



# NEOMATIC

build your business with NEO experience

## NEO.CPO Installation Guide

### Table of Contents

|                                                      |           |
|------------------------------------------------------|-----------|
| <b>Installation .....</b>                            | <b>2</b>  |
| <b>Permission Sets .....</b>                         | <b>4</b>  |
| <b>Licenses .....</b>                                | <b>4</b>  |
| User-based license .....                             | 4         |
| Org-wide license .....                               | 5         |
| <b>Maintaining the Sales Service Hierarchy .....</b> | <b>5</b>  |
| Record Creation.....                                 | 6         |
| Translations.....                                    | 9         |
| <b>Global App Settings .....</b>                     | <b>11</b> |
| Colouring.....                                       | 12        |
| Partner View Feature .....                           | 13        |
| Persona Feature .....                                | 13        |
| <b>Reading Path Settings .....</b>                   | <b>15</b> |
| Reading Path 1 - Opportunity.....                    | 16        |
| Reading Path 2 - Order .....                         | 16        |
| Reading Path 3 - Asset.....                          | 16        |
| <b>CPO Card Actions .....</b>                        | <b>18</b> |
| Screen Flow prerequisites.....                       | 18        |
| <b>Enhanced Customization for NEO.CPO .....</b>      | <b>21</b> |
| Custom Partner Feature.....                          | 21        |
| Custom Persona Feature .....                         | 22        |
| Custom Reading Path .....                            | 23        |



# NEOMATIC

build your business with NEO experience

## Installation

We only allow to install our Apps via the AppExchange.

Navigate to the AppExchange: <https://appexchange.salesforce.com/>

Search for NEO.CPO

The screenshot shows the Salesforce AppExchange interface. At the top, there's a navigation bar with 'AppExchange' and links for Home, Explore, Collections, Consultants, and Learn. A search bar contains 'NEO.CPO'. Below the search bar, the results are filtered to 'Apps & Solutions (2)' and 'Consultants (0)'. Two app tiles are visible: 'NEO.CPO - Customer Portfolio Overview' and 'NEO.PRM - Pro-active Sales Partner M...'. The first tile has a 5-star rating and is categorized under 'Sales Productivity' and 'Sales Intelligence'. The second tile has a 4-star rating and is categorized under 'Productivity'. On the left, there are filters for Pricing (Free, Freemium, Paid, Paid Add-On Required, Discounts for Nonprofits) and Rating (5 stars & Up, 4 stars & Up, 3 stars & Up, 2 stars & Up, 1 star & Up, Unrated).

Click the "NEO.CPO - Customer Portfolio Overview" Tile.

The screenshot shows the detailed page for the 'NEO.CPO - Customer Portfolio Overview' app. The app is by NEOMATIC AG and is described as 'Optimize your Customer 360 view with a unique product portfolio overview'. It is a Salesforce App with no ratings. The price is 'Starting at €2.95 EUR/user/month' with discounts available for nonprofits. The app is categorized under 'Industries' (Financial Services, High Tech, Manufacturing), 'Business Need' (Sales Productivity, Sales Intelligence, Data Visualization), and 'Requires' (Sales Cloud). It is compatible with Sales Cloud, Service Cloud, and Financial Services Cloud. The page includes two preview images: one showing the 'NEO.CPO - Customer Portfolio Overview' interface and another showing the 'NEO.CPO - Salesforce UI Component - Top Feature'. The top feature preview lists several benefits: a visual overview of all essential elements, a three-level structured layout, a toggle between compact and detailed views, a search bar, and a fast & easy implementation in minutes.

Click the "Get It Now" Button.





# NEOMATIC

build your business with NEO experience

We recommend installing the App on your sandbox first.

×

Where do you want to install this package?

Install in a Production Environment

Install this package in the org where you or your users work, including Developer Edition orgs.

\* Connected Salesforce Accounts ⓘ

Choose...

Don't see your account? [More Info](#)

Install in Production

Install in a Sandbox

Test this package in a copy of a production org.

Install in Sandbox

Install in a Trial

Try this package in a free org that includes sample data.

