Work".
- Report Definition Editor:**
 - Title Bar:** "Edit Report Definition: Assistance request [Available]"
 - Header:** "CL: GoGoR-GoGoRoadR-Work-AssistanceRequest" and "ID: DataTableEditorReport RS: GoGoRoadR:01-01-01". A yellow bar indicates "This record has 1 unreviewed warning" with a "Review/Edit" button.
 - Navigation:** "Query", "Chart", "Report Viewer", "Data Access", "Parameters", "Pages & Classes", "Specifications", "History".
 - Columns:** "Edit columns" section lists columns with their source and display names:

Column source	Column name	Summarize	Sort type	Sort order
::.pyID	Case ID	<blank>	Lowest to Highest	1
::.pyLabel	Label	<blank>	<blank>	
::.pxCreateDateTime	Creation date/time	<blank>	<blank>	
::.pxCreateOpName	Created by	<blank>	<blank>	
::.pyStatusWork	Case status	<blank>	<blank>	
 - Filters:** "Edit filters" section with a "Filter conditions to apply" input field containing "A" and a table:

Condition	Caption	Column source	Relationship	Value
A		<blank>		Select values



Data Access Tab Options

Description

- Report Definition > Data Access > General Data Access Settings panel > Data retrieval preference

Select the preferred method for retrieving data:

- **Elastic search** - optimizes reporting without putting additional requirements on the database.
 - To improve report generation performance, you can now run report definitions against Elasticsearch indexes instead of using SQL queries directly against the database.
 - This feature is disabled by default and does not apply to reports that have features that are not supported by Elasticsearch. If a report query cannot be run against Elasticsearch indexes, Pega® Platform automatically uses an SQL query.
- **Database** – used if Elasticsearch cannot be used to retrieve data.

Data retrieval preference

Prefer Elasticsearch index
 Use the database (default)

Reporting database

Prefer reporting data source if defined ▾
Primary data source only
Prefer reporting data source if defined



Information Retrieval

Implementation

Edit Report definition: Services offered [Available]
CL: GoGoR-GoGoRoadR-Data-ServicesOffered ID: DataTableEditorReport RS: GoGoRoadR:01-01-01

This record has 1 info warning (including 1 unjustified) [Review/Edit](#)

Query Chart Report Viewer Data Access Parameters Pages & Classes Test cases Specifications History

Edit columns

Column source	Column name	Summarize	Sort type	Sort order
.pyGUID	ID (required)	<blank>	Lowest to Highest	1
.pxCreateDateTime	Create Date/Time	<blank>	<blank>	
.pxCreateOperator	Create Operator	<blank>	<blank>	
.pxCreateOpName	Create operator name	<blank>	<blank>	
.pxCreateSystemID	Create System ID	<blank>	<blank>	

REPORT DEFINITION RULE

Record Count: 8					
Description	Line total	Quantity	Service	Unit cost	
Unlock a locked vehicle	0E-9		Unlock	\$25.00	
Replace a damaged tire with an equivalent model	0E-9		Tire replacement	\$100.00	
Replace the car battery with a compatible OEM model	0E-9		Battery replacement	\$150.00	
Reseal a leaking tire	0E-9		Tire repair	\$25.00	
Recharge the car battery	0E-9		Battery charge	\$40.00	
Provide the driver with a replacement key for the vehicle	0E-9		Replacement key	\$50.00	
Tow the vehicle the specified distance	0E-9		Tow vehicle	\$1.00	
Refuel the vehicle with the specified quantity of gasoline	0E-9		Gasoline	\$10.00	

REPORT INFORMATION

Pega Clipboard HX07403ACALO30V4816XW85NHRM29LNA

Data Page: D_ServicesofferedListR Edit Refresh Actions

Property Value

- pxMore false
- pxObjClass Code-Pega-List
- pxPageCount 0
- pxQueryTimestamp 20200214T145524.915 GMT
- pxResultCount 8

```
SELECT "PCO"."pyguid" AS "pyGUID", "PCO"."pxcreatedatetime" AS "pxCreateDateTime", "PCO"."pxcreateoperator" AS "pxCreateOperator", "PCO"."pxcreateopname" AS "pxCreateOpName", "PCO"."pxcreatesystemid" AS "pxCreateSystemID", "PCO"."pxupdateoperator" AS "pxUpdateOperator", "PCO"."pxupdateopname" AS "pxUpdateOpName", "PCO"."pxupdatesystemid" AS "pxUpdateSystemID", "PCO"."pxupdatedatetime" AS "pxUpdateDateTime", "PCO"."description" AS "Description", "PCO"."linetotal" AS "LineTotal", "PCO"."quantity" AS "Quantity", "PCO"."service" AS "Service", "PCO"."unitcost" AS "UnitCost" FROM customerdata.pr_gogor_gogorade_data_serve2880_1318 "PCO" ORDER BY 1 ASC
SELECT "PCO"."pyguid" AS "pyGUID", "PCO"."pxcreatedatetime" AS "pxCreateDateTime", "PCO"."pxcreateoperator" AS "pxCreateOperator", "PCO"."pxcreateopname" AS "pxCreateOpName", "PCO"."pxcreatesystemid" AS "pxCreateSystemID", "PCO"."pxupdateoperator" AS "pxUpdateOperator", "PCO"."pxupdateopname" AS "pxUpdateOpName", "PCO"."pxupdatesystemid" AS "pxUpdateSystemID", "PCO"."pxupdatedatetime" AS "pxUpdateDateTime", "PCO"."description" AS "Description", "PCO"."linetotal" AS "LineTotal", "PCO"."quantity" AS "Quantity", "PCO"."service" AS "Service", "PCO"."unitcost" AS "UnitCost" FROM customerdata.pr_gogor_gogorade_data_serve2880_1318 "PCO" ORDER BY 1 ASC
```

REPORT DATA ON THE CLIPBOARD



Overview

Pega applications allow application developers to improve report performance through a process called optimization.

