

PEGASYSTEMS PROVIDES THIS SOFTWARE 'AS IS', WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT. IN NO EVENT WILL PEGASYSTEMS, THE AUTHORS, OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES, OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT, OR OTHERWISE, ARISING FROM, OUT OF, OR IN CONNECTION WITH THIS SOFTWARE OR THE USE OR OTHER DEALINGS IN THIS SOFTWARE.

Introduction

The Pega React Starter Kit shows how to construct a React application to utilize the Pega-APIs with layout and field information.

If a business user changes the layout of section, number of fields or type of fields; then, with the information in the Pega-API, the React application should be able to adapt without re-coding.

The React Starter Kit example builds a Case Worker portal which supports *any* simple Pega application.

The CableConnect Sample Application is a simple sample application that can be loaded on to your Pega Infinity 8.1.3 (and higher) system. This sample app exercises many of the Pega-API DX features.

Pega-API

In Pega Infinity, Pega-API's for cases/casetypes/assignments were updated with new extension points to provide layout/field information. This information comes from Harness/Section model information and can be used as hints to allow your React displays to be model driven.

This allows your displays to be linked to the Pega application, such that if the business user changes the design via the Case Designer or App Studio, your display can instantly reflect the changes, without re-coding.

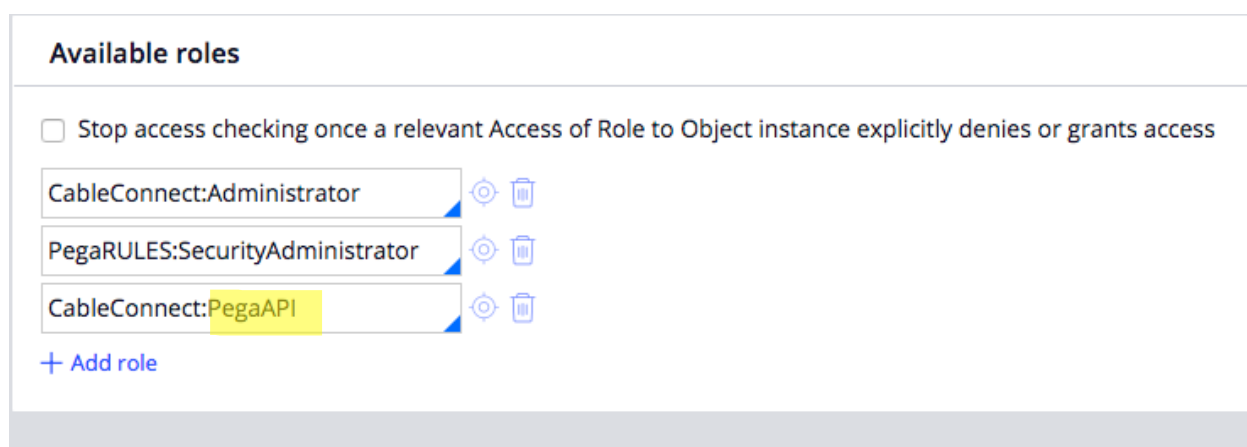
Layout and field information of a section/harness is passed through but organized differently to provide a simplified REST API JSON standard.

You can reference the React Starter Kit Installation Guide for more information.

Cable Connect Sample Application







In order to quickly understand the benefits and usage of the React Starter Kit, we have provided a sample Pega Infinity application called CableConnect which you can install on your Pega Infinity (8.1.3 and higher) system.

Note: If you create a new application or modify this application and create new access group, ensure that your access group contains the PegaAPI role, in order to interface with PegaAPI DX (the Pega Digital Experience APIs).



Available roles

☐ Stop access checking once a relevant Access of Role to Object instance explicitly denies or grants access

CableConnect:Administrator	 
PegaRULES:SecurityAdministrator	 
CableConnect:PegaAPI	 

[+ Add role](#)

Description

Cable Connect is a simple application that consists of 3 users and admin:

- Representative – person contacted by a customer to get new service from a cable provider
- Tech – person who would fulfill the customer request and check off the request as being done
- Manager – person who can add a customer discount, if the representative requests one, and has access to multiple workbaskets.

Please note, that this application is very simple and not designed to be a full use case, but instead, be a simple application that you can use as an example.

Installation

- Download zip
- Login to Pega system
- Import (Configure>Application>Distribution>Import)
- Log off
- Log on as admin.cableco (see below)

Login

There are 4 logins for this application:

- rep.cableco, password: pega - case worker
- tech.cableco, password: pega - case worker
- manager.cableco, password: pega - case manager
- admin.cableco, password: pega - developer/admin

You can use these logins to update/edit/run the application. You might choose to run the application in your browser first, to understand the flow. See “step through” section for more information.