Install in Trial

Cancel

We recommend installing the App for Admins only.

**Install SF\_CPO\_App**  
By

☒ Install for Admins Only

☐ Install for All Users

☐ Install for Specific Profiles...

Install

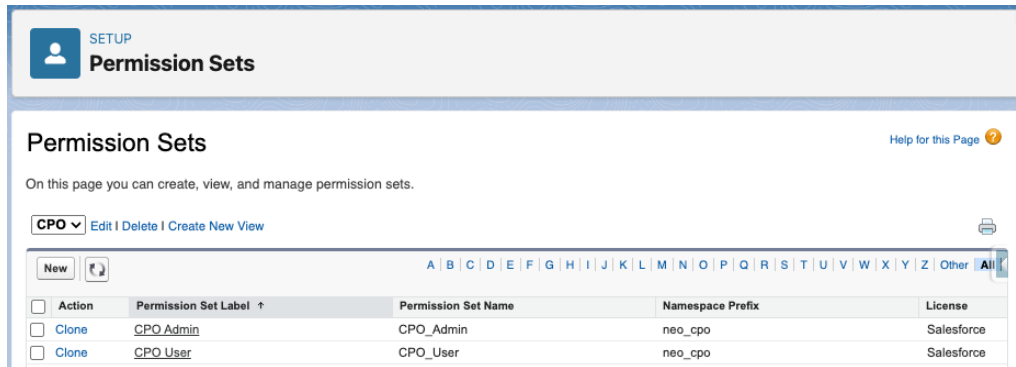
Cancel

| App Name   | Publisher | Version Name | Version Number |
|------------|-----------|--------------|----------------|
| SF_CPO_App |           | 2.1.0        | 2.1            |



## Permission Sets

With our App you will receive two permission sets to be used to assign to your users.



The CPO Admin Permission Set is supposed to be assigned to users that would maintain the sales and service hierarchy. (Note: More about the sales and service hierarchy will be explained later)

The CPO User Permission Set is supposed to be assigned to your standard users, so that they can use the App.

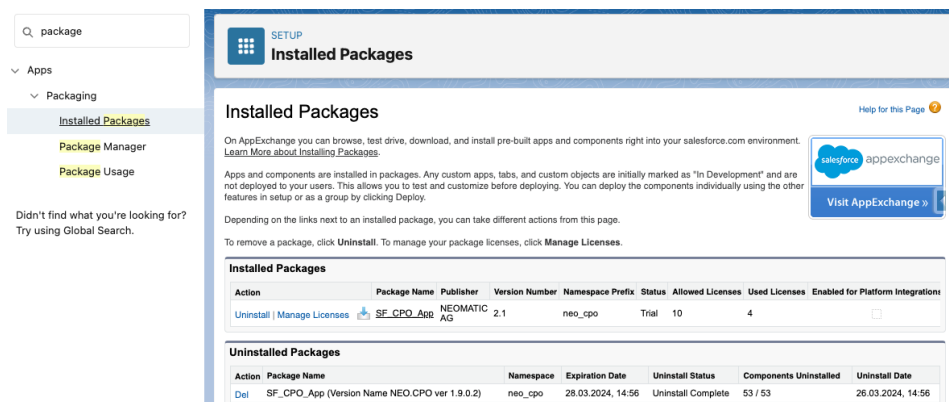
A user that is assigned to the system administrator profile does have already full access to the app per installation option you have chosen.

## Licenses

There is no license assignment needed on sandboxes. Additional steps may be necessary for productive orgs, depending on the licensing model you have chosen with us.

### User-based license

For a user-based license model you need to manage licenses per user.