To store case data, Pega writes to a BLOB field, which provides greater flexibility and performance for case data.

The screenshot shows the 'Property Optimization' dialog box. At the top, there are three tabs: 'Properties and Classes' (selected), 'Eligible Classes', and 'Optimization'. Below the tabs, a note states: 'Pega Platform creates a column, VehicleDetail.Color, in the database table and maps it to the property VehicleDetail.Color in the classes listed below. If this is a new property, the column value is NULL.' A section titled 'VehicleDetail.Color will be exposed as a VARCHAR(256) column -- to change the size of this column, set the 'Max length' field on the 'Advanced' tab of the property definition.' follows. Under 'Property Optimization for VehicleDetail.Color', there is a table with two columns: 'Classes (and their number of Instances)' and 'Database'. The table contains one row: 'BCS-EVehicle-Work-SellVehicle(6)' under 'Classes' and 'pc_BCS_Evehicle_Work(6)' under 'Database'. A note below the table says: 'NOTE: The list above does not show classes that are already optimized and mapped to a table without a BLOB'. At the bottom, there is a 'Population Schedule at:' section with radio buttons for 'Now' (selected) and 'Later'. At the very bottom of the dialog are 'Cancel' and 'Next >>' buttons.

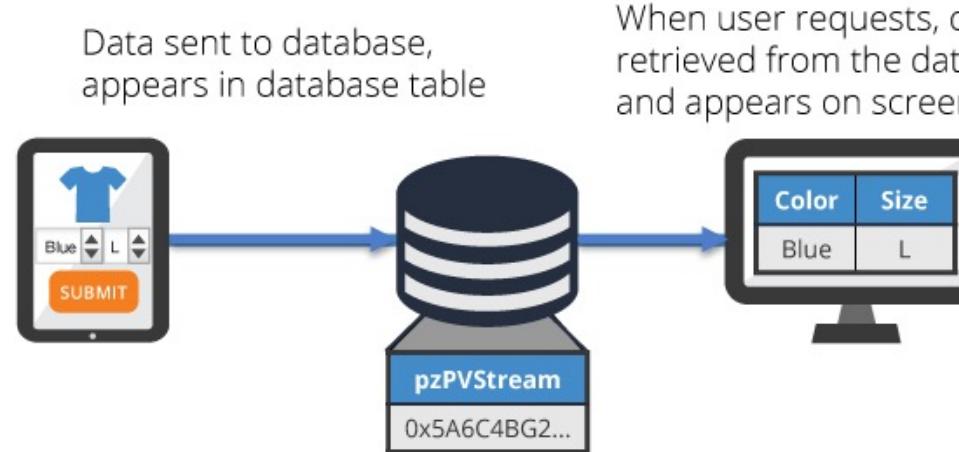


Data Storage

Definition

Pega Platform applications store each case as a unique record in a relational database in a **binary large object** (BLOB) field.

- Within the database table:
 - Each record is a row
 - Each column displays the contents of a field from the case record, including the BLOB field
- When an end user opens a case, Pega locates the record and reads the contents of the BLOB column of the table to extract the case data.

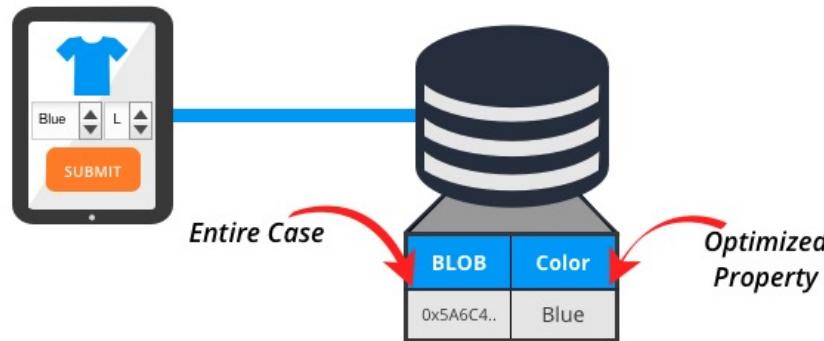




Property Optimization

Description

- To improve report performance, Pega offers a hybrid data storage model.
- Data is stored both in dedicated indexed columns and in a BLOB field.
- Optimize the property for reporting to store property values in an indexed column.
- Extracting data from a BLOB impacts performance compared to reading property values from a database table column.
- This impact is most pronounced when extracting data for report filters and sorting or grouping the content of a column.
- Pega's data storage model improves report performance with optimization.



Advantages

- “Unlimited” storage size
- Flexibility on the data model structure
- High performance for transaction

Disadvantages

- Penalizes performance for reporting
- Decompressing the BLOB fields significantly increases the time needed to run a report.