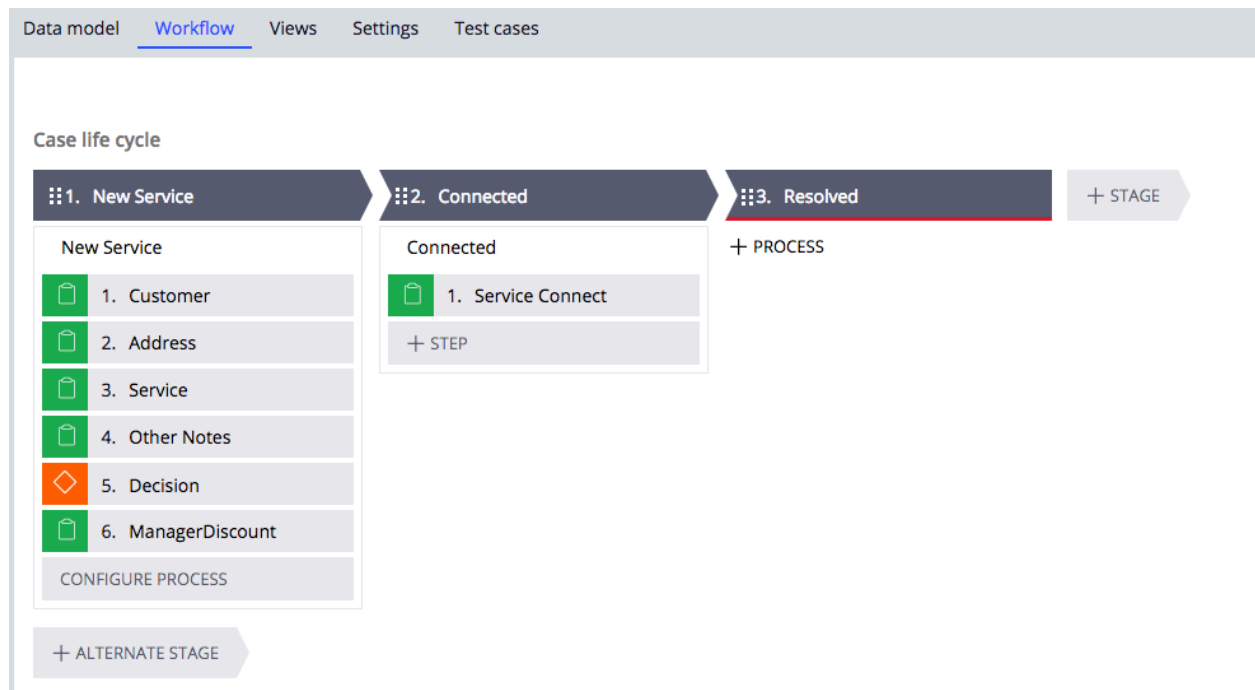
Supporting Functionality

Cable Connect application supports the following functionality, that will be exercised in the React Starter Kit.

- Fields & Controls: pxTextInput, pxTextArea, pxCheckbox, pxRadioButtons, pxDropDown, pxAutoComplete, pxDateTime, pxButton, pxNumber & Disabled controls
- Modes: Editable, ReadOnly
- Layouts: Single Column, 3 Column
- Actions: Refresh with Post, Perform Action

- Validation: Property Edit Validate, Rule Edit Validate, Required
- Routing: Rep->Tech, Rep->Manager->Tech

Here is the Case life cycle.



Note: The majority of this application was created within the Case Designer.

Note: "Service" case type skips the "New" harness:

Behavior

- ☒ Skip 'Create' view when users create a new case
- ☐ Create temporary case that is not saved until a 'Persist case' step is reached ?

If you create an application that does not "skip" the "New" harness, then you WILL need to override the "New" harness in your application, as the out of the box "New" harness currently does NOT work with the PegaAPI DX.

The flow of this application is as follows:

rep.cableco

- gathers customer information, address, requested service and notes
- Rep determines whether to request a discount

- Rep is now finished – routed to Manager (if a discount is needed) or Tech to fulfill

