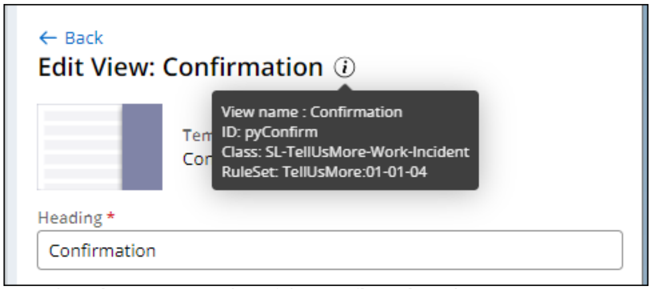
Troubleshooting Constellation

Wednesday, July 3, 2024 10:32 AM

Access View JSON data in Dev Studio

1. In the Pega instance for the challenge, enter the following credentials:
   a. In the User name field, enter Author@SL.
   b. In the Password field, enter pega123!
2. In the navigation pane of **App Studio**, click **Case Types > Incident** to open the Incident Case Type.
3. On the Case Types **UX** tab, navigate to the **Confirmation** partial View.
   Note: Pega provides several out-of-the-box Views that can be used in developing Case Types. Some of these Views have unusual configurations. An example of this is the **Confirmation** View (ID: *pyConfirm*). This View is responsible for displaying information in the Assignment confirmation.
4. Hover over the **Information** icon to see the View's metadata.



← Back

Edit View: Confirmation ⓘ

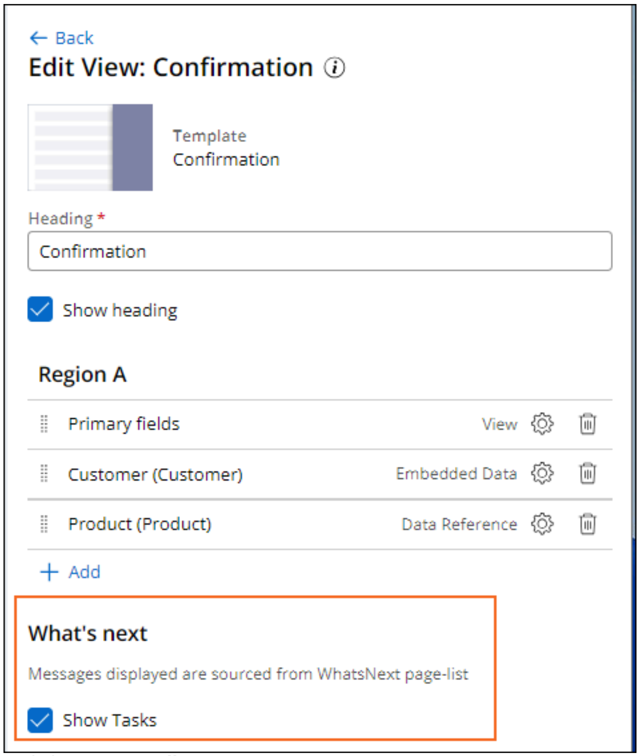
View name : Confirmation
ID: pyConfirm
Class: SL-TellUsMore-Work-Incident
RuleSet: TellUsMore:01-01-04

Tem
Cor

Heading *

Confirmation

5. Note the **What's next** section of the **Confirmation** View.



← Back

Edit View: Confirmation ⓘ

Template
Confirmation

Heading *

Confirmation

☑ Show heading

Region A

Primary fields — View

Customer (Customer) — Embedded Data

Product (Product) — Data Reference

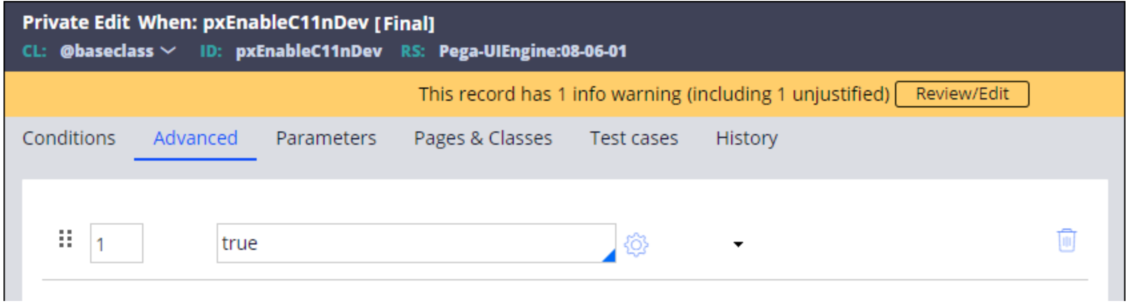
+ Add

What's next

Messages displayed are sourced from WhatsNext page-list

☑ Show Tasks

6. Switch to **Dev Studio**.
7. Search for the Rule named *pxEnableC11nDev*. If the Rule cannot be found in the search menu of Dev Studio, click **Records > Decision > When** and filter the **Name** column by *pxEnableC11nDev* to locate the Rule.
8. Open the *pxEnableC11nDev* Rule, continue with a **Private edit**, and then set the value equal to **true**.



Private Edit When: pxEnableC11nDev [Final]

CL: @baseclass ID: pxEnableC11nDev RS: Pega-UIEngine:08-06-01

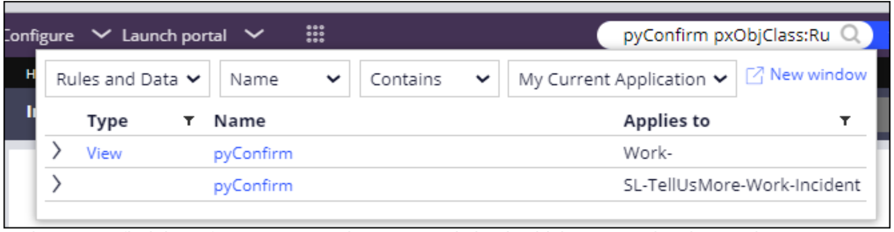
This record has 1 info warning (including 1 unjustified) Review/Edit

Conditions | Advanced | Parameters | Pages & Classes | Test cases | History

1 true

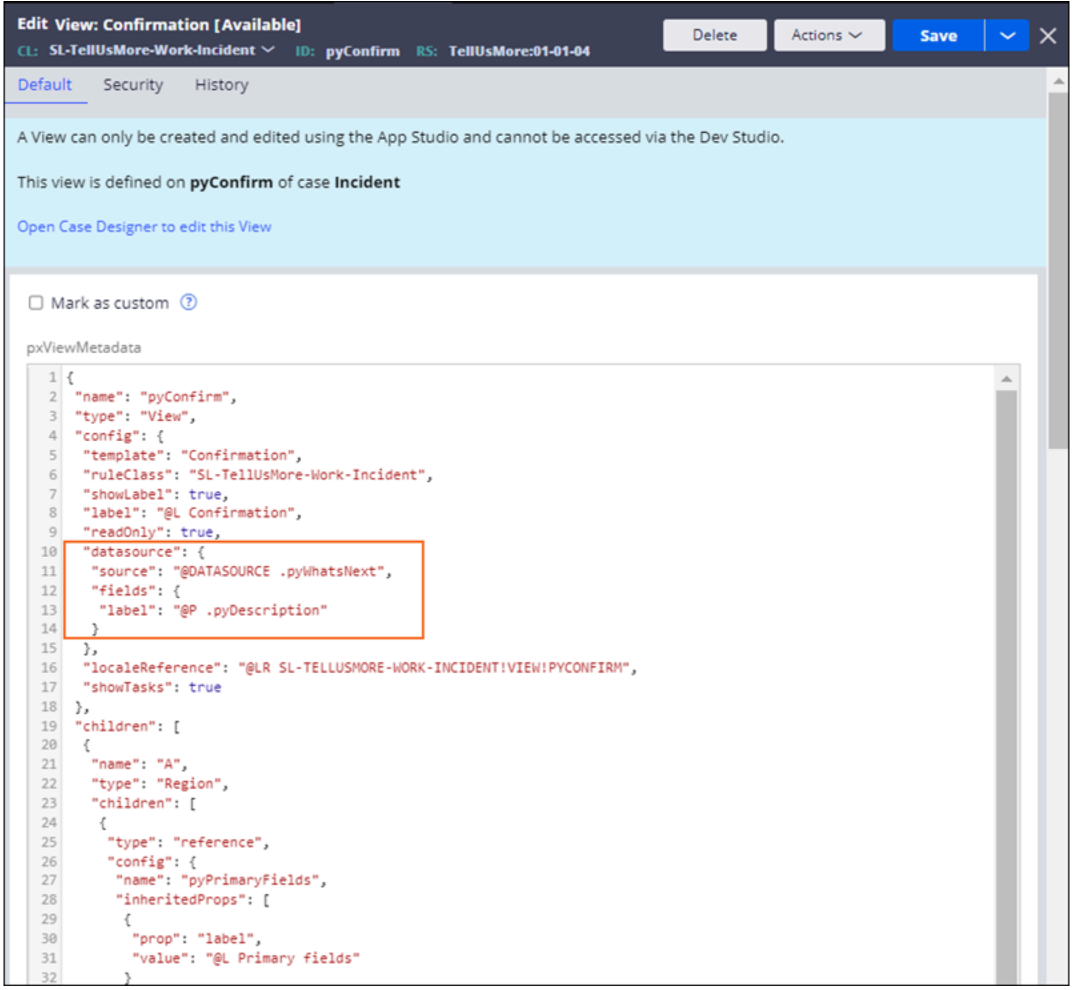
Add condition

9. In the Search bar of Dev Studio, enter pyConfirm pxObjClass:Rule-UI-View to open the *pyConfirm* View.
10. Select the *pyConfirm* View Rule associated with the **SL-TellUsMore-Work-Incident** class.



Type	Name	Applies to
View	pyConfirm	Work-
	pyConfirm	SL-TellUsMore-Work-Incident

11. In the JSON, find the reference to *.pyWhatsNext* and check which property is taken and displayed on the UI.



Edit View: Confirmation [Available]

CL: SL-TellUsMore-Work-Incident ID: pyConfirm RS: TellUsMore:01-01-04

A View can only be created and edited using the App Studio and cannot be accessed via the Dev Studio.