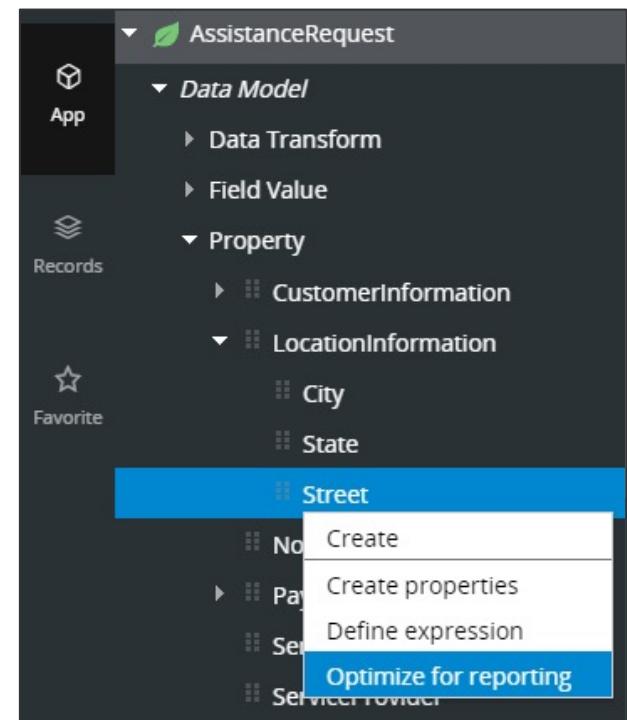


Optimize for reporting

Implementation

Improve reporting and searching performance by optimizing, or exposing, Single Value properties as distinct columns. Pega provides the Property Optimization tool to optimize properties for reporting.

- When a property is optimized, Pega creates a column for the property in a database table.
- Optimizing a property for reporting is also called “exposing” the property.
- When a case or report uses an optimized property, Pega:
 - Writes data to both the property field and the BLOB field
 - Reads from the property column rather than the BLOB





Optimizing Properties for Reporting

Implementation

Property Optimization

1. ✓Properties and Classes
2. ✓Eligible Classes
3. Optimization

Optimization started successfully for the following classes. A background thread is updating the instances of the below classes by copying the property value to the newly created column.

▼ Class-InstanceCount

Class Name	No of Instances
1 GoGoR-GoGoRoadR-Work-AssistanceRequest	1

To see the status of the jobs, click here... [Column Population Jobs Dashboard](#)

System-Database

Schema Change Tracking Optimize Schema Modify Schema [Column Population jobs](#) Refresh Help X

[Refresh All](#)

Job ID	Created at	Created by	Scheduled at	On Node(s)	For Property(ies)	In Class	Instances	Processed %	Status
JOB-1	Jun 8, 2020 5:01:25 PM	author_rep@gogoroad	6/8/20 5:01 PM		InvoiceTotal	GoGoR-GoGoRoadR-Work-AssistanceRequest	1		New



.px .py .pz Properties

Description



Optimization reduces the time and memory needed to run the report because decompressing does not occur.



By default, Pega optimizes many properties that store process data. Some of these include:

The creation date of a case
The status of a case
The case ID



Pega properties that store process data begin with the letters *px*, *py*, or *pz*.



Guardrail Warnings

Description

- Properties created to store business data are not optimized by default.
- Reports that use an unoptimized property display a warning that states the potential impact on performance.
- These warnings due to an unoptimized property are resolved by running the Property Optimization tool.

This record has 1 severe or moderate warning (including 1 unjustified) and 3 info warnings [Review/Edit](#)

Data Access X

Guardrail warnings

Performance	Not optimized for reporting
These properties are not optimized: .VehicleDetail.Color, .VehicleDetail.FuelType, .VehicleDetail.Model. Displaying them may result in poor performance. Consult your system administrator about optimizing properties for reporting. Incorrect practice must be justified *	
<input type="button" value="Add justification"/>	



Data Relationship properties

Description



When the property has a one-to-one relationship with the case, Pega will extend the case database table with one column.



When the property has a one-to-many relationship with the case Pega will need to create a separate database table to store the information.

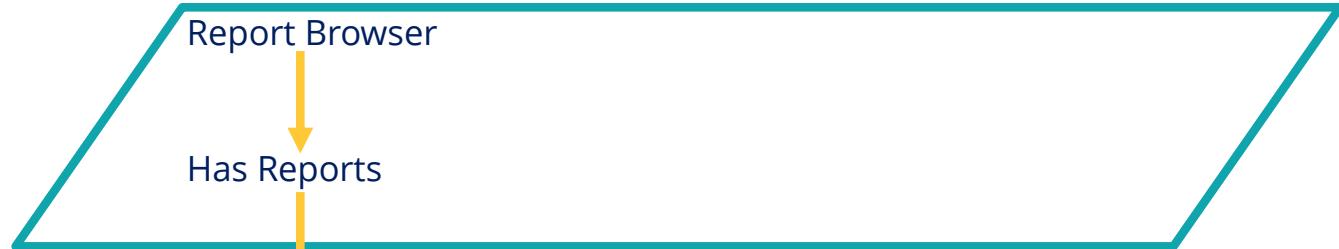


Among the rules generated by the wizard you will find a Declare Index rule



Schematics

Design Layer



Rules Layer



Java HTML CSS XML JavaScript JSP

11001010101110001

Skill Mastery

- The role of reports
- Business and process reports
- The Report Browser
- Creating a report
- Organizing a report



Security



SKILL
LESSON

Inviting users to an Application



Overview

The Pega platform provides a comprehensive set of features for configuring users and roles associated with them.