Click on Manage Licenses

# NEOMATIC

build your business with NEO experience

Package Details  
**SF\_CPO\_App**  
[Back to Previous Page](#) [Help for this Page](#)

☐ Enable for Platform Integrations

|                                   |                          |                  |             |
|-----------------------------------|--------------------------|------------------|-------------|
| Package Name                      | SF_CPO_App               | Publisher        | NEOMATIC AG |
| Status                            | Trial                    | Allowed Licenses | 10          |
| Expiration Date                   | 25.04.2024               | Used Licenses    | 2           |
| Enabled for Platform Integrations | <input type="checkbox"/> |                  |             |

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | **All**

**Licensed Users** [Add Users](#) [Remove Multiple Users](#)

| Action                 | Full Name ↑    | Role | Active | Profile              |
|------------------------|----------------|------|--------|----------------------|
| <a href="#">Remove</a> | Neomatic_Admin |      | ✓      | System Administrator |
| <a href="#">Remove</a> | User_Test      |      | ✓      | Standard User        |

Click on Add Users to add as many users as you have available licenses.

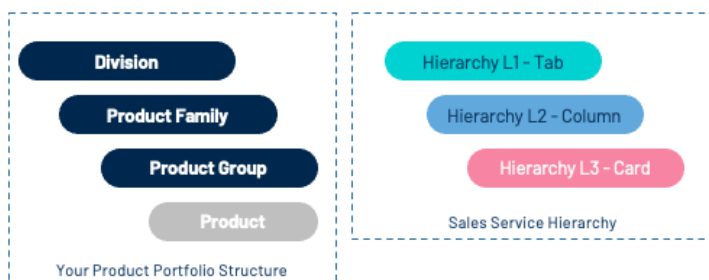
## Org-wide license

If you have with org-wide license, you do not need to care about single user assignment as every active user of the org is entitled to use the App.

## Maintaining the Sales Service Hierarchy

The Sales Service Hierarchy is mandatory for the App, hence need to be maintained. We recommend to create the hierarchy based on your product portfolio structure.

The following illustration shows you an example of a product portfolio structure combined with the corresponding hierarchy level setup.



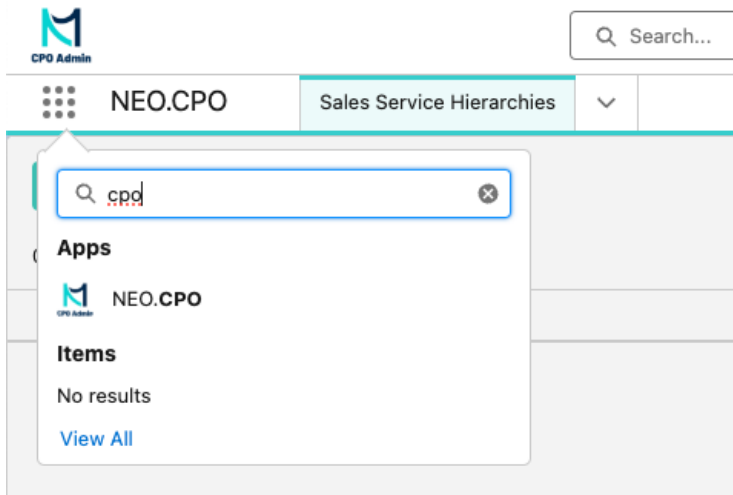
A full App configuration requires records for all hierarchy levels.

To maintain the sales service hierarchy you can use the NEO.CPO App to be opened from the App Launcher.



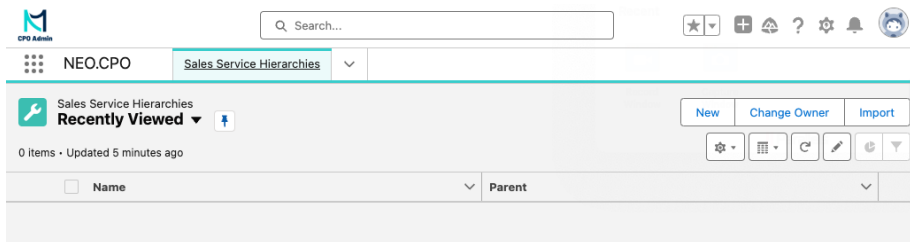
# NEOMATIC

build your business with NEO experience



The initial tab of that console tab is for the Sales Service Hierarchy Object. As you can see there are no objects, yet.

## Record Creation



Click on New to create the first object.