This view is defined on **pyConfirm** of case **Incident**

Open Case Designer to edit this View

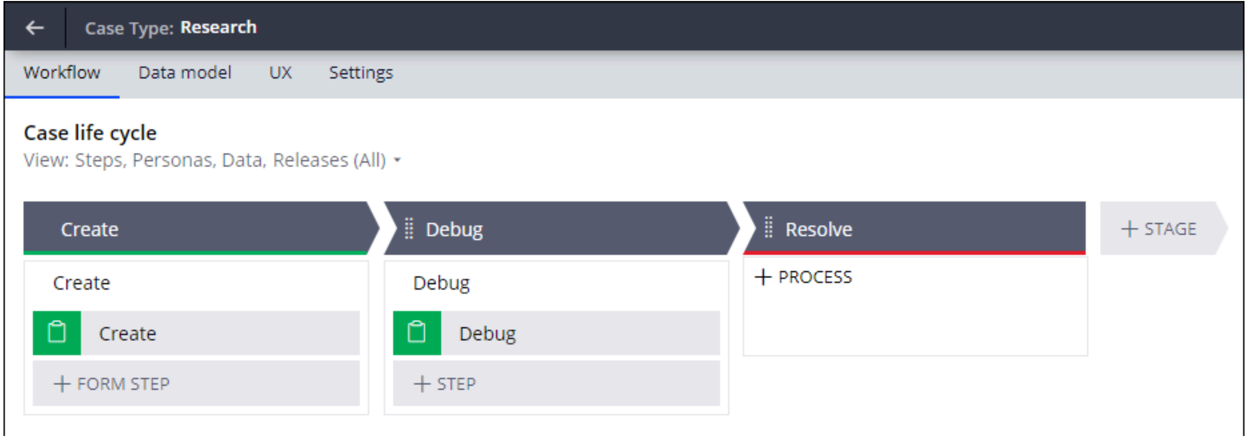
□ Mark as custom

pxViewMetadata

```
{
 "name": "pyConfirm",
 "type": "View",
 "config": {
  "template": "Confirmation",
  "ruleClass": "SL-TellUsMore-Work-Incident",
  "showLabel": true,
  "label": "@L Confirmation",
  "readOnly": true,
  "datasource": {
   "source": "@DATASOURCE .pyWhatsNext",
   "fields": {
    "label": "@P .pyDescription"
   }
  },
  "localeReference": "@LR SL-TELLUSMORE-WORK-INCIDENT!VIEW!PYCONFIRM",
  "showTasks": true
 },
 "children": [
  {
   "name": "A",
   "type": "Region",
   "children": [
    {
     "type": "reference",
     "config": {
      "name": "pyPrimaryFields",
      "inheritedProps": [
       {
        "prop": "label",
        "value": "@L Primary fields"
       }
```

2Create a Research Case Type

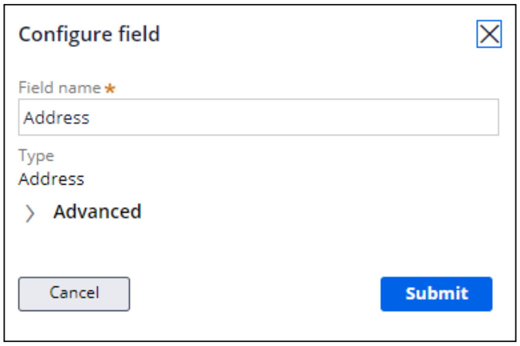
1. Switch to **App Studio**, and in the navigation pane, click **Case Types**.
2. Create a new Case Type named Research.
3. The Research Case Type should have two additional Stages: Debug and Resolve.
4. In the Resolve Stage contextual properties pane, select **Resolve the Case**.
5. In the Debug Stage, create a **Collect information** Step labeled Debug.



6. Click **Save**.

3Examine View Rules using the Tracer

1. In the Research Case Type, open the Case Type Data Model.
2. Add a new **Address** field named Address.



Configure field

Field name *

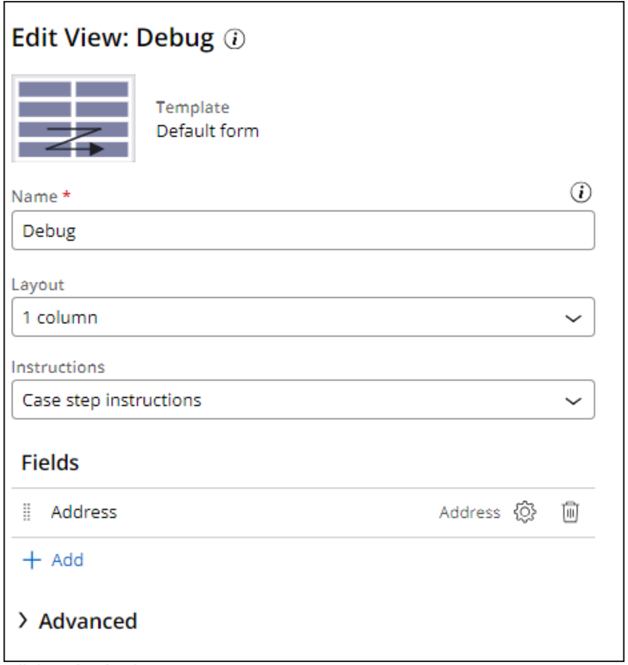
Address

Type

Address

Note: By default, the **Address** Field Type has a default View named *pyAddressForm*.

3. From the Debug Step, open the **Debug** View, and add the **Address** field to it.



Edit View: Debug

Template

Default form

Name *

Debug

Layout

1 column

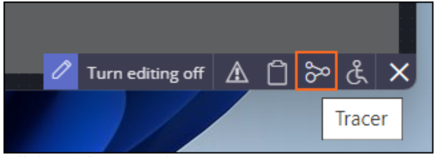
Instructions

Case step instructions

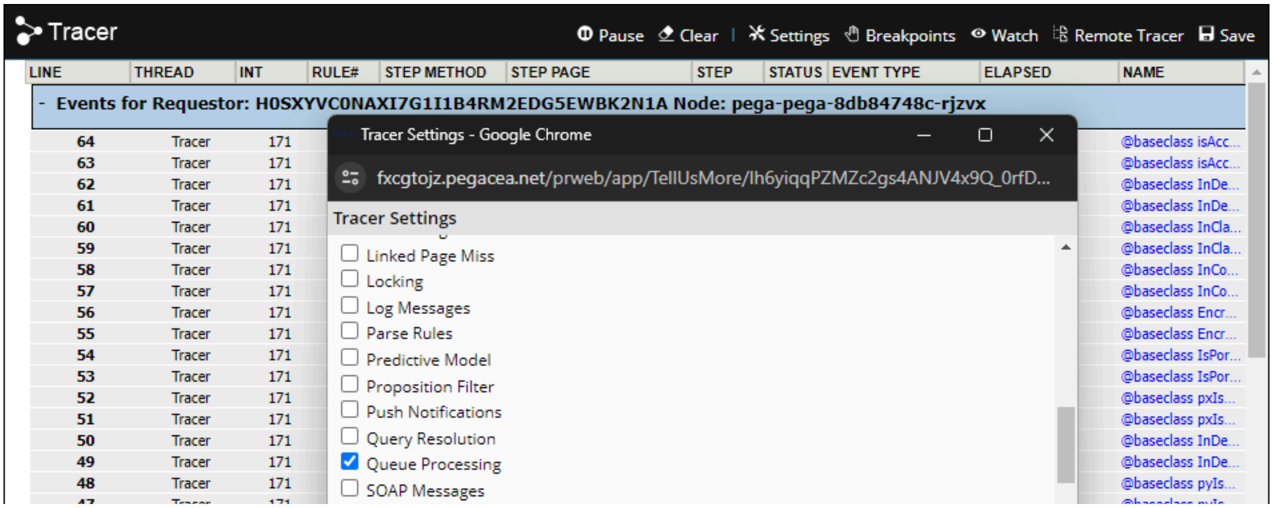
Fields

Address   Address

4. Click **Submit**, then **Save**.
5. Click the **Preview** button to open the Portal window.
6. Create a Research Case Type. In the **Create-Research** View, click **Submit**.
7. In the bottom-right corner of App Studio, open the **Tracer**.



8. Click **Settings**. In the **EVENT TYPES TO TRACE** section, scroll down and enable the **View Rules**. Then click **OK**.



- Events for Requestor: H0SXYVC0NAXI7G1I1B4RM2EDG5EWBK2N1A Node: pega-pega-8db84748c-rjzvx

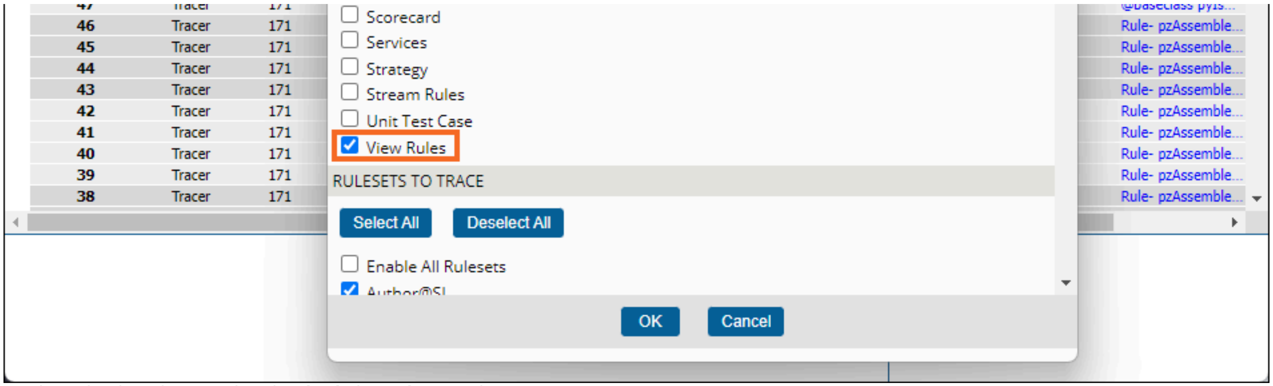
LINE	THREAD	INT	RULE#	STEP METHOD	STEP PAGE	STEP	STATUS	EVENT TYPE	ELAPSED	NAME
64	Tracer	171								@baseclass isAcc...
63	Tracer	171								@baseclass isAcc...
62	Tracer	171								@baseclass InDe...
61	Tracer	171								@baseclass InDe...
60	Tracer	171								@baseclass InCla...
59	Tracer	171								@baseclass InCla...
58	Tracer	171								@baseclass InCo...
57	Tracer	171								@baseclass InCo...
56	Tracer	171								@baseclass Encr...
55	Tracer	171								@baseclass Encr...
54	Tracer	171								@baseclass IsPor...
53	Tracer	171								@baseclass IsPor...
52	Tracer	171								@baseclass pxIs...
51	Tracer	171								@baseclass pxIs...
50	Tracer	171								@baseclass InDe...
49	Tracer	171								@baseclass InDe...
48	Tracer	171								@baseclass pyIs...

Tracer Settings - Google Chrome

fxcgtojz.pegacea.net/prweb/app/TellUsMore/Ih6yiqqPZMZc2gs4ANJV4x9Q_0rfD...

Tracer Settings

☐ Linked Page Miss
☐ Locking
☐ Log Messages
☐ Parse Rules
☐ Predictive Model
☐ Proposition Filter
☐ Push Notifications
☐ Query Resolution
☑ Queue Processing
☐ SOAP Messages



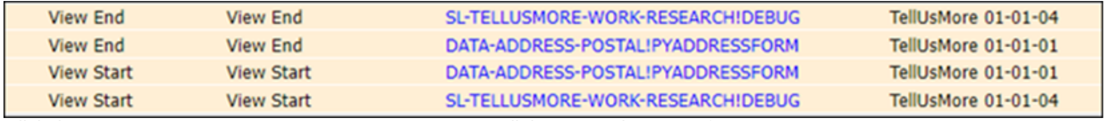
47	Tracer	171	@baseclass pyIs...
46	Tracer	171	Rule- pzAssemble...
45	Tracer	171	Rule- pzAssemble...
44	Tracer	171	Rule- pzAssemble...
43	Tracer	171	Rule- pzAssemble...
42	Tracer	171	Rule- pzAssemble...
41	Tracer	171	Rule- pzAssemble...
40	Tracer	171	Rule- pzAssemble...
39	Tracer	171	Rule- pzAssemble...
38	Tracer	171	Rule- pzAssemble...