Roles determine what users may do in the application controlling such things as security and work assignments.

Applications are accessed by users, but users undertake different roles.

Role	Channel interface	Description
Name		
Administrator	Admin Studio	Generated by New Application API
Author	App Studio	Generated by New Application API
Manager	Case Manager	Generated by New Application API
User	Case Worker	Generated by New Application API

Use Case

- A mechanism to give users access to applications:
 - Users are assigned roles based on how they interact with the application
 - Users interact through channels
 - Personas help group users according to the responsibilities that they assume



Definitions

Users

An individual who interacts with an application

Roles

Defines how users interact with the application

Channels

Denotes where data associated with a data type is located

Persona

Defines the type of access that you grant to the users of your application

Access Groups

An access group is a group of permissions within an application

Operator

An operator defines a unique identifier, password, preferences, and personal information for a user

Users

Definition

- A user is an individual who interacts with an application
 - Users are also called Operators
- Users are created with an email address and an assigned role together with other attributes
- Multiple users are often associated with the same role when they interact with the application in the same ways

The screenshot shows the 'Edit Operator ID: Administrator@pega.com' screen. The top navigation bar includes 'Delete', 'Actions', 'Save', and a close button. Below the navigation is a tabs menu with 'Profile' (selected), 'Work', 'Security', and 'History'.

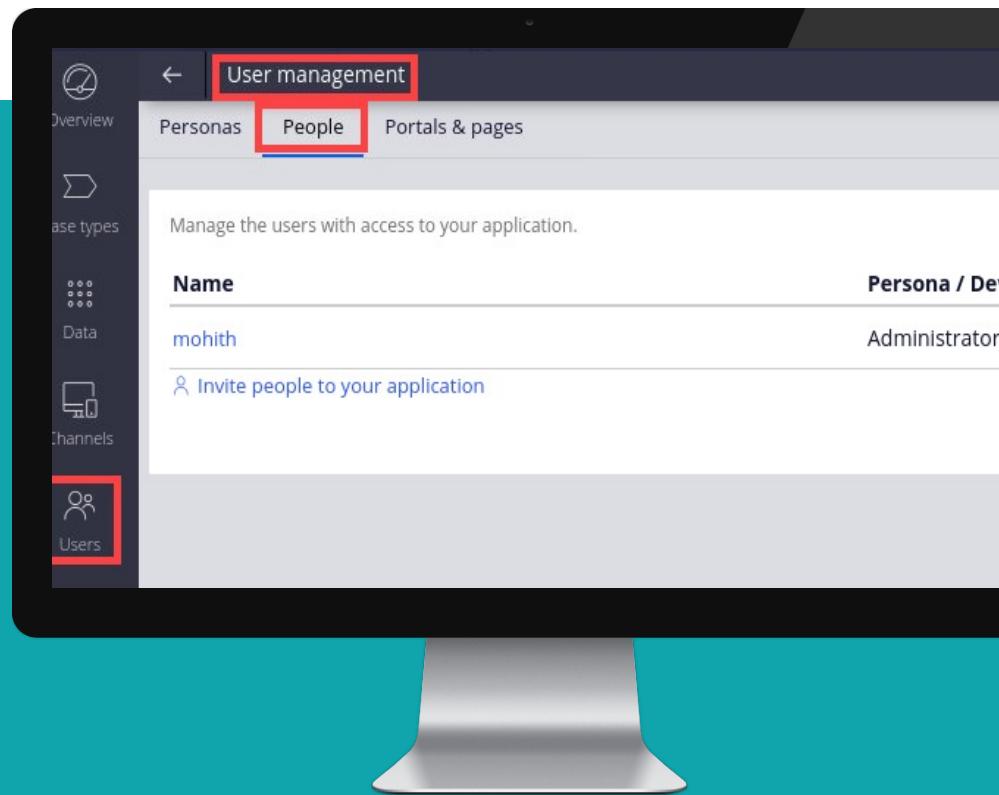
The main content area is divided into several sections:

- Contact Information:** Includes fields for Profile picture (choose file, upload image), Title, First name, Last name, and Full name (Administrator).
- Application Access:** A section titled 'Use the radio button to select a default access group' with two options: 'PRPC:Administrators' (selected) and 'DMSample:Administrators'. It lists applications: 'PegaRULES 8' and 'DMSample 01.01.01'.
- Localization:** A section for setting localization preferences.
- Personal information:** Fields for Full Name (mohith), Email (mohithg019@gmail.com), and Telephone.
- Professional information:** Fields for Title, Role (Administrator), Business unit (Unit), and Reports to.
- Skills:** A section showing 'Skill' and 'No items' with a '+ New skill' button.
- Proficiency:** A large yellow area with a grid of blue and orange dots.