manager.cableco

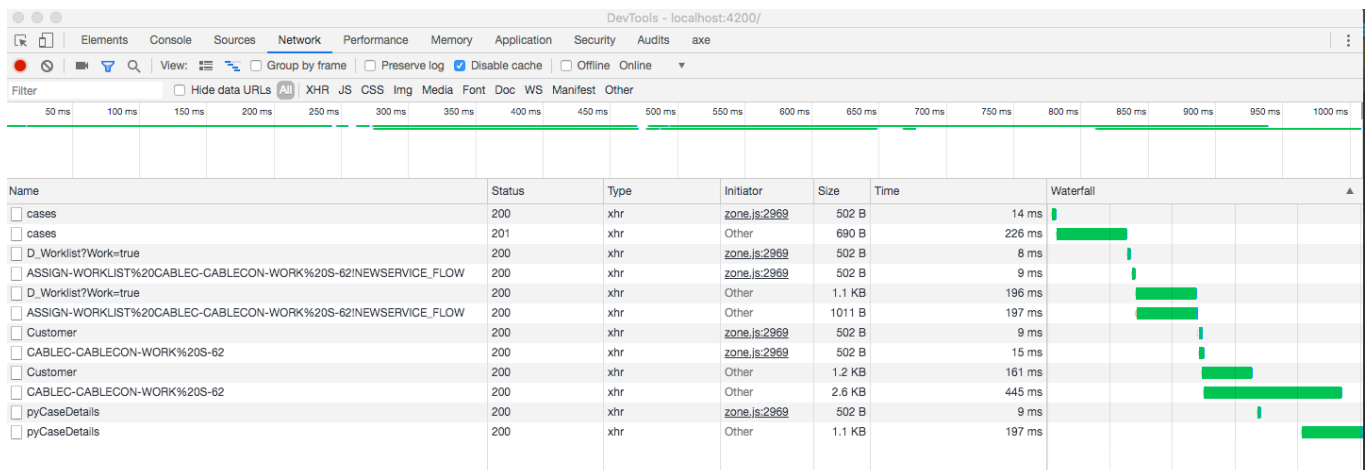
- Manager needs to get work items from manager workbasket
- If any work items request a discount, then manager can enter a discount
- Manager is now finished – routed to Tech to fulfill

tech.cableco

- Works is routed to Tech from either Rep or Manager (if discount)
- Tech is supposed to verify fulfillment of service
- Tech can edit “server request” and change if via “Update service”
- Tech resolves work item

A few points:

- This application is simple, so one can understand the basics.
- Work items get routed, so you can see multiple users and their corresponding work list and/or workbasket.
- Works items for manager get routed to a “manager workbasket” so you can see an example of work sent to a workbasket as well as see an example of how to show lists (display) of workbaskets based upon user’s access group.
- Not all Pega API DX functionality and information is exercised here.
- Utilize the Chrome browser Developer Tools (Network tab) when running the React app so you can trace the network traffic and see the API calls, and responses.



Step Through on the regular desktop

Customer

This is the first screen when logged in as “Rep”. The following is exercised here:

- Text Input
- Required fields
- Autocomplete based upon Prompt List (Suffix)
- Email with Property Edit validate
- Date time with Rule Edit validation on Submit (date can't be in the past)

Service (S-57) NEW

Customer R

First Name *

Middle Name

Last Name *

Suffix

Email *

Service Date

- Rule Edit Validate (ValidateCustomer)

PROPERTY	*Req Conditions	Edit
<input type="text" value="ServiceDate"/>	<p>IF @IsDateInThePast(.ServiceDate)</p> <p>THEN display message:</p> <p>Service date MUST either be TODAY or be in the future</p>	

Address

This is the second screen. The following is exercised here:

- Drop down with data based upon a data page (for State value)
- Phone number with Property edit validation (ValidPhoneNumber is overridden)

Service (S-57) NEW

Address

R

Street

1 Rogers St

City

Cambridge

State

MA ▾

Postal Code

02142

Phone number

6178666000

Cancel

Save

Submit

Service

This is the third screen. The following is exercised here:

- Checkboxes have “Refresh” section. When checked, sub section appears
- Sub sections are visible via a Server side “when rule”
- Radio buttons

Service (S-57) NEW

Service

TV/Cable Service
☒ TV
TV Option
☐ Basic
☒ Basic Plus
☐ Deluxe
☐ Premium

Internet Service
☒ Internet
Internet Option
☐ 25 Mbps
☒ 100 Mbps
☐ 300 Mbps

Home Phone Service
☐ Phone

Cancel

Save

Submit

Notes

This is the fourth screen. The following is exercised here:

- If checkbox is checked, upon submit, work item will be routed to a manager workbasket. Otherwise, will be routed to "Tech" user.