□ Scorecard
□ Services
□ Strategy
□ Stream Rules
□ Unit Test Case
☑ View Rules

RULESETS TO TRACE

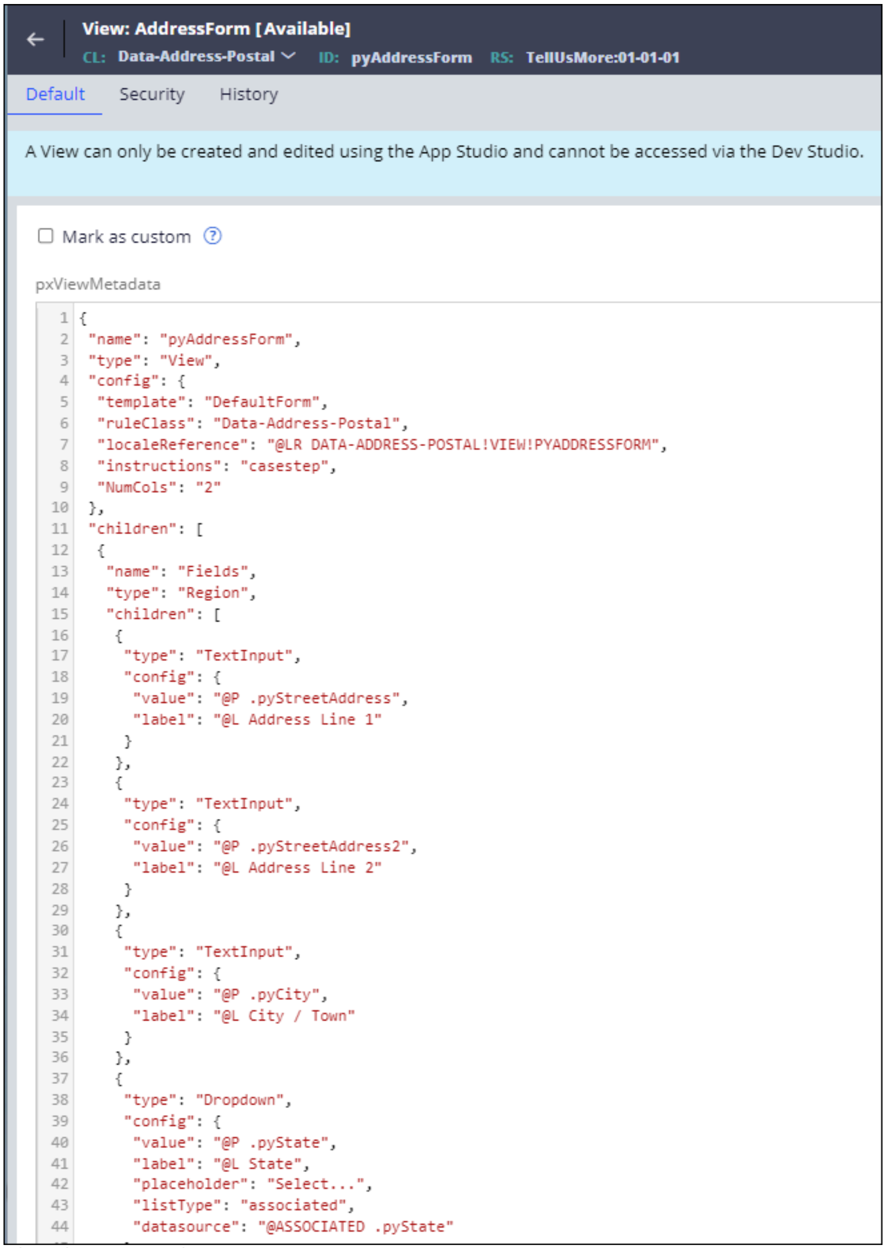
□ Enable All Rulesets
☑ Author@SI

9. Navigate back to the Portal and **Submit** the **Debug** Assignment.
10. Navigate back to the Tracer, expand **Resource: /application/v2/assignments** and search for View Start to locate the name of all the Views requested for rendering.
11. Find the Views associated with the **pyAddressForm**.



View End	View End	SL-TELLUSMORE-WORK-RESEARCH!DEBUG	TellUsMore 01-01-04
View End	View End	DATA-ADDRESS-POSTAL!PYADDRESSFORM	TellUsMore 01-01-01
View Start	View Start	DATA-ADDRESS-POSTAL!PYADDRESSFORM	TellUsMore 01-01-01
View Start	View Start	SL-TELLUSMORE-WORK-RESEARCH!DEBUG	TellUsMore 01-01-04

12. Click the DATA-ADDRESS-POSTAL!PYADDRESSFORM link to open the JSON for the **AddressForm** View.



View: AddressForm [Available]

CL: Data-Address-Postal ID: pyAddressForm RS: TellUsMore:01-01-01

A View can only be created and edited using the App Studio and cannot be accessed via the Dev Studio.

□ Mark as custom

pxViewMetadata

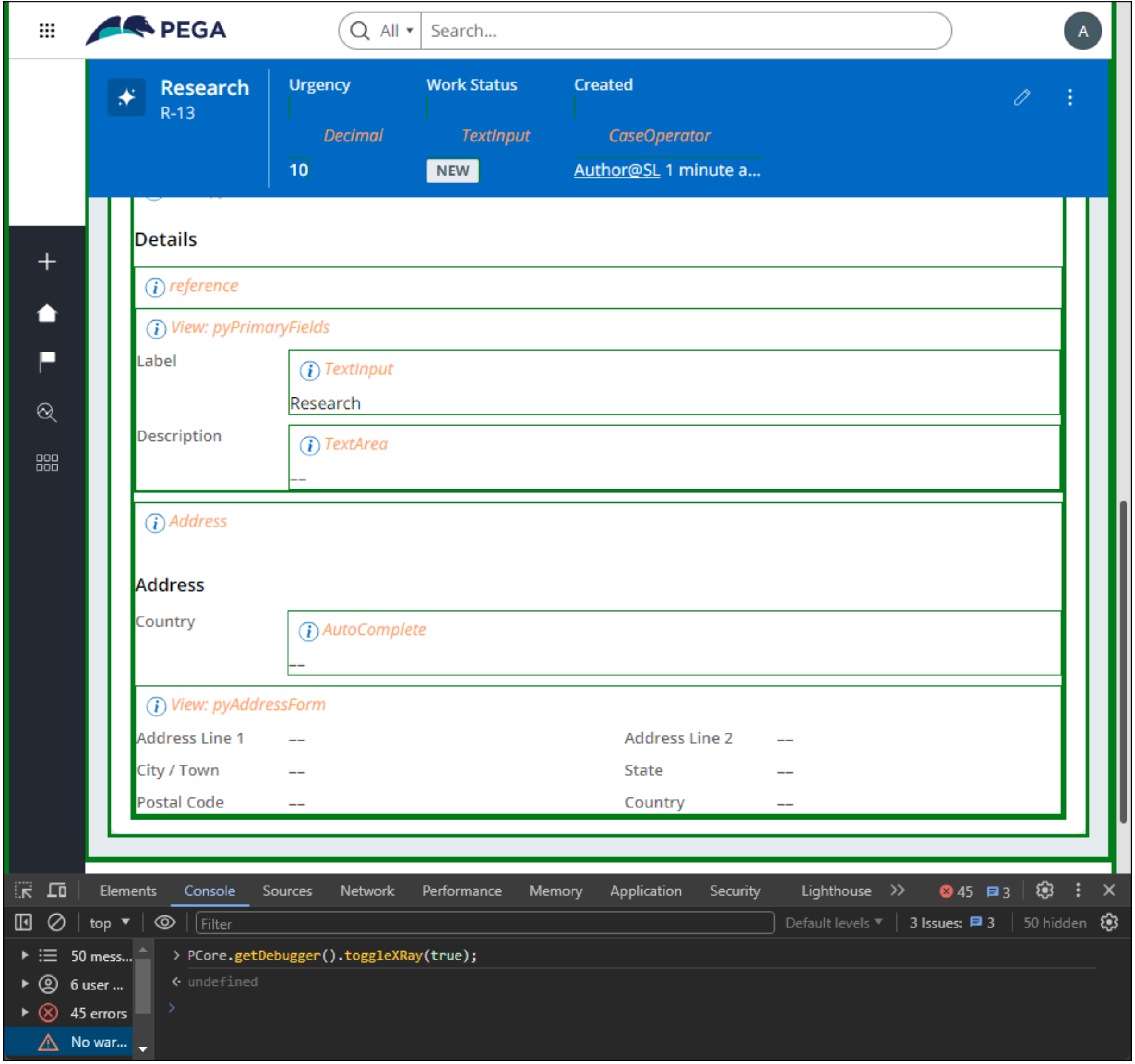
```
{
 "name": "pyAddressForm",
 "type": "View",
 "config": {
  "template": "DefaultForm",
  "ruleClass": "Data-Address-Postal",
  "localeReference": "@LR DATA-ADDRESS-POSTAL!VIEW!PYADDRESSFORM",
  "instructions": "casestep",
  "NumCols": "2"
 },
 "children": [
  {
   "name": "Fields",
   "type": "Region",
   "children": [
    {
     "type": "TextInput",
     "config": {
      "value": "@P .pyStreetAddress",
      "label": "@L Address Line 1"
     }
    },
    {
     "type": "TextInput",
     "config": {
      "value": "@P .pyStreetAddress2",
      "label": "@L Address Line 2"
     }
    },
    {
     "type": "TextInput",
     "config": {
      "value": "@P .pyCity",
      "label": "@L City / Town"
     }
    },
    {
     "type": "Dropdown",
     "config": {
      "value": "@P .pyState",
      "label": "@L State",
      "placeholder": "Select...",
      "listType": "associated",
      "datasource": "@ASSOCIATED .pyState"
```

13. Close the Tracer tool.

4Explore View metadata with X-Ray

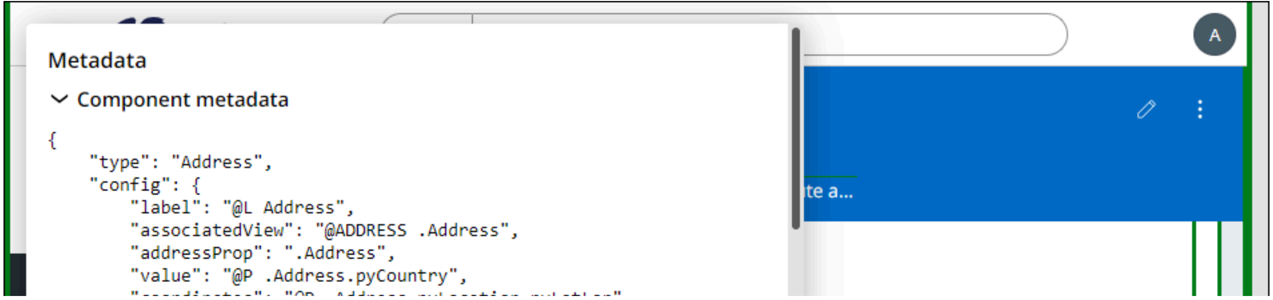
Note: The X-Ray tool provides an intuitive way to debug screen issues, such as metadata, fields metadata, and state properties. X-Ray is primarily used for debugging Views and other UI controls, rather than Assignments.