Viewing users

Navigation

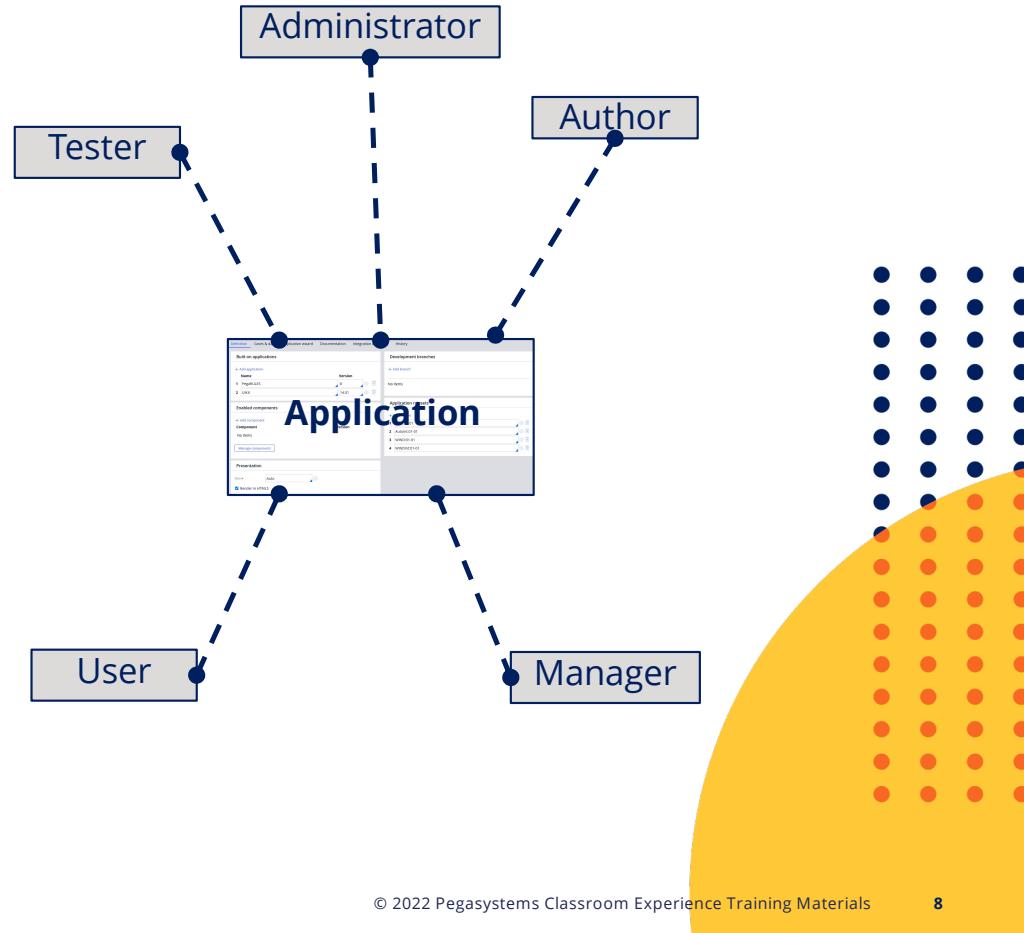
- To view from App Studio select:
Users > User management > People
- To view users from Dev Studio select:
Configure > Org & Security > Organization > Operators