Service (S-57) NEW

Notes R

Other Notes

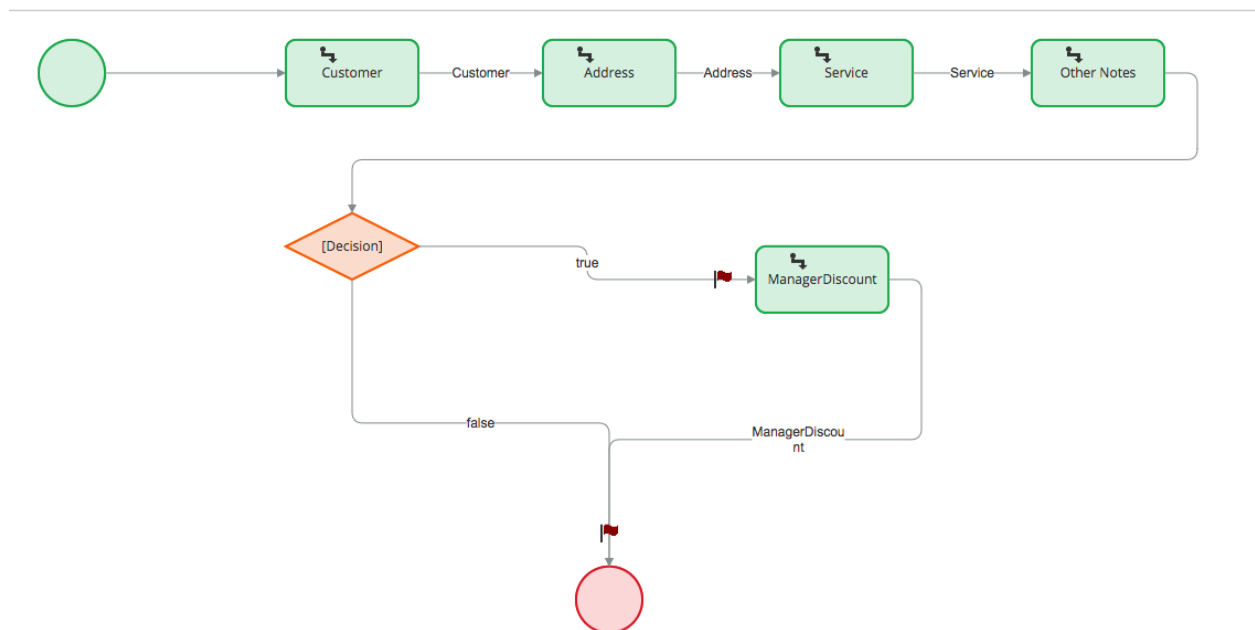
Notes

asking for \$25 off

☒ Send to Manager for Discount

Cancel Save Submit

In the flow, we have a decision based upon the checkbox



Manager Discount

This screen is seen by the manager. Work item will appear in manager worklist. Manager can add a customer discount. The following is exercised here:

- ReadOnly
- Number

Service (S-57) **PENDING MANAGER**

ManagerDiscount

Notes to Manager

asking for \$25 off

Customer Discount

25.00

Cancel

Save

Submit

Service Connect

This screen is seen by the “Tech” user. The “Tech” user is expected to fulfill the request and check off that which has been done. The “Tech” user has the ability to update the “service” (see update service)

The following is exercised here:

- Display Text (formatted text)
- Disabled fields
- Perform Action (call local action to update server) – Update Service

Service (S-57) **PENDING-FULFILLMENT**

Service Connect

Customer
First Name
John
Last Name
Smith
Email
john@smith.com
Phone number
6178666000
Expected Service Date
Friday, February 15, 2019
Discount
\$25.00

Chosen Services
TV Option
☐ Basic
☒ Basic Plus
☐ Deluxe
☐ Premium
Internet Option
☐ 25 Mbps
☒ 100 Mbps
☐ 300 Mbps

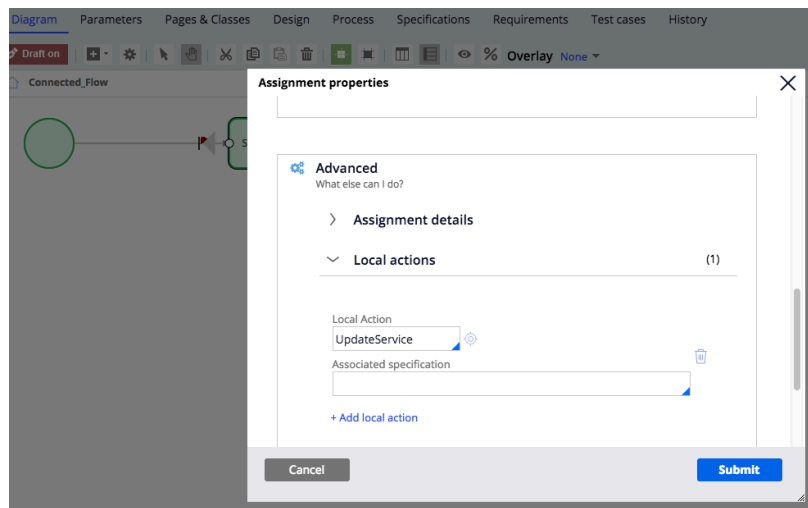
Fulfillment
☒ TV Connected
☒ Internet Connected
☐ Phone Connected

Update Service

This screen is seen if “Update Service” button is pressed in prior screen. This is a local action that can update the “service”. Once the change of service is made, submit will be pressed and the work item will be routed back to the “ServiceConnect” screen.

The following is exercised here:

- Local action on an assignment



Service (S-57) PENDING-FULFILLMENT

UpdateService

TV/Cable Service

☒ TV

TV Option

☐ Basic

☒ Basic Plus

☐ Deluxe

☐ Premium

Internet Service

☒ Internet

Internet Option

☐ 25 Mbps

☒ 100 Mbps

☐ 300 Mbps

Home Phone Service

☐ Phone

Cancel

Save

Submit

Confirm

During the process, the user can see a confirm screen when no further action is required. The Confirm harness has to be overridden, as the default will currently not work with PegaAPI DX.