# NEOMATIC

build your business with NEO experience

To define the first level fill SSH Name Field and select from picklist Hierarchy - **Tab - Level 1**. This level requires only three fields to be populated.

To define the second level, click on the previously created record → Related Tab → New Button and fill the SSH field Hierarchy - **Column - Level 2**. This level requires the same fields like level 2 but with additional Parent field which is prefilled if created from related list.





# NEOMATIC

build your business with NEO experience

Information

\*Name

↺

Column

Active ⓘ

↺

☒

\*Hierarchy

↺

Column - Level2 ▼

Is For All Parents ⓘ

☐

Parent

 Tab

×

For the third level repeat the previous step with filling Hierarchy - **Card - Level 3** and additional to that add icon to each third level by inserting their name text from [Icons - Lightning Design System](#). It is recommended to use the utility icons, for example: utility:agent\_home

## New Sales Service Hierarchy

\* = Required Information

Information

\*Name

↺

Card 1

Active ⓘ

↺

☒

\*Hierarchy

↺

Card - Level3 ▼

Is For All Parents ⓘ

☐

Parent

 Column

×

Hide Empty Cards ⓘ

☐

Icon ⓘ

↺

utility:agent\_home

External Id

Owner

 Greg Chrzaszczyk

Description

For the fourth level (optional) repeat the previous step with filling Hierarchy - **Badge - Level 4** but do not add Icons. You can either add Parent lookup to apply this Badge configuration on specific Card Level 3 or select **Is For All Parents** to apply it on all Level 3 cards.





# NEOMATIC

build your business with NEO experience

Information

\* Name ↶

Badge

Active ⓘ ↶

☒

\* Hierarchy ↶

Badge - Level4 ▾

Is For All Parents ⓘ ↶

☐

Parent

 Card 1

×

or

Information

\* Name ↶

Badge

Active ⓘ ↶

☒

\* Hierarchy ↶

Badge - Level4 ▾

Is For All Parents ⓘ ↶

☒

Parent

Search Sales Service Hierarchies...

🔍

To change the order of the tabs, columns, cards and badges within their respective place, fill an integer number into the field "Sort Order", the lower the number, the further left or up a tab/column/card/tab will appear. It is recommended to use below range:

| Tab - Level 1 | Column - Level 2 | Card - Level 3 | Badge - Level 4 |
|---------------|------------------|----------------|-----------------|
| 1-9           | 10-99            | 100-999        | > 1000          |

And start numbering with:

| Tab - Level 1 | Column - Level 2 | Card - Level 3 | Badge - Level 4 |
|---------------|------------------|----------------|-----------------|
| 3             | 30               | 300            | 3000            |

## Translations

NEO CPO App supports translations via standard Salesforce Custom Labels. By default English values are provided.

Use Translation Workbench to maintain translated values. Specify languages for translation and assign translators for each language. Manage translated values for any Salesforce supported language. Translators can maintain translations directly through the workbench, or you can export translation files for bulk translation imports.

You can create dedicated List View for CPO labels only with the filter:

Step 2. Specify Filter Criteria

Filter By Additional Fields (Optional):

| Field              | Operator | Value   |
|--------------------|----------|---------|
| Namespace Prefix ▾ | equals ▾ | neo_cpo |

If you would like to translate your sales service hierarchy records, then please follow these steps.



# NEOMATIC

build your business with NEO experience

1. Create Custom Labels
2. Add Custom Label API name to your sales service hierarchy records

**Custom Labels**

CPO

Custom labels enable developers to create multilingual applications by preselecting that Salesforce supports. You can create up to 6,000 custom labels, and each

View: CPO Edit Create New View

| Action     | Name         | Value      |
|------------|--------------|------------|
| Edit   Del | SSH_01       | Non life   |
| Edit   Del | SSH_01_01    | Vehicle    |
| Edit   Del | SSH_01_01_01 | Car        |
| Edit   Del | SSH_01_01_02 | Motorcycle |
| Edit   Del | SSH_01_01_03 | Bicycle    |
| Edit   Del | SSH_01_01_04 | Truck      |

Vehicle

- Car
- Motorcycle
- Motorcycle
- Bicycle
- Truck

The easiest way to populate API names to your records is to use an editable list view on the object.