Roles

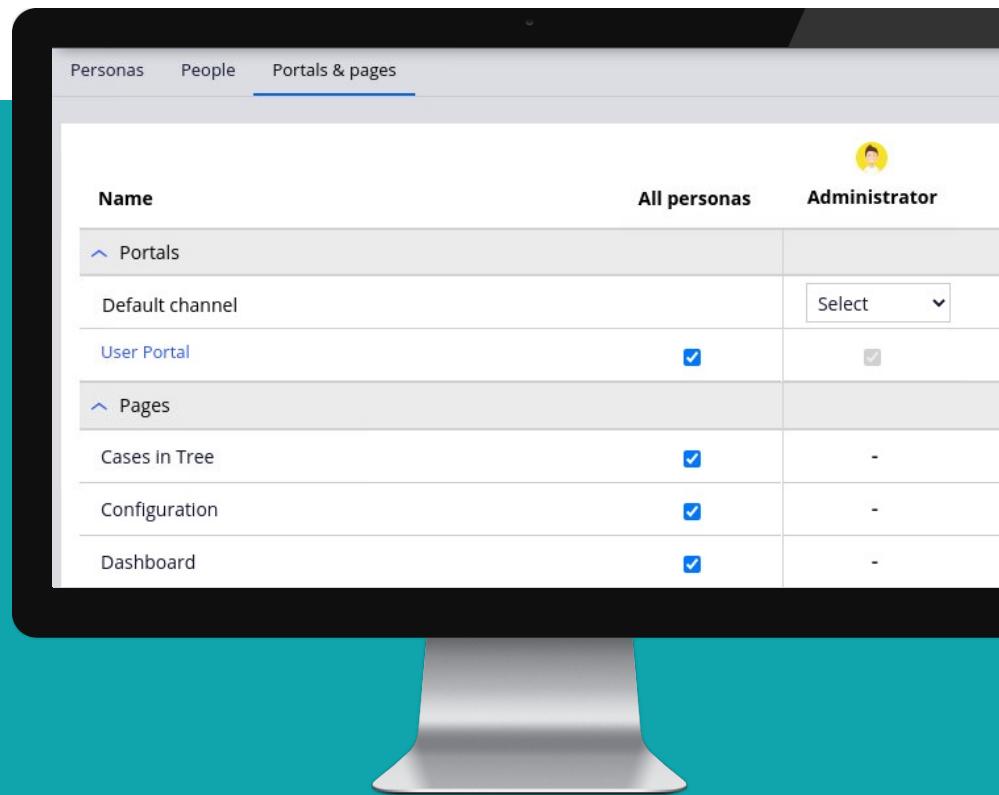
Definition

- A role defines how users interact with the application including:
 - What user interface is presented
 - What users can or cannot do
- When you create an application, four roles are created by default:
 - Admin, Author, User, and Manager  
 - Additional roles can be created for unique combinations of the user interfaces, page permissions, and work routing  
- User interfaces are also referred to as channel interfaces.
- Roles can be associated to Studio channels (App Studio, Admin Studio, and Dev Studio) and Web Interfaces (such as default Case Manager and Case Worker or newly created Web Interfaces).



Navigation: Viewing roles

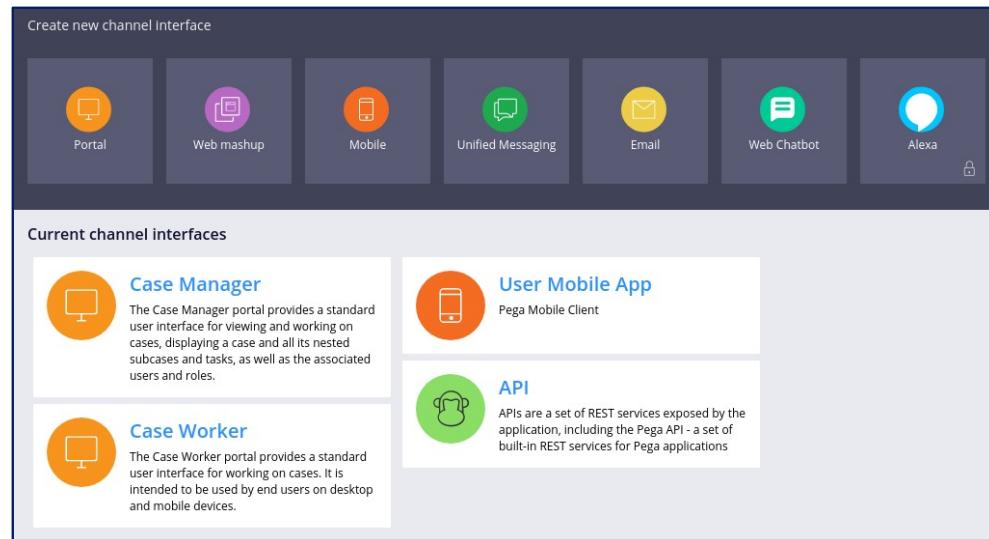
- To view and configure roles from App Studio select:
Users > User Management > Portals & pages
- Assign a Default channel and associate pages



Channels

Definition

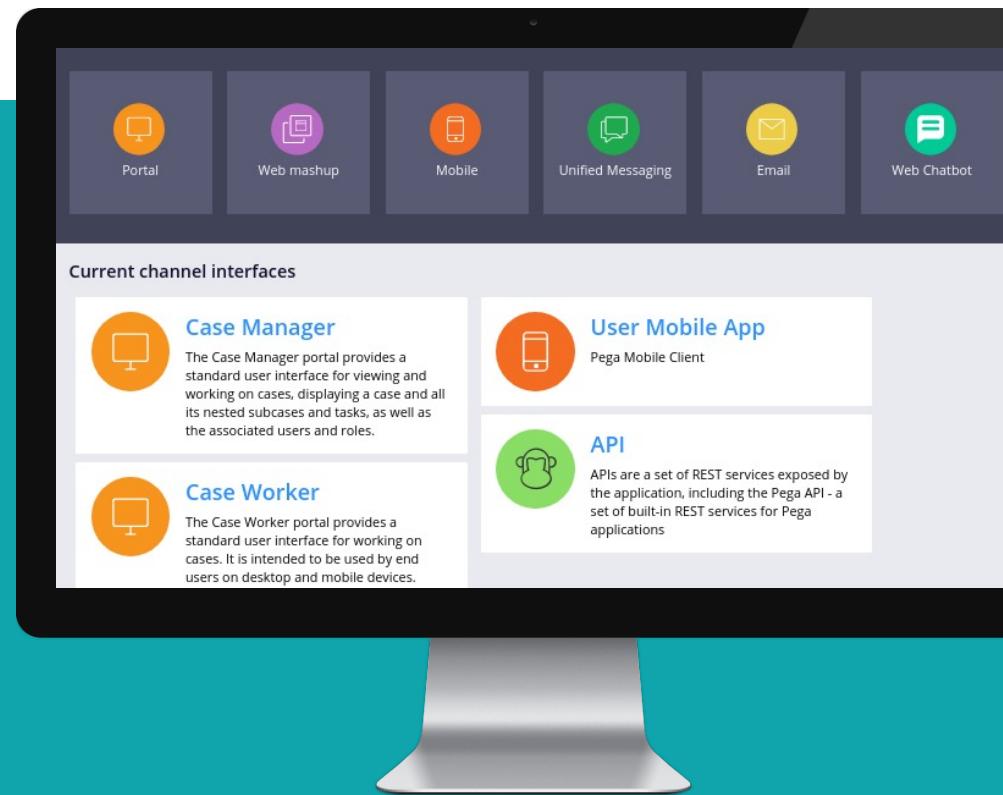
- User interfaces are also referred to as channel interfaces.
- Conversational channels help organizations reach additional users of applications.
 - Allows seamless interactions with the Pega Platform
- Roles can be associated to Studio channels (App Studio, Admin Studio, and Dev Studio) and Web Interfaces (such as default Case Manager and Case Worker or newly created Web Interfaces).