The following is exercised here:

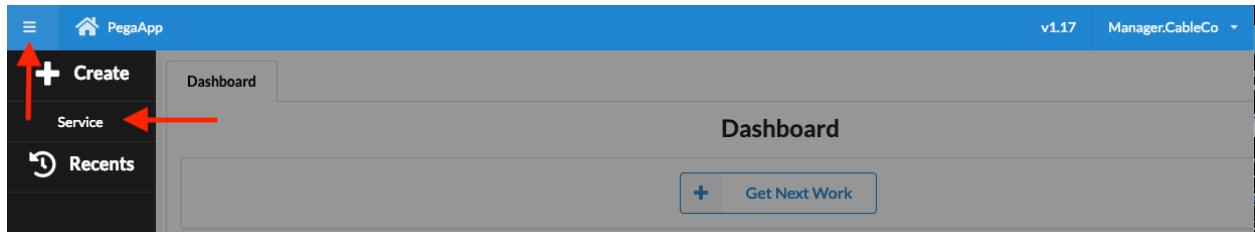
- Overridden Confirm harness with screen layout

Thank you for your submission. No further action is required.

Step Through React Application

After login

Create a new Service work item, select the hamburger icon, then select "Service"



Customer

This is the first screen when logged in as "Rep". The following is exercised here:

- Text Input
- Required fields, error handling
- Autocomplete based upon Prompt List (Suffix)
- Tooltips (React hints)

A screenshot of the 'New Service' form in the PegaApp. The form is divided into two main sections: 'Customer' and 'Case details'. The 'Customer' section contains fields for 'First Name' (with a red asterisk), 'Middle Name', 'Last Name' (with a red asterisk), 'Email' (with a red asterisk), and 'Service Date'. The 'Case details' section contains fields for 'Last updated by' and 'Created by'. The form has a 'Cancel' button and a 'Submit' button. The top navigation bar is blue and contains the PegaApp logo, version v1.17, and the user Manager.CableCo. The sidebar on the left shows the 'Create' button and the 'Service' option.

- Autocomplete via prompt list

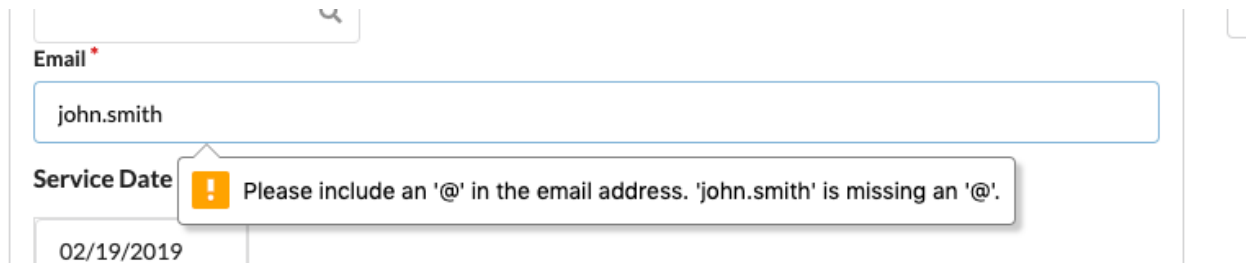
The screenshot shows a PegaApp interface with a blue header bar containing a home icon, the text 'PegaApp', and version information 'v1.17' and 'Manager.CableCo'. Below the header, there's a breadcrumb trail: 'Dashboard > S-13'. The main content area is divided into two panels. The left panel, titled 'Customer', contains form fields for 'First Name' (with a red asterisk), 'Middle Name', 'Last Name' (with a red asterisk), and 'Service Date'. The 'Last Name' field has an autocomplete dropdown showing 'Sr'. The right panel, titled 'Case details', contains fields for 'Last updated by' and 'Created by', both showing 'Manager.CableCo' and '2 minutes ago'. At the bottom of the 'Customer' panel are 'Cancel', 'Save', and 'Submit' buttons.