Sales Service Hierarchies

SSH Level 1

3 Items • Sorted by Translation Custom Label API Name • Filtered by All sales service hierarchies - Hierarchy • Updated 29 minutes ago

| Name     | Translation Custom Label API Name | Hierarchy    | Active                              | SortOrder |
|----------|-----------------------------------|--------------|-------------------------------------|-----------|
| Non life | SSH_01                            | Tab - Level1 | <input checked="" type="checkbox"/> | 1         |
| Life     | SSH_02                            | Tab - Level1 | <input checked="" type="checkbox"/> | 2         |
| Health   | SSH_03                            | Tab - Level1 | <input checked="" type="checkbox"/> | 3         |

Example:

**Custom Labels**

| Action     | Name          | Value           | Categories | Language | Protected Component      | Namespace |
|------------|---------------|-----------------|------------|----------|--------------------------|-----------|
| Edit   Del | Configuration | Configuration   | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | Filter        | Filter          | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | Load          | Load            | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | Loading       | Loading         | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | Parent        | Parent          | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_01        | Non life        | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_01_01     | Vehicle         | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_01_02     | Home            | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_01_03     | Personal        | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_02        | Life            | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_02_01     | Life Group      | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_02_02     | Life Individual | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_03        | Health          | CPO        | English  | <input type="checkbox"/> | neo_cpo   |
| Edit   Del | SSH_03_01     | Health          | CPO        | English  | <input type="checkbox"/> | neo_cpo   |

Non life Life Health

Vehicle

Home

Personal

Liability

Life Group

Life Individual

Health

Comprehensive

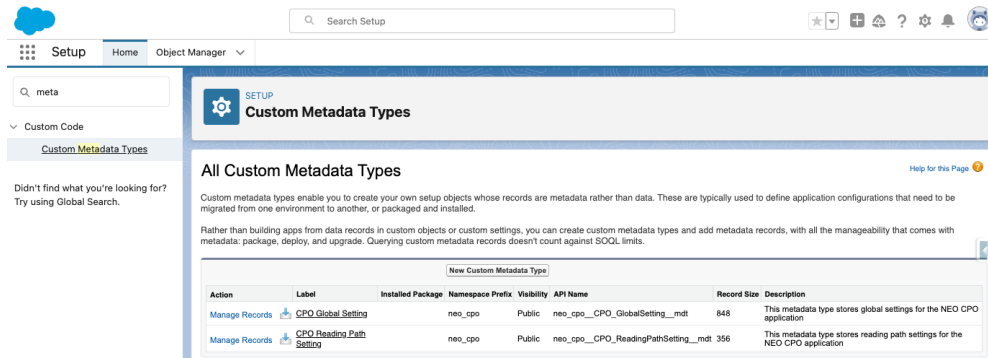


# NEOMATIC

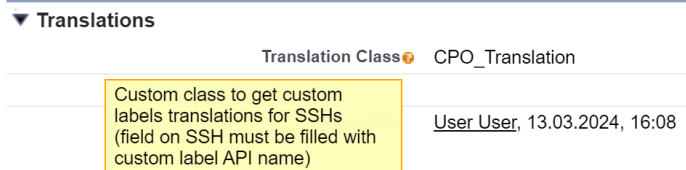
build your business with NEO experience

If you need some specific translation logic, you can add custom apex class for that.

The class name must be configured in the custom metadata settings for our App. Navigate to custom metadata types.



To add custom class Click on Manage Records for CPO Global Settings.



In the section translations add the class name of your custom translation class.

New Apex Class with this name must be deployed. Use below skeleton of custom translation class:

```
global with sharing class CPO_Translation implements Callable {
    global Object call(String param, Map<String, Object> args) {
        return System.Label
            .get(
                null,
                args.keySet().iterator().next(),
                UserInfo.getLanguage()
            );
    }
}
```

## Global App Settings

Navigate to custom metadata types.