1. In **App Studio**, open the Research Case Type.
2. Click the **UX** tab, and open the **Details** partial View.
3. Delete the **Primary fields** View and add the **Address** field to the **Details** partial View.
4. Switch to **Dev Studio**, and **Launch** the **Customer Portal**.
5. Create a new Research Case.
6. Navigate to the **Debug** Assignment.
7. Open the **Dev Tools** for your browser - likely using **F12** (for Chrome, Fire Fox, Edge)
8. Navigate to the **Console** tab and clear the window.
9. In the **Dev Tools Console**, enter: PCore.getDebugger().toggleXRay(true); to start the X-Ray tool.
   Note: If you see the following message: PCore is not defined in the Console, it is probable that you launched the Portal from App Studio. The PCore and PConnect methods are not available in the App Studio shell. For PCore and PConnect to work, you must launch the Portal from Dev Studio.
10. Navigate to the **Details**, tab and then find the partial View that is related to the **Address** property.



```
> PCore.getDebugger().toggleXRay(true);
< undefined
```

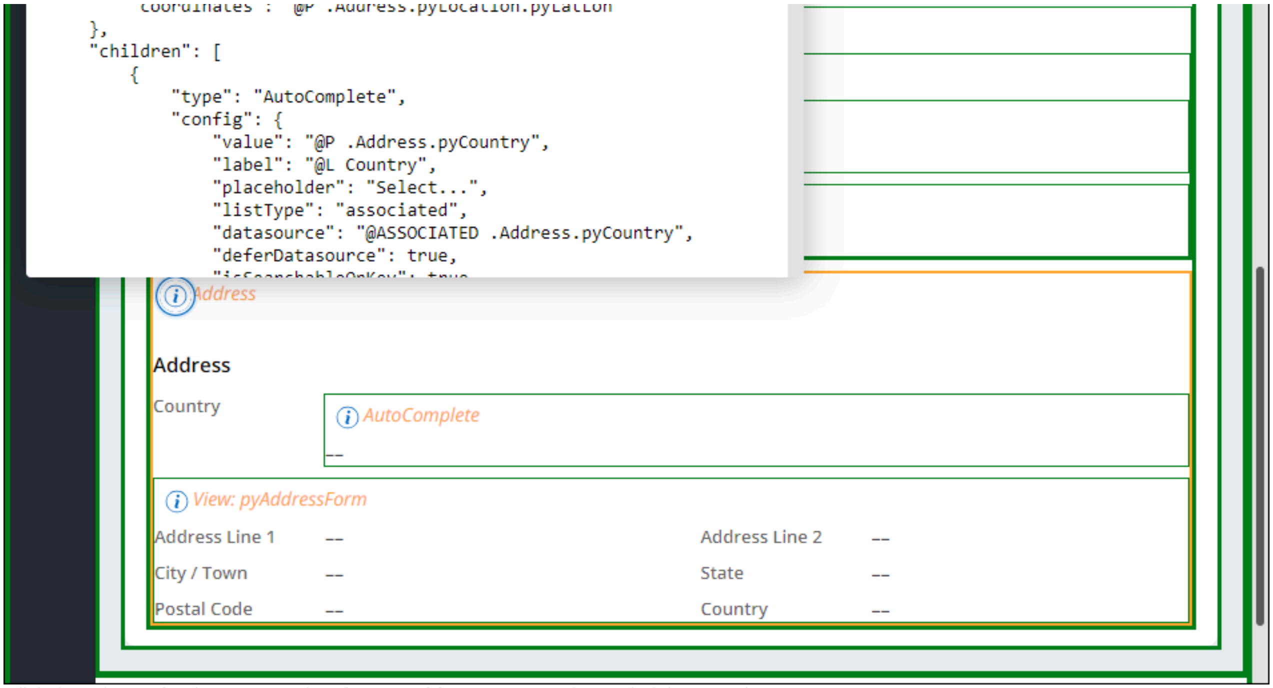
11. Click the **Information** icon next to the **Address** section to find the metadata related to the Address property.



Metadata

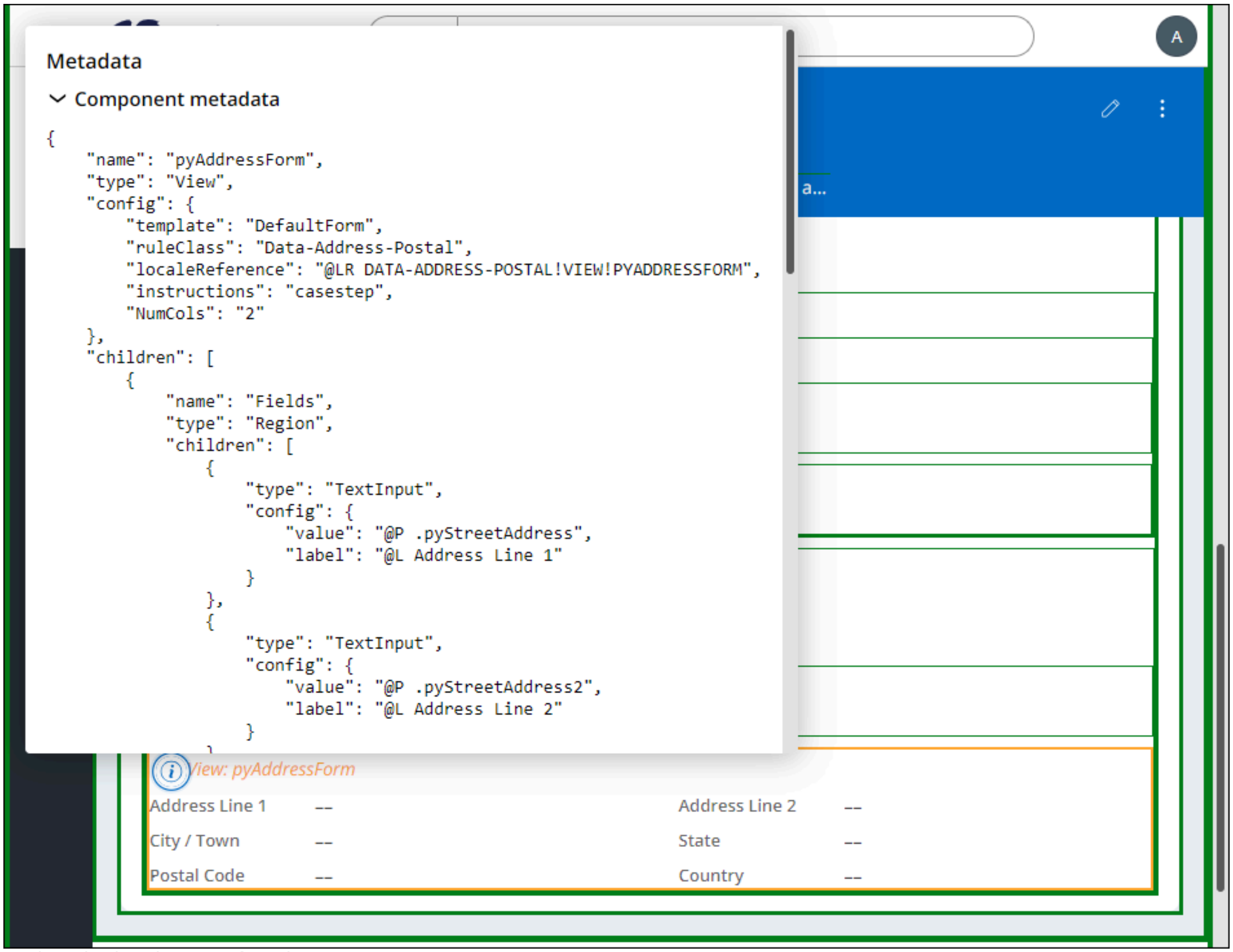
Component metadata

```
{
    "type": "Address",
    "config": {
        "label": "@L Address",
        "associatedView": "@ADDRESS .Address",
        "addressProp": ".Address",
        "value": "@P .Address.pyCountry",
        "coordinates": "@P .Address.pyLocation.pyLatLon"
```



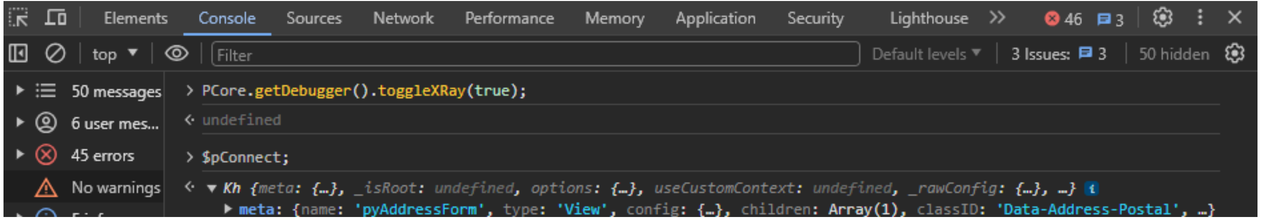
```
            "coordinates": "@P .Address.pyLocation.pyLatLon"
        },
        "children": [
            {
                "type": "AutoComplete",
                "config": {
                    "value": "@P .Address.pyCountry",
                    "label": "@L Country",
                    "placeholder": "Select...",
                    "listType": "associated",
                    "datasource": "@ASSOCIATED .Address.pyCountry",
                    "deferDatasource": true,
                    "isSearchableOnKey": true
```

12. Click the **Information** icon next to the **View:pyAddressForm** section to find the metadata related to the View.

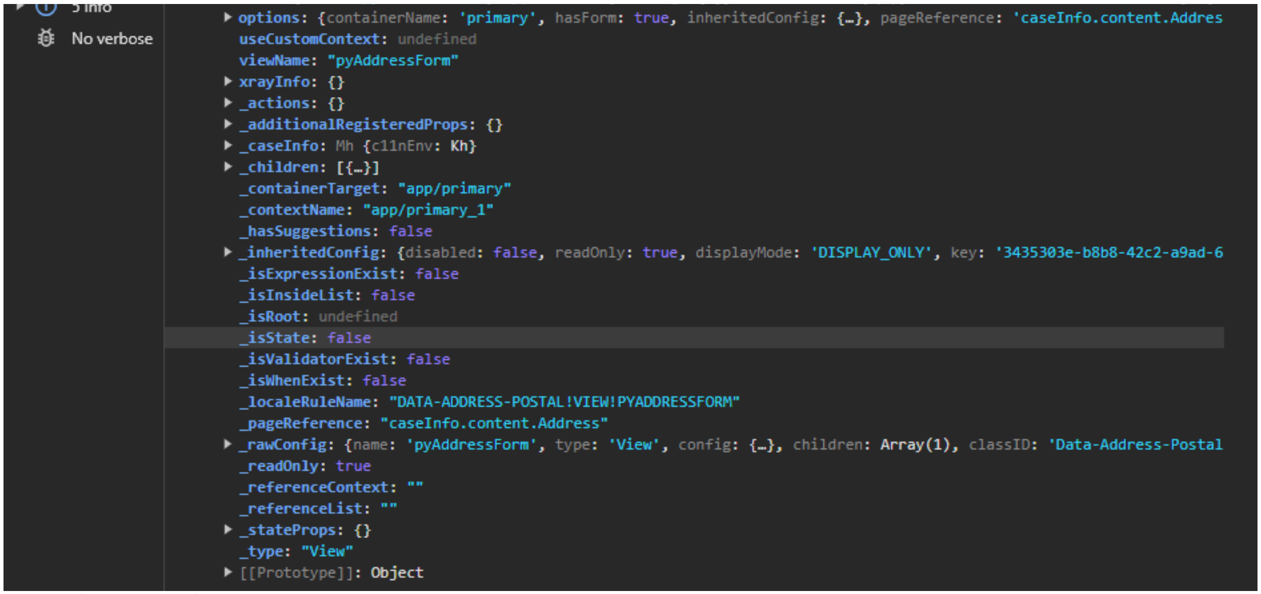


```
{
    "name": "pyAddressForm",
    "type": "View",
    "config": {
        "template": "DefaultForm",
        "ruleClass": "Data-Address-Postal",
        "localeReference": "@LR DATA-ADDRESS-POSTAL!VIEW!PYADDRESSFORM",
        "instructions": "casestep",
        "NumCols": "2"
    },
    "children": [
        {
            "name": "Fields",
            "type": "Region",
            "children": [
                {
                    "type": "TextInput",
                    "config": {
                        "value": "@P .pyStreetAddress",
                        "label": "@L Address Line 1"
                    }
                },
                {
                    "type": "TextInput",
                    "config": {
                        "value": "@P .pyStreetAddress2",
                        "label": "@L Address Line 2"
                    }
                },
```

13. Navigate back to the **Console** and enter $pConnect;
14. Explore the information provided in the Console related to the *pyAddressForm* View.



```
> PCore.getDebugger().toggleXRay(true);
< undefined
> $pConnect;
< ▼ Kh {meta: {…}, _isRoot: undefined, options: {…}, useCustomContext: undefined, _rawConfig: {…}, …}
   ▶ meta: {name: 'pyAddressForm', type: 'View', config: {…}, children: Array(1), classID: 'Data-Address-Postal', …}
```