Error handling

- Required – local error handling in React App

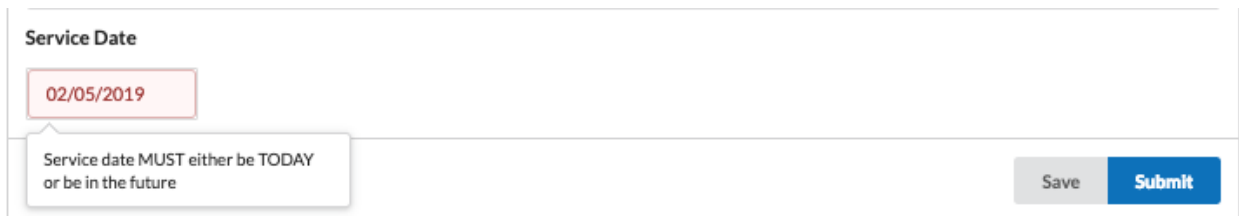
The screenshot shows a PegaApp interface with a blue header bar containing a home icon, the text 'PegaApp', and version information 'v1.17' and 'Manager.CableCo'. Below the header, there's a breadcrumb trail: 'New Service > Connected > Resolved'. The main content area is divided into two panels. The left panel, titled 'Customer', contains form fields for 'First Name' (with a red asterisk), 'Middle Name', and 'Last Name'. The 'First Name' field has a red border and a tooltip message 'Please fill out this field.' is displayed over it. The right panel, titled 'Case details', contains fields for 'Last updated by' and 'Created by', both showing 'Manager.CableCo' and '6 minutes ago'. At the bottom of the 'Customer' panel are 'Cancel', 'Save', and 'Submit' buttons.

- Bad email – both local in React App, and on the server via a submit



A form with two fields. The first field is labeled "Email" with a red asterisk. It contains the text "john.smith". The second field is labeled "Service Date" and contains the date "02/19/2019". A tooltip with an orange exclamation mark icon is displayed over the "Service Date" field, containing the text: "Please include an '@' in the email address. 'john.smith' is missing an '@'."

- Bad Date (not in the future) – on server via submit



A form with a single field labeled "Service Date" containing the date "02/05/2019". The date is highlighted with a red border. A tooltip with a white background and a grey border is displayed over the date, containing the text: "Service date MUST either be TODAY or be in the future". At the bottom right of the form are two buttons: "Save" (grey) and "Submit" (blue).

Address

This is the second screen. The following is exercised here:

- Drop down with data based upon a data page (State)
- Phone number with Property edit validation (default is overridden)

The screenshot shows the 'Address' form in the PegaApp. The form is titled 'Address' and is part of a 'New Service' process. It includes the following fields:

- Street:** 1 Roger St.
- City:** Cambridge
- State:** MA (dropdown menu)
- Postal Code:** 02142
- Phone number:** 6178666000 (with a validation tooltip showing '###) ###-####')

At the bottom of the form are buttons for 'Cancel', 'Save', and 'Submit'. To the right of the form is a 'Case details' section with the following information:

- Last updated by:** Rep.CableCo, a minute ago
- Created by:** Rep.CableCo, 5 minutes ago

- Drop Down via data page

This screenshot shows the 'Address' form with the 'State' dropdown menu open. The dropdown menu displays a list of US states, including AK, AL, AR, AZ, CA, and CO. The 'State' label is visible above the dropdown.

Service

This is the third screen. The following is exercised here:

- Checkboxes have “Refresh” section. When checked, the sub section appears
- Sub sections are visible via a Server side “when rule”, so when you call the “refresh” API, the server will recalculate based upon “when rules” and return data that will have the new section(s).
- Radio buttons

The screenshot shows the 'New Service' screen in the PegaApp. The top navigation bar includes a home icon, 'PegaApp', version 'v1.17', and the user 'Manager.CableCo'. Below the navigation bar, there's a breadcrumb trail: 'Dashboard' > 'S-13' > 'New Service' > 'Connected' > 'Resolved'. The main content area is divided into two sections. The 'Service' section is a form with three columns: 'TV/Cable Service', 'Internet Service', and 'Home Phone Service'. Each column has a checkbox: 'TV', 'Internet', and 'Phone'. Below these checkboxes are 'Cancel', 'Save', and 'Submit' buttons. The 'Case details' section on the right shows 'Last updated by' and 'Created by' as 'Manager.CableCo' with timestamps 'a few seconds ago' and '16 minutes ago' respectively.

- Selecting checkboxes causes refresh and visible sub sections

The screenshot shows the 'New Service' screen after selecting checkboxes. The 'Service' section now displays additional options. Under 'TV/Cable Service', the 'TV' checkbox is checked, and the 'TV Option' section is visible with radio buttons: 'Basic', 'Basic Plus' (selected), 'Deluxe', and 'Premium'. Under 'Internet Service', the 'Internet' checkbox is checked, and the 'Internet Option' section is visible with radio buttons: '25 Mbps', '100 Mbps' (selected), and '300 Mbps'. The 'Home Phone Service' column still shows 'Phone' unchecked. The 'Case details' section remains the same, showing 'Last updated by' and 'Created by' as 'Manager.CableCo' with timestamps 'a few seconds ago' and '16 minutes ago' respectively.

Notes

This is the fourth screen. The following is exercised here:

- If checkbox is checked, upon submit, work item will be routed to a manager workbasket. Otherwise, will be routed to “Tech” user.