# NEOMATIC

build your business with NEO experience

The screenshot shows the 'Setup' tab in the NEOMATIC interface. The main heading is 'Custom Metadata Types'. Below it, there's a section titled 'All Custom Metadata Types' with a 'Help for this Page' link. A paragraph explains that custom metadata types are used to define application configurations. Below this is a table titled 'New Custom Metadata Type' with columns: Action, Label, Installed Package, Namespace Prefix, Visibility, API Name, Record Size, and Description. The table lists two entries: 'CPO Global Setting' and 'CPO Reading Path Setting'. The 'CPO Global Setting' entry has a 'Record Size' of 848 and a description: 'This metadata type stores global settings for the NEO CPO application'. The 'CPO Reading Path Setting' entry has a 'Record Size' of 356 and a description: 'This metadata type stores reading path settings for the NEO CPO application'. A 'Manage Records' link is visible next to the 'CPO Global Setting' entry.

Click on Manage Records for CPO Global Settings.

The screenshot shows the 'CPO Global Setting (Managed)' configuration page. At the top, there's a yellow warning box stating: 'This CPO Global Setting is managed, meaning that you may only edit certain attributes. Display More Information'. Below this is the 'CPO Global Setting Detail' section with 'Edit' and 'Clone' buttons. The configuration is organized into sections: 'CPO Global Setting Name' (Global), 'Protected Component' (checkbox), 'Label' (Global), 'Namespace Prefix' (neo\_cpo), 'Colors' (Empty Card Color: 000000, Partner Card Color: 101010), 'Partner View Standard' (Partner Object Name: Partner, Partner Role Field: Role, Partner Account From Relation Name: AccountFrom, Partner Account To Relation Name: AccountTo), 'Partner View Custom' (Partner View Class), 'Persona Feature Standard' (Personas, Persona Field: Type), 'Persona Feature Custom' (Persona Feature Class), and 'Translations' (Translation Class). At the bottom, it shows 'Created By: Marcus Findeisen, 28.03.2024, 15:34' and 'Last Modified By: Marcus Findeisen, 28.03.2024, 15:34' with 'Edit' and 'Clone' buttons.

! DO NOT MODIFY **"Global"** record API name.  
Only **"Global"** value is valid for CPO Global Setting Name!

## Colouring

The App works in a way that every card where no match to any record does exists, the card shows up as an empty card. It is possible to define a colour as a hex-value, so that users can easily differ those cards from others.

The screenshot shows the 'Colors' configuration section. It has a heading '▼ Colors' and a label 'Empty Card Color' with a hex value '000000'. A yellow tooltip box is overlaid on the 'Empty Card Color' label, containing the text: 'This setting specifies cards icon color where there is no data for active SSH - Carl Level 4 (in HEX value - example: 000000)'. Below the tooltip, there are two radio button options: 'Partner' and 'Role'.





# NEOMATIC

build your business with NEO experience

If the partner feature is activated, the App does also show cards from other root Objects such as other Accounts on an Account Record Page. With the other colour setting you can define a background highlight for those to differentiate them.

Partner Card Color  f0f0f0

This setting specifies cards icon color for cards owned by a partner.  
(in HEX value - example: 000000)

 AccountFrom  
 AccountTo

## Partner View Feature

Default standard configuration considers standard Salesforce **Partner** object ([sforce\\_api\\_objects\\_partner](#)) with standard relation names.

**SETUP** Custom Metadata Types

CPO Global Setting

CPO Global Setting Detail Edit Delete Clone

|                         |        |                     |                          |
|-------------------------|--------|---------------------|--------------------------|
| CPO Global Setting Name | Global | Protected Component | <input type="checkbox"/> |
| Label                   | Global | Namespace Prefix    | neo_cpo                  |

▼ Colors

Empty Card Color  000000

▼ Partner View Settings

|                                                                                                         |         |                                                                                                                        |             |
|---------------------------------------------------------------------------------------------------------|---------|------------------------------------------------------------------------------------------------------------------------|-------------|
| Partner Object Name  | Partner | Partner Account From Relation Name  | AccountFrom |
| Partner Role Field   | Role    | Partner Account To Relation Name    | AccountTo   |

Partner View feature is not active if Partner Object Name field on CPO Global Settings is empty. Remaining field values in Partner View Settings section on layout will not be considered at all in this case.