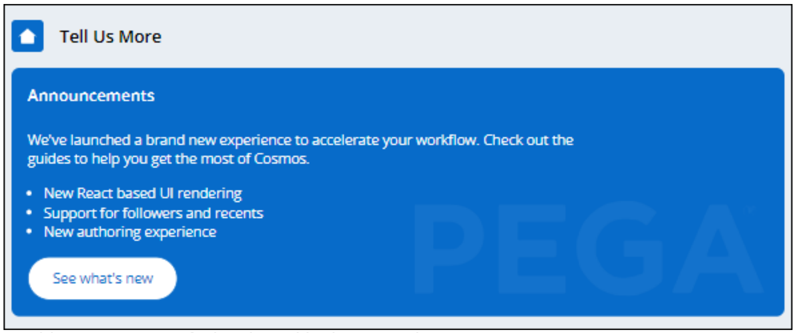


```
▶ options: {containerName: 'primary', hasForm: true, inheritedConfig: {…}, pageReference: 'caseInfo.content.Addres
  useCustomContext: undefined
  viewName: "pyAddressForm"
▶ xrayInfo: {}
▶ _actions: {}
▶ _additionalRegisteredProps: {}
▶ _caseInfo: Mh {c11nEnv: Kh}
▶ _children: [{…}]
  _containerTarget: "app/primary"
  _contextName: "app/primary_1"
  _hasSuggestions: false
▶ _inheritedConfig: {disabled: false, readOnly: true, displayMode: 'DISPLAY_ONLY', key: '3435303e-b8b8-42c2-a9ad-6
  _isExpressionExist: false
  _isInsideList: false
  _isRoot: undefined
  _isState: false
  _isValidatorExist: false
  _isWhenExist: false
  _localeRuleName: "DATA-ADDRESS-POSTAL!VIEW!PYADDRESSFORM"
  _pageReference: "caseInfo.content.Address"
▶ _rawConfig: {name: 'pyAddressForm', type: 'View', config: {…}, children: Array(1), classID: 'Data-Address-Postal
  _readOnly: true
  _referenceContext: ""
  _referenceList: ""
▶ _stateProps: {}
  _type: "View"
▶ [[Prototype]]: Object
```

15. To stop the X-Ray tool, enter PCore.getDebugger().toggleXRay(false);

5Examine the App Announcement Section of the Portal Home Page

1. In the Customer Portal, navigate to the Portal Home page.
2. Note the contents in the Announcements banner.

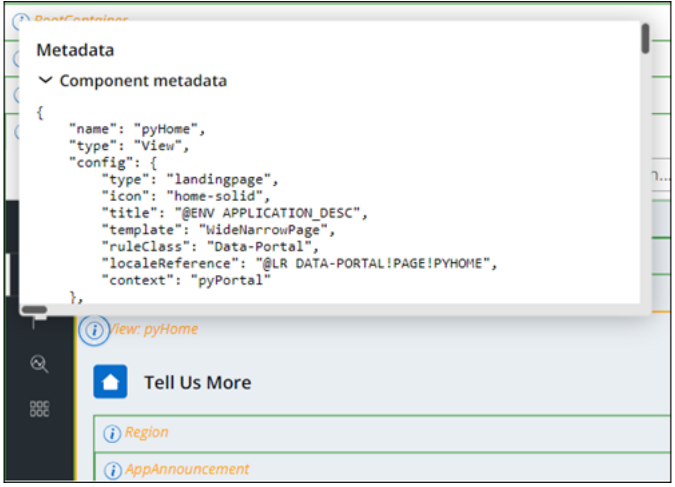


Announcements

We've launched a brand new experience to accelerate your workflow. Check out the guides to help you get the most of Cosmos.

- New React based UI rendering
- Support for followers and recents
- New authoring experience

3. Enable your **Dev Tools** (F12), and in the Console, enter PCore.getDebugger().toggleXRay(true);
4. Investigate the metadata for the Views associated with the Tell Us More announcements, including:
   ○ View: pyHome

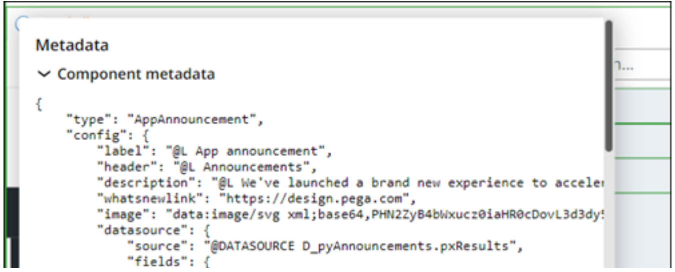


Metadata

Component metadata

```
{
    "name": "pyHome",
    "type": "View",
    "config": {
        "type": "landingpage",
        "icon": "home-solid",
        "title": "@ENV APPLICATION_DESC",
        "template": "WideNarrowPage",
        "ruleClass": "Data-Portal",
        "localeReference": "@LR DATA-PORTAL!PAGE!PYHOME",
        "context": "pyPortal"
    },
```

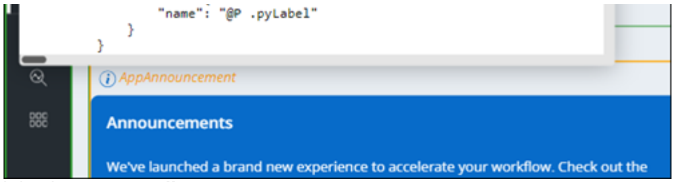
   ○ AppAnnouncement



Metadata

Component metadata

```
{
    "type": "AppAnnouncement",
    "config": {
        "label": "@L App announcement",
        "header": "@L Announcements",
        "description": "@L We've launched a brand new experience to acceler
        "whatsnewlink": "https://design.pega.com",
        "image": "data:image/svg xml;base64,PHN2ZyB4bWxucz0iaHR0cDovL3d3dy
        "datasource": {
            "source": "@DATASOURCE D_pyAnnouncements.pxResults",
            "fields": {
```



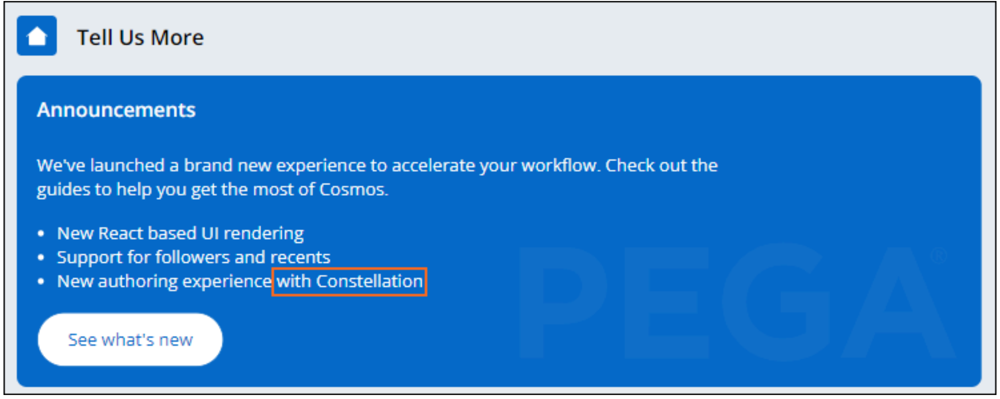
```
            "name": "@P .pyLabel"
        }
    }
```

AppAnnouncement

Announcements

We've launched a brand new experience to accelerate your workflow. Check out the

5. Switch back to **Dev Studio.**
6. Using the **App** explorer, enter Rule-UI-View.
7. From the list of Views, open the **pyHome** View and review the JSON data.
8. Note that the datasource for the pyHome Announcements information is driven by the *D_pyAnnouncements* Data Page.
9. Using the Dev Studio Search bar, open the *D_pyAnnouncements* Data Page.
10. In the Data Sources section, open the *pyInitAnnouncements* Data Transform.
11. In line 3.1, update the text in the Source field to read New authoring experience with Constellation
12. Launch a new instance of the Customer Portal. On the Home page, note that the third bullet point is updated.



Tell Us More

Announcements

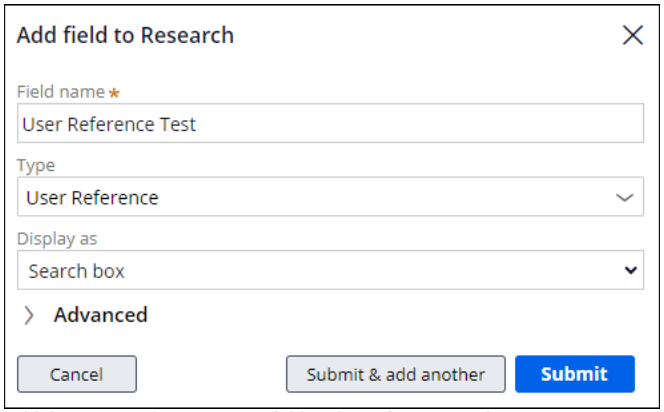
We've launched a brand new experience to accelerate your workflow. Check out the guides to help you get the most of Cosmos.

- New React based UI rendering
- Support for followers and recents
- New authoring experience with Constellation

6Debug DX API calls

Note: You can collect more details about the request and response of your DX API with the DebugPegaAPI system setting. When you set the DebugPegaAPI to true, your system captures information about API errors in the PEGA log. The DebugPegaAPI system setting is particularly useful for debugging page instructions and field security errors.

1. Switch back to **App Studio**. In the Research Case Type Data Model, create a field named User Reference Test with a Type of **User reference**.



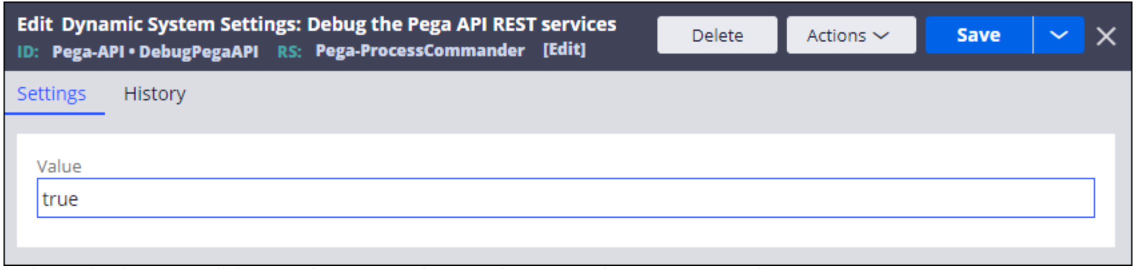
Add field to Research

Field name *: User Reference Test

Type: User Reference

Display as: Search box

2. Add the **User Reference Test** field to the **Debug** Step View.
3. Switch back to **Dev Studio**, search for the *DebugPegaAPI* Dynamic System setting and set it equal to true.