Dashboard S-1

New Service > Connected > Resolved Actions

Notes

Other Notes

Notes

Asking for \$25 off

☒ Send to Manager for Discount

Cancel Save Submit

Case details

Last updated by

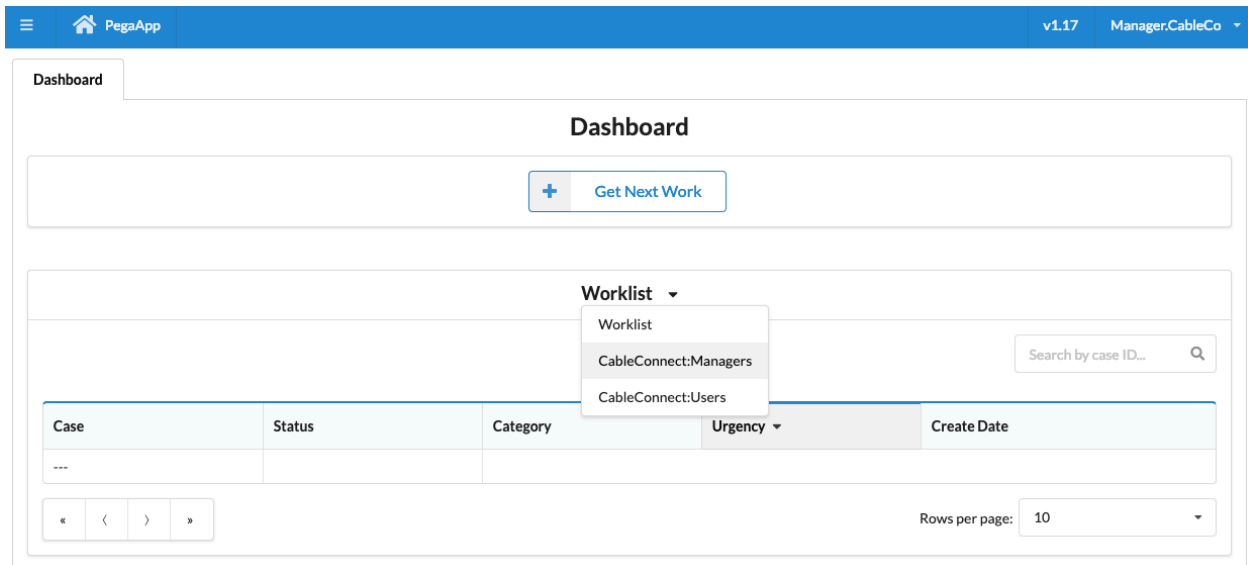
Rep.CableCo
a few seconds ago

Created by

Rep.CableCo
9 minutes ago

Manager Discount

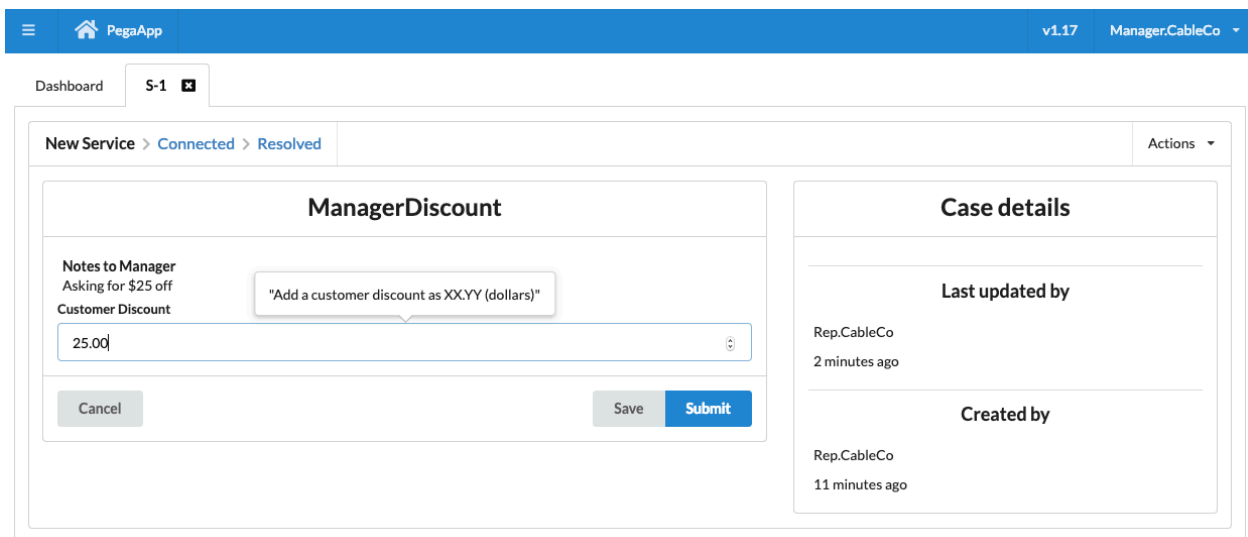
This screen is seen by the manager. Work item will appear in manager worklist.