To deactivate Partner View feature, simply remove value from Partner Object Name field on CPO Global Settings and keep or remove remaining field values in Partner View Settings section.

Standard Partner View feature can be customized with other Standard or Custom Object to be used.

## Persona Feature

This feature allows you to steer different CPO Dashboards for different account groupings. As part of our standard feature they are grouped by Account Type. You can customize that to any other field on Account.

To use the feature you need to add the values of the selected Account field (in our standard Account Type) to the Persona field.





# NEOMATIC

build your business with NEO experience

**Persona Feature Standard**

Personas  Partner  
Other

**Persona Feature Custom**

Standard Persona Feature - specify allowed Personas separated by new line (leave empty to deactivate Standard Persona globally - SSH -> Persona field is ignored)

Besides of this you need to add those values to the Sales Service Hierarchy Object – Field Persona.

SETUP > OBJECT MANAGER  
**Sales Service Hierarchy**

Details

**Fields & Relationships**

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Restriction Rules

Triggers

Flow Triggers

Validation Rules

Sales Service Hierarchy Custom Field  
**Persona (Managed)**  
[Back to Sales Service Hierarchy](#)

 This Custom Field Definition is managed, meaning that you may only edit certain attributes. [Display More Information](#)

[Validation Rules \(0\)](#)

**Custom Field Definition Detail** [Edit](#) [Set Field-Level Security](#) [View Field Accessibility](#) [Where is this used?](#)

**Field Information**

|                           |                                     |             |                                     |
|---------------------------|-------------------------------------|-------------|-------------------------------------|
| Field Label               | Persona                             | Object Name | Sales Service Hierarchy             |
| Field Name                | Persona                             | Data Type   | Picklist (Multi-Select)             |
| Namespace Prefix          | neo_cpo                             |             |                                     |
| API Name                  | neo_cpo__Persona__c                 |             |                                     |
| Description               |                                     |             |                                     |
| Help Text                 |                                     |             |                                     |
| Data Owner                |                                     |             |                                     |
| Field Usage               |                                     |             |                                     |
| Data Sensitivity Level    |                                     |             |                                     |
| Compliance Categorization |                                     |             |                                     |
| Created By                | Marcus Fendelsen, 28.03.2024, 15:34 | Modified By | Marcus Fendelsen, 28.03.2024, 15:34 |

**Package Information**

|                   |             |                       |               |
|-------------------|-------------|-----------------------|---------------|
| Installed Package | SF_CPO_Apco | Available in Versions | 2.1 - Current |
|-------------------|-------------|-----------------------|---------------|

**General Options**

Required ☐

Default Value [i](#)

**Picklist (Multi-Select) Options**

Restrict picklist to the values defined in the value set ☐

Controlling Field [New](#)

# Visible Lines 4

**Picklist Values Used**

|                          |               |
|--------------------------|---------------|
| Active picklist values   | 1 (500 max)   |
| Inactive picklist values | 0 (4,000 max) |

**Validation Rules** [New](#) [Validation Rules Help](#)

No validation rules defined.

**Values** [New](#) [Values Help](#)

| Action               | Values | API Name | Default                  | Modified By                         |
|----------------------|--------|----------|--------------------------|-------------------------------------|
| <a href="#">Edit</a> | Other  | Other    | <input type="checkbox"/> | Marcus Fendelsen, 28.03.2024, 15:34 |

Click New

Setup Home Object Manager

SETUP > OBJECT MANAGER  
**Sales Service Hierarchy**

Details

**Fields & Relationships**

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Add Picklist Values  
**Persona**

Add one or more picklist values below. Each value should be on its own line and it is used for both a value's label and API name.

If a value matches an inactive value's API name, that value is reactivated with its previous label.

If a value matches an inactive value's label but not the API name, a new value is created.