Viewing and assigning Channels

Navigation

- To access and manage channels from App Studio select: **Channels**



Personas

Definition

- Personas define type of access that you grant to the users of your application
- Personas are a design tool to help group users according to the responsibilities that they assume within a process, cases on which they work, and the channels that they can access
- You can manage users intuitively through personas, that store comprehensive information about the roles and access rights of all stakeholders in a process
 - Provides for a more granular level of control over the user experience
- Create as many personas as you like, and use them as reference for building access groups in your application

Include draft relationships ⓘ

Personas	Channels	Case types	Users
CSR	Case Manager	Evaluate and Sell a Vehicle	0 users  
CSG			0 users  



Viewing, configuring and using Personas

Navigation

- To view and configure personas from App Studio select:

Users > User management > Personas

- To add a Persona to case type:

Case type > + PERSONA

The screenshot shows the 'User management' screen with the 'Personas' tab selected. A search bar at the top says 'Search personas...' and shows '1 persona'. Below it, there's a table with three columns: 'Persona', 'Channels', and 'Users'. One row is visible for the 'Administrator' persona, which is associated with the 'User Portal' channel and 1 user.

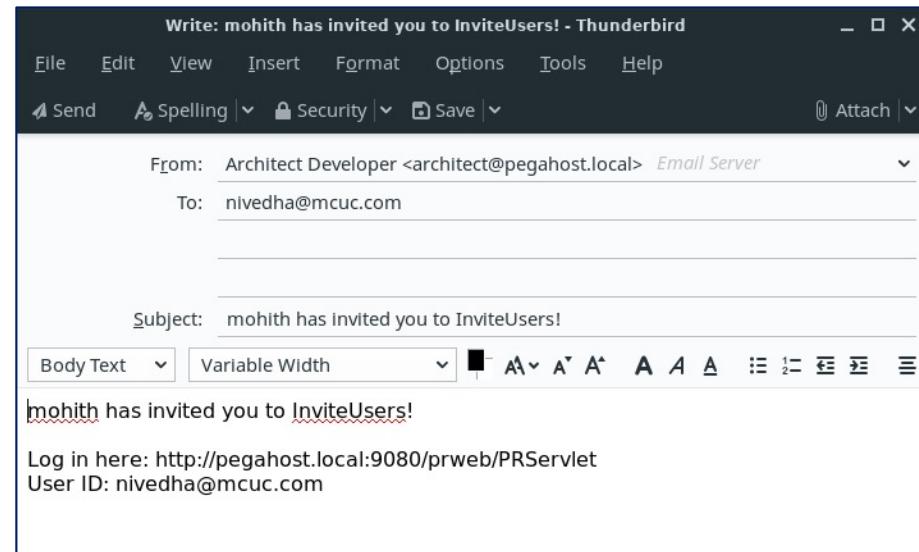
Persona	Channels	Users
Administrator	User Portal	1 user



Inviting Collaborators

Description

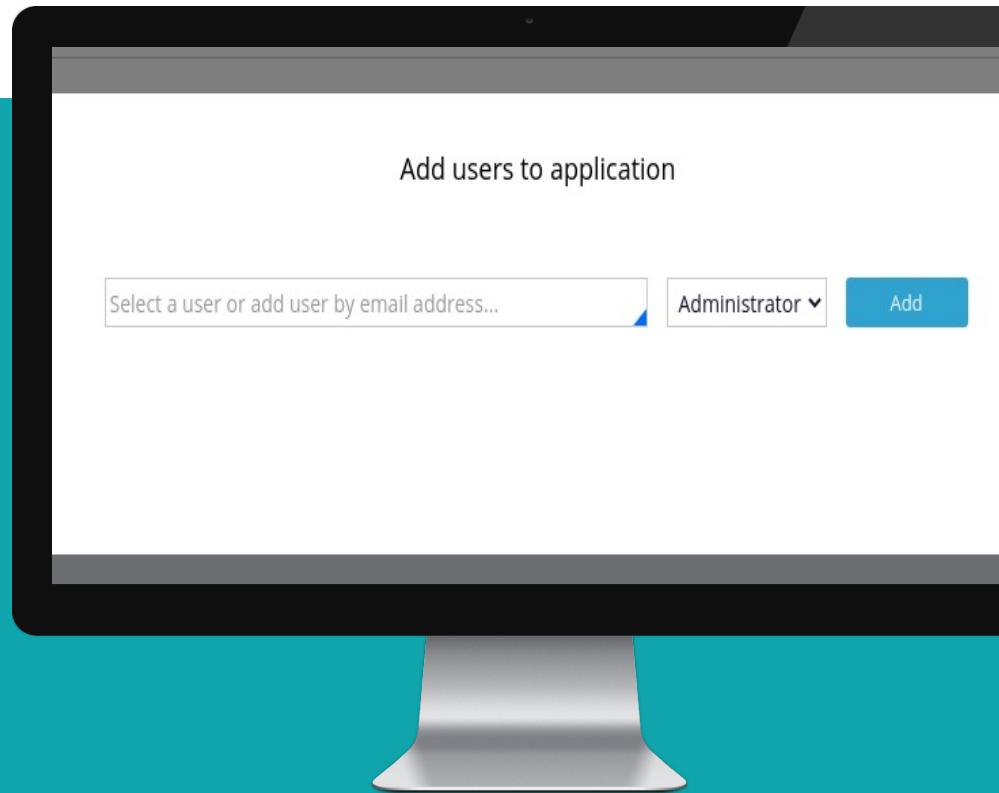
- You can enhance your application by inviting collaborators with different skills and roles
 - For example, invite members of a development team to start working on building your application and begin processing your business cases
- When you invite collaborators, you associate them with personas