The screenshot shows the PegaApp Dashboard. At the top, there is a blue header bar with the PegaApp logo, version v1.17, and the user Manager.CableCo. Below the header, the main content area is titled "Dashboard". It features a "Get Next Work" button with a plus icon. A "Worklist" dropdown menu is open, showing options: "Worklist", "CableConnect:Managers", and "CableConnect:Users". Below the dropdown is a table with columns: Case, Status, Category, Urgency, and Create Date. The table is currently empty. At the bottom right, there is a "Rows per page" dropdown set to 10.

Manager can add a customer discount. The following is exercised here:

- ReadOnly field
- Number (only numbers can be entered)



The screenshot shows the PegaApp ManagerDiscount form. The header bar is the same as the previous screenshot. The main content area is titled "ManagerDiscount". It has a breadcrumb trail: "New Service > Connected > Resolved". The form contains a "Notes to Manager" section with the text "Asking for \$25 off Customer Discount". Below this is a text input field with the value "25.00". A tooltip above the input field says "Add a customer discount as XX.YY (dollars)". At the bottom of the form are three buttons: "Cancel", "Save", and "Submit". To the right of the form is a "Case details" section. It contains two subsections: "Last updated by" and "Created by". Both subsections show the user "Rep.CableCo" and the time "2 minutes ago" and "11 minutes ago" respectively.

Service Connect

This screen is seen by the “Tech” user. The “Tech” user is expected to fulfill the request and check off that which has been done. The “Tech” user has the ability to update the “service” (see update service)

The following is exercised here:

- Display Text (formatted text, email, date, currency)
- Disabled fields
- Perform Action (call local action to update server)
- Triple Layout

Service Connect

Customer	Chosen Services	Fulfillment
First Name John	TV Option <input type="radio"/> Basic <input checked="" type="radio"/> Basic Plus <input type="radio"/> Deluxe <input type="radio"/> Premium	<input checked="" type="checkbox"/> TV Connected <input checked="" type="checkbox"/> Internet Connected <input type="checkbox"/> Phone Connected
Last Name Smith	Internet Option <input type="radio"/> 25 Mbps <input checked="" type="radio"/> 100 Mbps <input type="radio"/> 300 Mbps	
Email john.smith@pega.com	Update Service	
Phone number 6178666000		
Expected Service Date Invalid date		
Discount 25.00		

Case details

Last updated by
Manager.CableCo
a minute ago

Created by
Rep.CableCo
13 minutes ago

Cancel Save Submit

Update Service

This screen is seen if “Update Service” is pressed in prior screen. This is a local action that can update the “service”. After submit, the work item will be routed back to the “ServiceConnect” screen.

The following is exercised here:

- Local action on an assignment

The screenshot displays the 'Update Service' interface within the PegaApp. The top navigation bar includes the PegaApp logo, version v1.17, and the user Manager.CableCo. Below the navigation bar, a breadcrumb trail indicates the current path: 'New Service > Connected > Resolved'. The main content area is divided into two primary sections: 'Service' and 'Case details'.

The 'Service' section is further divided into three columns:

- TV/Cable Service:** Includes a checked 'TV' option and four radio button options for 'TV Option': Basic, Basic Plus (selected), Deluxe, and Premium.
- Internet Service:** Includes a checked 'Internet' option and three radio button options for 'Internet Option': 25 Mbps, 100 Mbps (selected), and 300 Mbps.
- Home Phone Service:** Includes an unchecked 'Phone' option.

At the bottom of the 'Service' section, there are three buttons: 'Cancel', 'Save', and 'Submit'.

The 'Case details' section on the right contains two fields:

- Last updated by:** Manager.CableCo, a few seconds ago.
- Created by:** Manager.CableCo, 16 minutes ago.

Confirm

During the process, the user can see a confirm screen when no further action is required. The Confirm harness has to be overridden, as the default will currently not work with PegaAPI DX (Digital Experience APIs).

The following is exercised here:

- Overridden Confirm harness with screen layout

The screenshot displays the PegaApp interface. At the top, a blue header bar contains a menu icon, the text 'PegaApp', the version 'v1.17', and the user 'Tech.CableCo'. Below the header, a breadcrumb trail reads 'New Service > Connected > Resolved'. The main content area is divided into two panels. The left panel, titled 'Confirm', shows the status 'Resolved-Completed' and a message: 'Thank you for your submission. No further action is required.' with a 'Close' button. The right panel, titled 'Case details', shows 'Last updated by' as 'Tech.CableCo' 'a few seconds ago' and 'Created by' as 'Rep.CableCo' '16 minutes ago'.

Confirm	
Status: Resolved-Completed	
Status Resolved-Completed	
Thank you for your submission. No further action is required.	
<button>Close</button>	

Case details	
Last updated by	
Tech.CableCo a few seconds ago	
Created by	
Rep.CableCo 16 minutes ago	