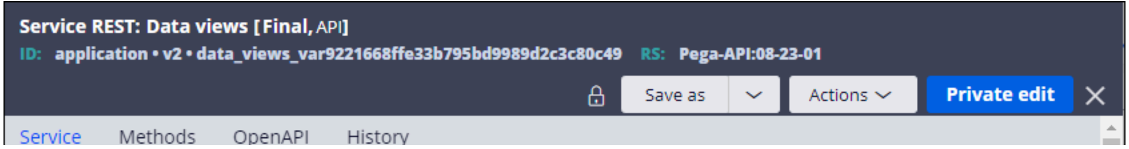


Edit Dynamic System Settings: Debug the Pega API REST services

ID: Pega-API • DebugPegaAPI RS: Pega-ProcessCommander [Edit]

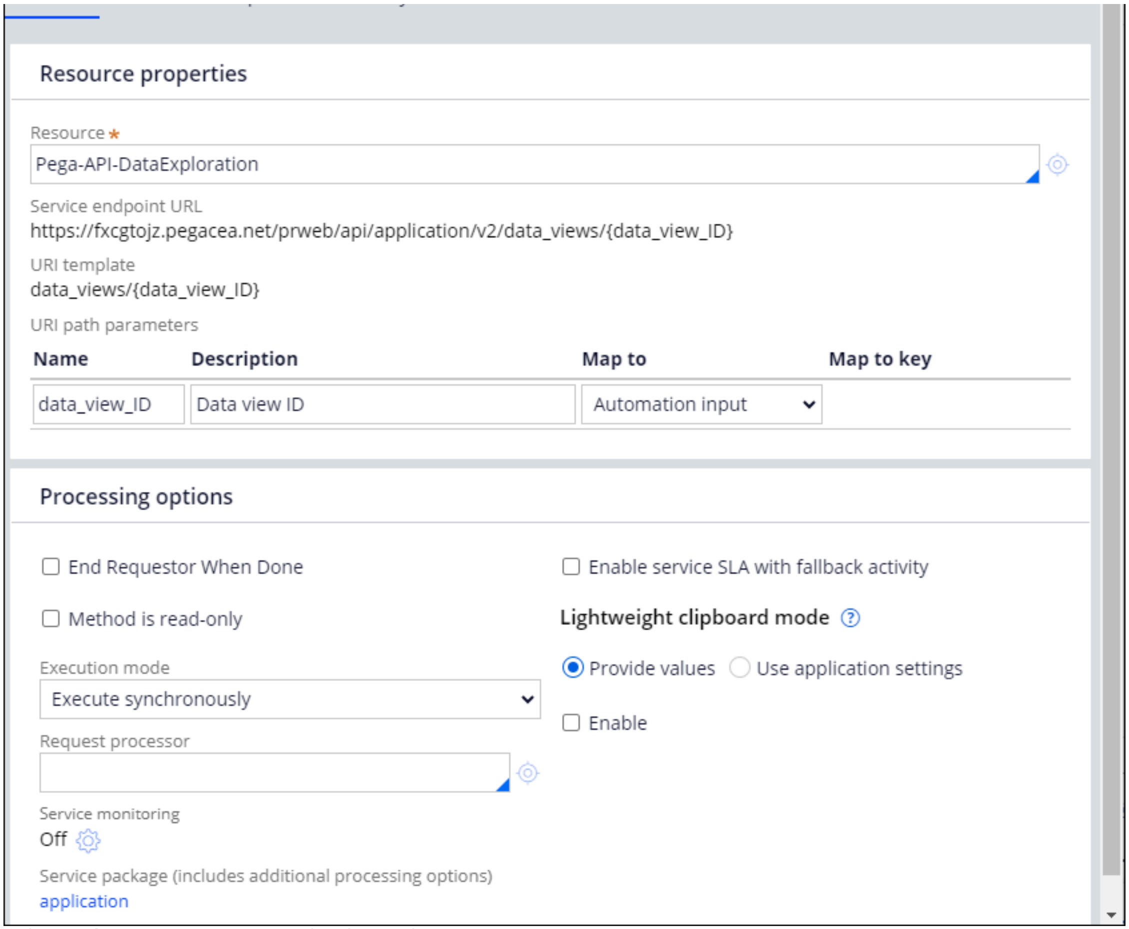
Value: true

4. In the navigation pane, click **Records > Integration-Services > Service Rest** to open the Service Rest that handles data Views (URL template: **data_views/{data_view_ID}**).



Service REST: Data views [Final, API]

ID: application • v2 • data_views_var9221668ffe33b795bd9989d2c3c80c49 RS: Pega-API:08-23-01



Resource properties

Resource *
Pega-API-DataExploration

Service endpoint URL
https://fxcgtojz.pegacea.net/prweb/api/application/v2/data_views/{data_view_ID}

URI template
data_views/{data_view_ID}

URI path parameters

Name	Description	Map to	Map to key
data_view_ID	Data view ID	Automation input	

Processing options

□ End Requestor When Done

□ Method is read-only

Execution mode
Execute synchronously

Request processor

Service monitoring
Off

Service package (includes additional processing options)
application

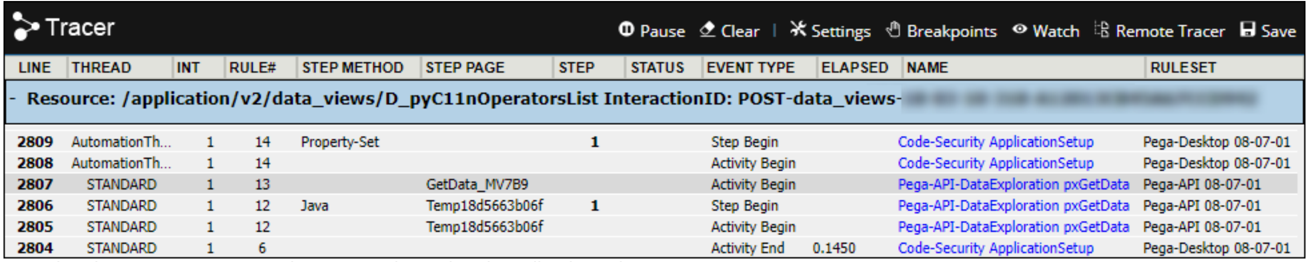
□ Enable service SLA with fallback activity

Lightweight clipboard mode

● Provide values ○ Use application settings

□ Enable

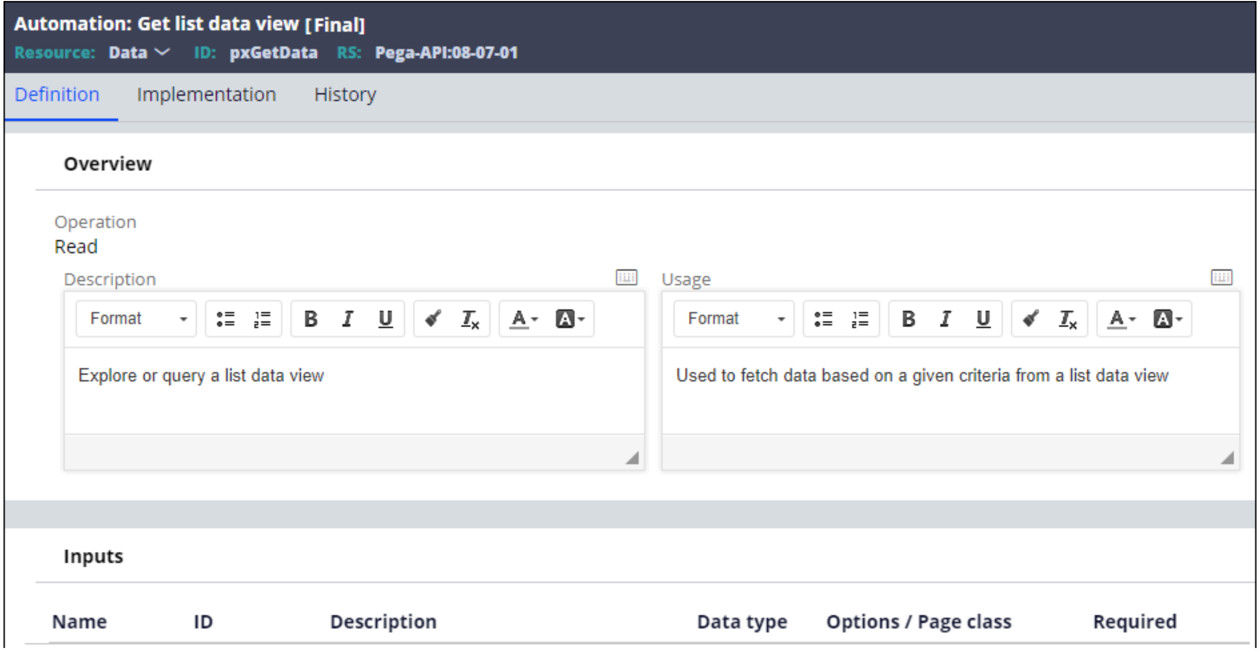
5. Select **Actions > Trace** to start tracing the service.
6. Launch the Customer Portal and create a new Research Case.
7. Navigate to the **Debug** Assignment, and return to the **Tracer**.
8. Expand **Resource: /application/v2/data_views/D_pyC11nOperatorsList** and search for the Automation *pxGetData*.



- Resource: /application/v2/data_views/D_pyC11nOperatorsList InteractionID: POST-data_views-[redacted]

LINE	THREAD	INT	RULE#	STEP METHOD	STEP PAGE	STEP	STATUS	EVENT TYPE	ELAPSED	NAME	RULESET
2809	AutomationTh...	1	14	Property-Set		1		Step Begin		Code-Security ApplicationSetup	Pega-Desktop 08-07-01
2808	AutomationTh...	1	14					Activity Begin		Code-Security ApplicationSetup	Pega-Desktop 08-07-01
2807	STANDARD	1	13		GetData_MV7B9			Activity Begin		Pega-API-DataExploration pxGetData	Pega-API 08-07-01
2806	STANDARD	1	12	Java	Temp18d5663b06f	1		Step Begin		Pega-API-DataExploration pxGetData	Pega-API 08-07-01
2805	STANDARD	1	12		Temp18d5663b06f			Activity Begin		Pega-API-DataExploration pxGetData	Pega-API 08-07-01
2804	STANDARD	1	6					Activity End	0.1450	Code-Security ApplicationSetup	Pega-Desktop 08-07-01

9. Note that the *pxGetData* Automation returns data related to a list data View.



Automation: Get list data view [Final]