Inviting users

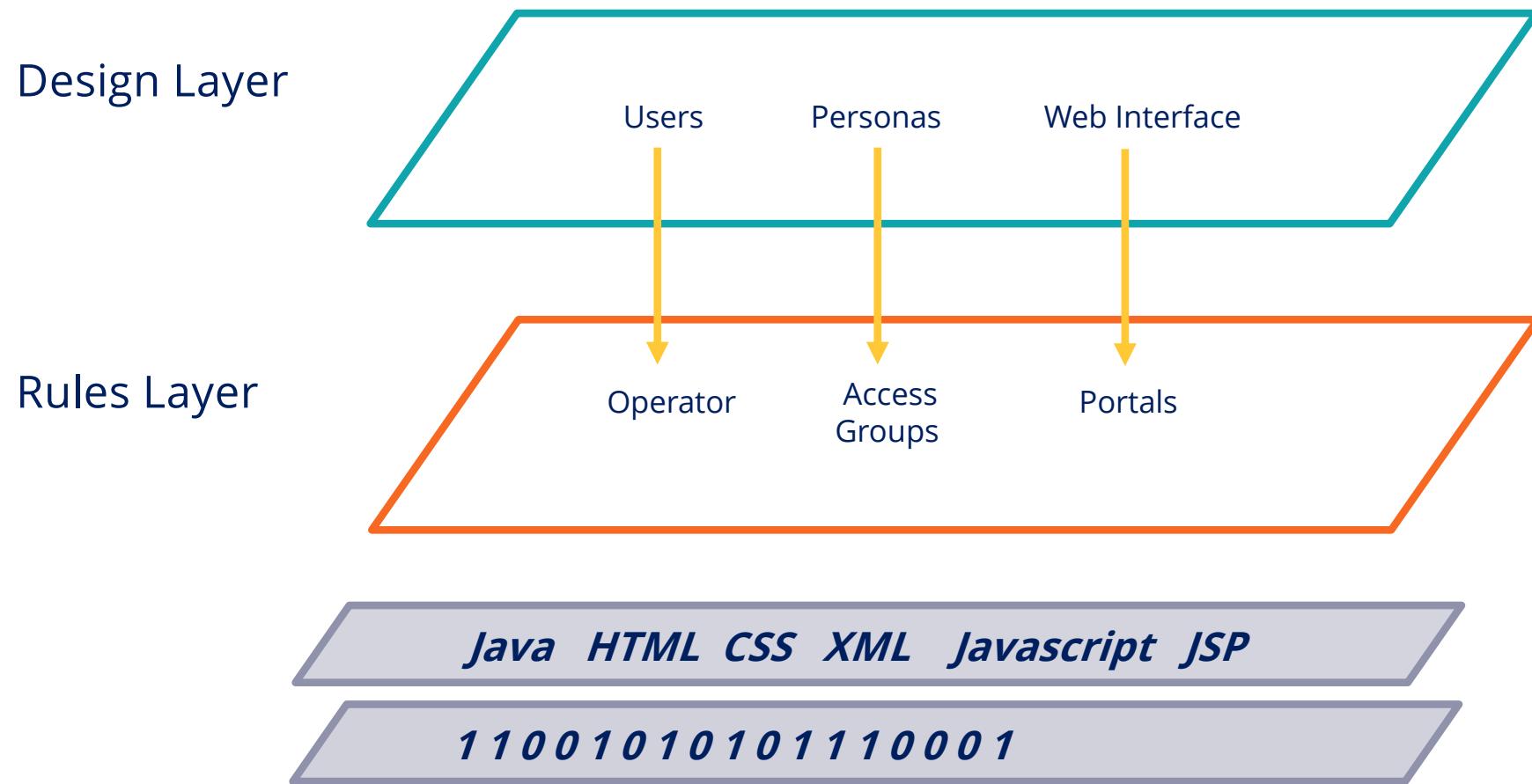
Navigation

- To invite users to the application from App Studio select:
Users > User Management > People > Invite people to your application
- Enter the email address and select a role then click **Add**





Schematic



Skill Mastery

- Describe users, roles, channels, personas
- Inviting a user to the application