Resource: Data ID: pxGetData RS: Pega-API:08-07-01

Overview

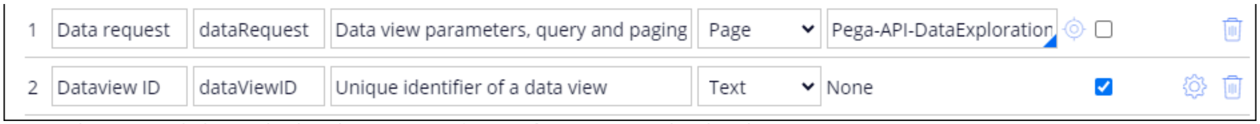
Operation
Read

Description
Explore or query a list data view

Usage
Used to fetch data based on a given criteria from a list data view

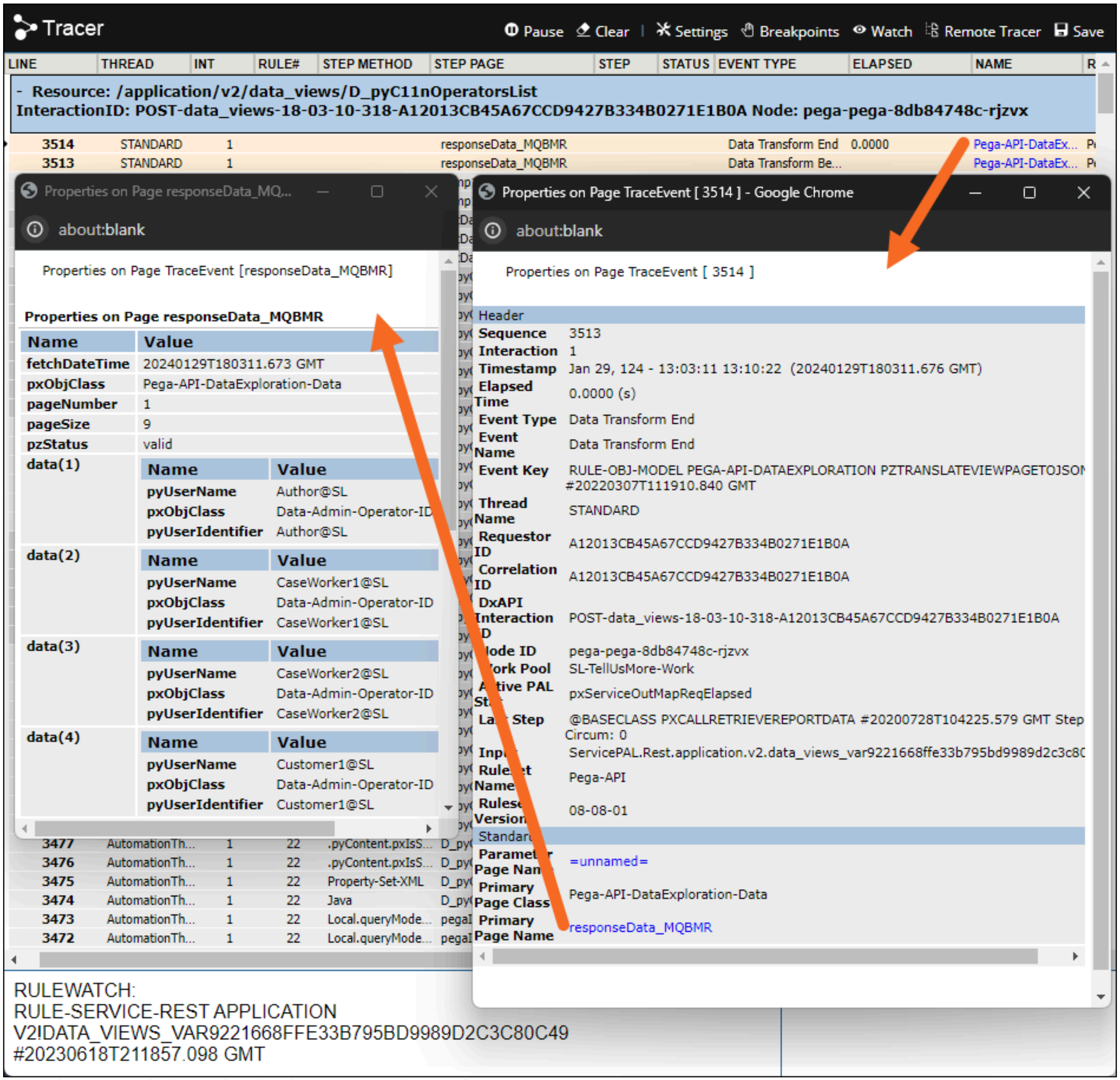
Inputs

Name	ID	Description	Data type	Options / Page class	Required



1	Data request	dataRequest	Data view parameters, query and paging	Page	Pega-API-DataExploration	☐	
2	Dataview ID	dataViewID	Unique identifier of a data view	Text	None	☑	

10. Access the Tracer window and select the **DataTransformEnd** Event type and review the list returned as JSON.



LINE	THREAD	INT	RULE#	STEP METHOD	STEP PAGE	STEP	STATUS	EVENT TYPE	ELAPSED	NAME	R
- Resource: /application/v2/data_views/D_pyC11nOperatorsList InteractionID: POST-data_views-18-03-10-318-A12013CB45A67CCD9427B334B0271E1B0A Node: pega-pega-8db84748c-rjzvx											
3514	STANDARD	1			responseData_MQBMR			Data Transform End	0.0000	Pega-API-DataEx...	P
3513	STANDARD	1			responseData_MQBMR			Data Transform Be...		Pega-API-DataEx...	P
3477	AutomationTh...	1	22	.pyContent.pxIsS...	D_py						
3476	AutomationTh...	1	22	.pyContent.pxIsS...	D_py						
3475	AutomationTh...	1	22	Property-Set-XML	D_py						
3474	AutomationTh...	1	22	Java	D_py						
3473	AutomationTh...	1	22	Local.queryMode...	pegaI						
3472	AutomationTh...	1	22	Local.queryMode...	pegaI						

Properties on Page responseData_MQ...

about:blank

Properties on Page TraceEvent [responseData_MQBMR]

Properties on Page responseData_MQBMR

Name	Value
fetchDateTime	20240129T180311.673 GMT
pxObjClass	Pega-API-DataExploration-Data
pageNumber	1
pageSize	9
pzStatus	valid
data(1)	Name: pyUserName – Value: Author@SL pxObjClass – Data-Admin-Operator-ID pyUserIdentifier – Author@SL
data(2)	pyUserName – CaseWorker1@SL pxObjClass – Data-Admin-Operator-ID pyUserIdentifier – CaseWorker1@SL
data(3)	pyUserName – CaseWorker2@SL pxObjClass – Data-Admin-Operator-ID pyUserIdentifier – CaseWorker2@SL
data(4)	pyUserName – Customer1@SL pxObjClass – Data-Admin-Operator-ID pyUserIdentifier – Customer1@SL

Properties on Page TraceEvent [3514] - Google Chrome

about:blank

Properties on Page TraceEvent [3514]

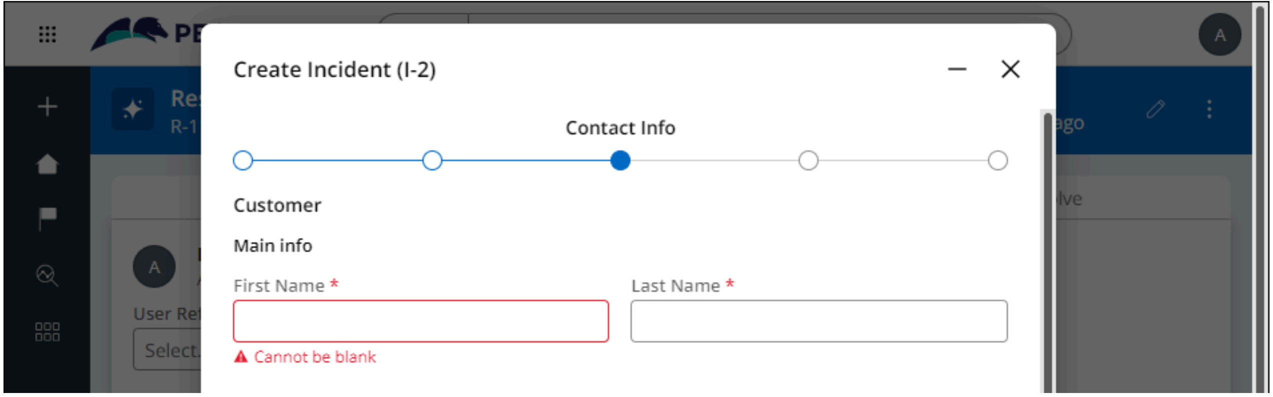
Header	
Sequence	3513
Interaction	1
Timestamp	Jan 29, 124 - 13:03:11 13:10:22 (20240129T180311.676 GMT)
Elapsed Time	0.0000 (s)
Event Type	Data Transform End
Event Name	Data Transform End
Event Key	RULE-OBJ-MODEL PEGA-API-DATAEXPLORATION PZTRANSLATEVIEWPAGETOJSON #20220307T111910.840 GMT
Thread Name	STANDARD
Requestor ID	A12013CB45A67CCD9427B334B0271E1B0A
Correlation ID	A12013CB45A67CCD9427B334B0271E1B0A
DxAPI Interaction ID	POST-data_views-18-03-10-318-A12013CB45A67CCD9427B334B0271E1B0A
Node ID	pega-pega-8db84748c-rjzvx
Work Pool	SL-TellUsMore-Work
Active PAL Stat	pxServiceOutMapReqElapsed
Last Step	@BASECLASS PXCALLRETRIEVEREPORTDATA #20200728T104225.579 GMT Step Circum: 0
Input	ServicePAL.Rest.application.v2.data_views_var9221668ffe33b795bd9989d2c3c80
Ruleset Name	Pega-API
Ruleset Version	08-08-01
Standard	
Parameter Page Name	=unnamed=
Primary Page Class	Pega-API-DataExploration-Data
Primary Page Name	responseData_MQBMR

RULEWATCH:
RULE-SERVICE-REST APPLICATION
V2!DATA_VIEWS_VAR9221668FFE33B795BD9989D2C3C80C49
#20230618T211857.098 GMT

11. Close the Tracer windows related to the **DataTransformEnd** Event type.

7Review the Data Page cache

1. Return to the Customer Portal and create a new Incident Case.
2. Navigate to the **Contact Info** Step.
3. Open the Developer Tools (F12), navigate to the Console tab, and enter: PCore.getDataPageUtils();
4. Explore the Data Pages that are cached at the Contact Info Step.



Create Incident (I-2)

Contact Info

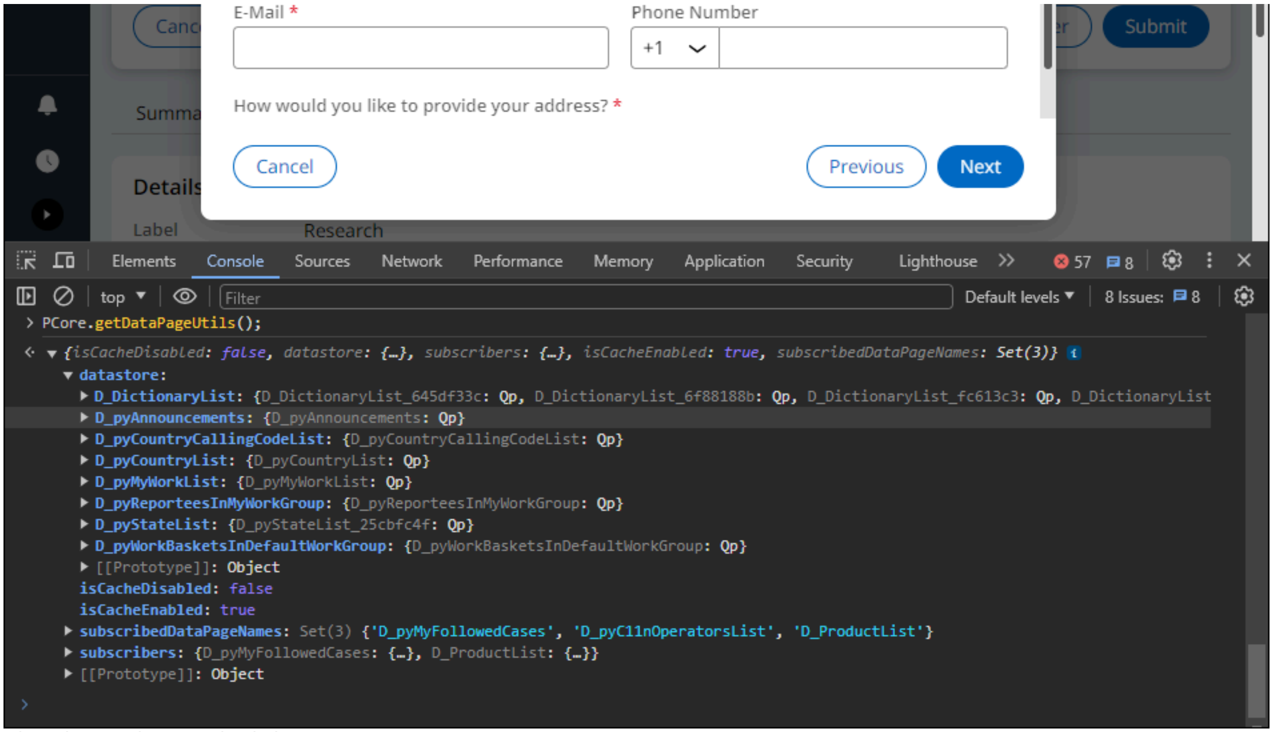
Customer

Main info

First Name *

Cannot be blank

Last Name *



E-Mail *

Phone Number

+1

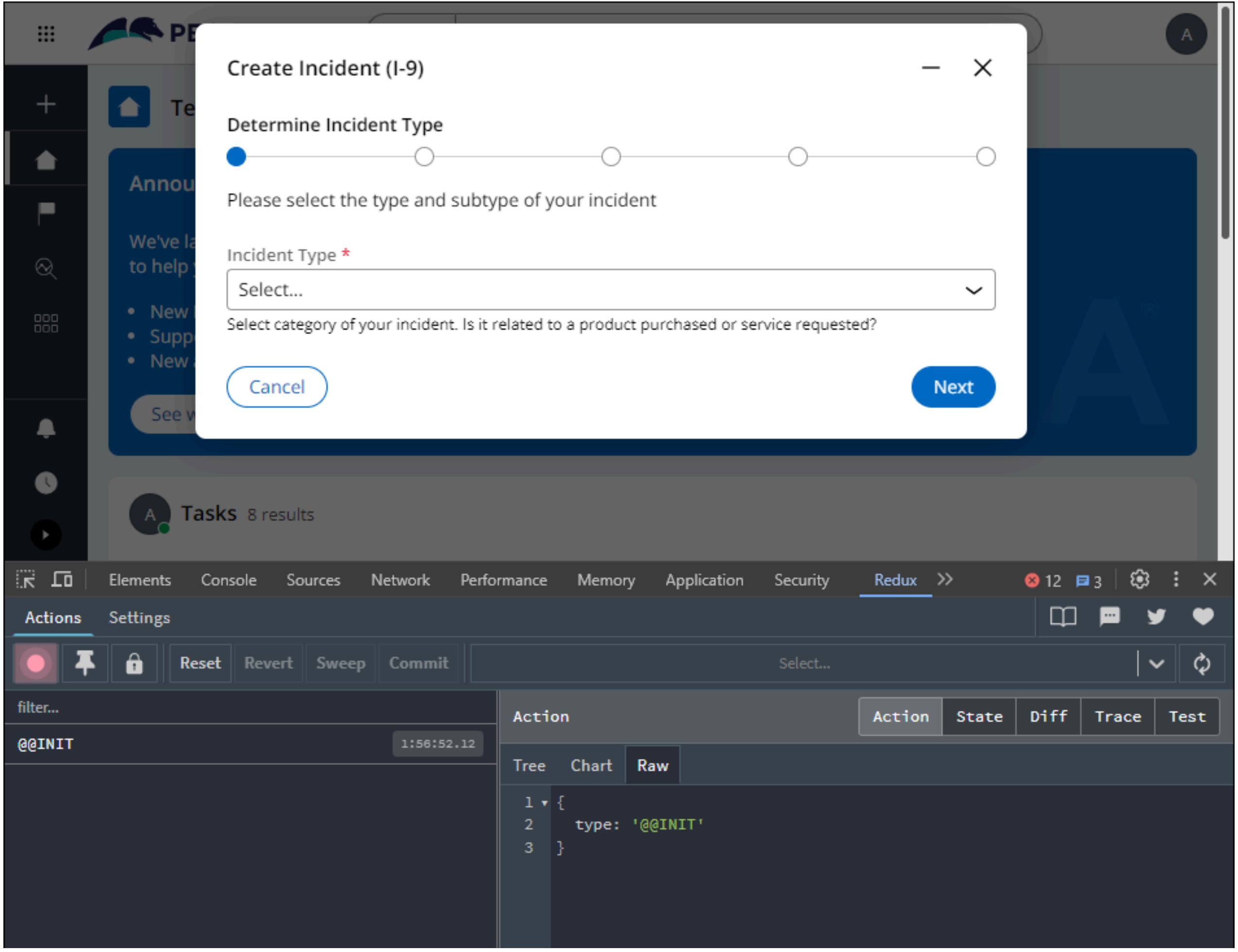
How would you like to provide your address? *

```
> PCore.getDataPageUtils();
< ▼ {isCacheDisabled: false, datastore: {…}, subscribers: {…}, isCacheEnabled: true, subscribedDataPageNames: Set(3)}
    ▼ datastore:
      ▶ D_DictionaryList: {D_DictionaryList_645df33c: Qp, D_DictionaryList_6f88188b: Qp, D_DictionaryList_fc613c3: Qp, D_DictionaryList
      ▶ D_pyAnnouncements: {D_pyAnnouncements: Qp}
      ▶ D_pyCountryCallingCodeList: {D_pyCountryCallingCodeList: Qp}
      ▶ D_pyCountryList: {D_pyCountryList: Qp}
      ▶ D_pyMyWorkList: {D_pyMyWorkList: Qp}
      ▶ D_pyReporteesInMyWorkGroup: {D_pyReporteesInMyWorkGroup: Qp}
      ▶ D_pyStateList: {D_pyStateList_25cbfc4f: Qp}
      ▶ D_pyWorkBasketsInDefaultWorkGroup: {D_pyWorkBasketsInDefaultWorkGroup: Qp}
      ▶ [[Prototype]]: Object
      isCacheDisabled: false
      isCacheEnabled: true
    ▶ subscribedDataPageNames: Set(3) {'D_pyMyFollowedCases', 'D_pyC11nOperatorsList', 'D_ProductList'}
    ▶ subscribers: {D_pyMyFollowedCases: {…}, D_ProductList: {…}}
    ▶ [[Prototype]]: Object
>
```

5. Close the Developer Tools window.

8Examine React Changes with Redux Dev Tools

1. Install the Redux-DevTools in your browser by searching for Redux DevTools, or:
   - For Chrome, you can add the Redux DevTools from here: Google Chrome Store Redux DevTools
   - For Firefox, you can add the Redux DevTools from here: Firefox Redux DevTools
   - For Edge, you can add the Redux DevTools from here: Edge Redux DevTools
2. Open the Dev Tools (F12) and navigate to the **Redux** Tab.
3. Create a new Incident Case, and navigate to the **Determine Incident Type** Step.
4. Note the changes to the Action information as you fill out the form.



Create Incident (I-9)

Determine Incident Type

Please select the type and subtype of your incident

Incident Type *

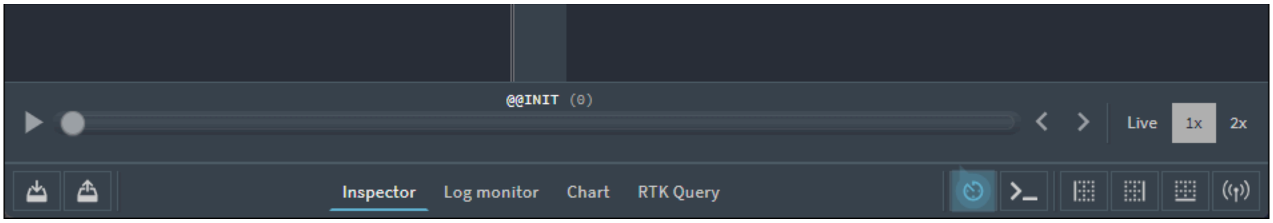
Select...

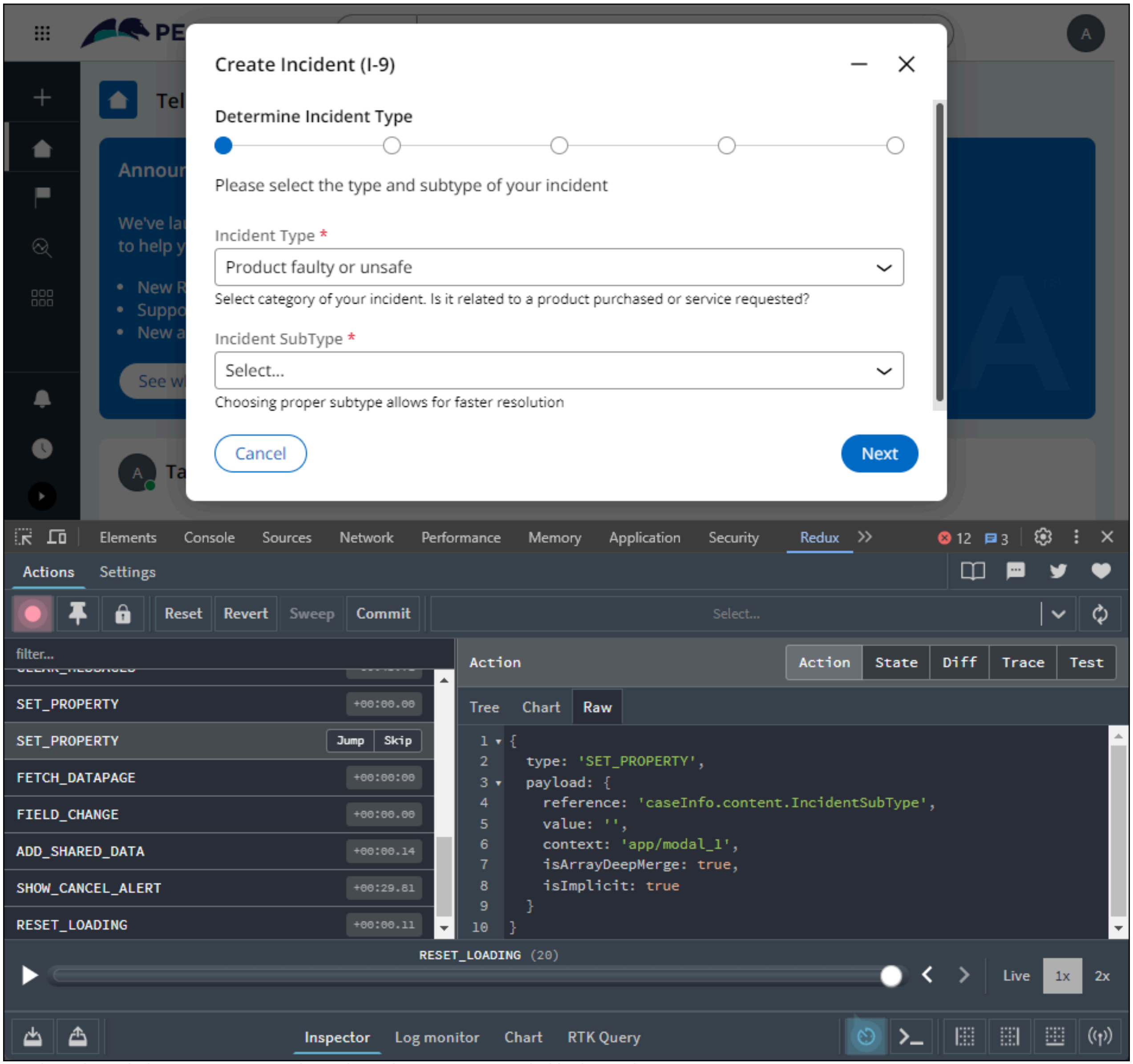
Select category of your incident. Is it related to a product purchased or service requested?

@@INIT 1:56:52.12

Action

```
{
  type: '@@INIT'
}
```





Create Incident (I-9)

Determine Incident Type

Please select the type and subtype of your incident

Incident Type *

Product faulty or unsafe

Select category of your incident. Is it related to a product purchased or service requested?

Incident SubType *

Select...

Choosing proper subtype allows for faster resolution

```
SET_PROPERTY          +00:00.00
SET_PROPERTY
FETCH_DATAPAGE        +00:00:00
FIELD_CHANGE          +00:00.00
ADD_SHARED_DATA       +00:00.14
SHOW_CANCEL_ALERT     +00:29.81
RESET_LOADING         +00:00.11
```

```
{
  type: 'SET_PROPERTY',
  payload: {
    reference: 'caseInfo.content.IncidentSubType',
    value: '',
    context: 'app/modal_1',
    isArrayDeepMerge: true,
    isImplicit: true
  }
}
```

5. Continue through the Incident Case and note how changes are recorded in the Redux Dev Tools.

From <https://academy.pega.com/challenge/debugging-constellation-applications/v1/in/70261>