Competitor  
Partner  
Prospect

After you did these two steps you need to populate the persona value to your Sales Service Hierarchy Records. With that you can steer different views.





# NEOMATIC

build your business with NEO experience

Open Card – Level 3 Sales Service Hierarchy record and add needed Personas to the possible picklist values of the field.

Sales Service Hierarchy  
**Card 1**

Related **Details**

\* Name  
Card 1

Active ⓘ  
☒

\* Hierarchy  
Card - Level3 ▼

Is For All Parents ⓘ  
☐

Parent  
Col X

Translation Custom Label API Name ⓘ

**Persona** ↺

Available

Other  
Competitor  
Prospect

Chosen

Partner

## Reading Path Settings

NEO CPO App provides out-of-the box three standard Reading Paths for standard objects:

- Opportunity
- Order
- Asset

All are configured via Custom Metadata Types: **CPO Reading Path Setting**



# NEOMATIC

build your business with NEO experience

 **SETUP**  
**Custom Metadata Types**

### All Custom Metadata Types

[Help for this Page](#)

Custom metadata types enable you to create your own setup objects whose records are metadata rather than data. These are typically used to define application configurations that need to be migrated from one environment to another, or packaged and installed.

Rather than building apps from data records in custom objects or custom settings, you can create custom metadata types and add metadata records, with all the manageability that comes with metadata: package deploy, and upgrade. Querying custom metadata records doesn't count against SOQL limits.

New Custom Metadata Type

| Action                                               | Label                                    | Namespace Prefix | Visibility | API Name                             | Record Size | Description                                                                 |
|------------------------------------------------------|------------------------------------------|------------------|------------|--------------------------------------|-------------|-----------------------------------------------------------------------------|
| <a href="#">Del</a>   <a href="#">Manage Records</a> | <a href="#">CPO Global Setting</a>       | neo_cpo          | Public     | neo_cpo__CPO_GlobalSetting__mdt      | 808         | This metadata type stores global settings for the NEO CPO application       |
| <a href="#">Del</a>   <a href="#">Manage Records</a> | <a href="#">CPO Reading Path Setting</a> | neo_cpo          | Public     | neo_cpo__CPO_ReadingPathSetting__mdt | 356         | This metadata type stores reading path settings for the NEO CPO application |

Click on Manage Records

 **SETUP**  
**Custom Metadata Types**

### CPO Reading Path Settings

View: All [Create New View](#)

| Action                                     | Label                         | CPO Reading Path Setting Name |
|--------------------------------------------|-------------------------------|-------------------------------|
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Opportunities</a> | ReadingPath1                  |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Orders</a>        | ReadingPath2                  |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Assets</a>        | ReadingPath3                  |

## Reading Path 1 - Opportunity

**Data flow:** Opportunity → Opportunity Line Item → Product → Sales Service Hierarchy

## Reading Path 2 - Order

**Data flow:** Order → Order Item → Product → Sales Service Hierarchy

## Reading Path 3 - Asset

Data flow: Asset → Product → Sales Service Hierarchy

with **Warranty Assets object, field Warranty Type** used to render Badges on cards.

Note: Each reading path require that Sales Service Hierarchy records exist and lookup on Products → Sales Service Hierarchy field is assigned

Standard Reading Path Considerations:





# NEOMATIC

build your business with NEO experience

Access to all needed fields need to be granted for the user to render data specified in standard reading paths.

| Standard reading path 1 | Standard reading path 2 | Standard reading path 3 |
|-------------------------|-------------------------|-------------------------|
| Opportunity fields      | Order fields            | Asset fields            |
| AccountId               | AccountId               | AccountId               |
| Amount                  | Name                    | Name                    |
| Description             | OrderNumber             | InstallDate             |
| Name                    | ContractId              | UsageEndDate            |
| NextStep                | TotalAmount             | Price                   |
| StageName               | EffectiveDate           | PurchaseDate            |
| Type                    | Description             | Description             |





# NEOMATIC

build your business with NEO experience

## CPO Card Actions

CPO Card Actions support running:

- Screen Flows
- Auto-lunched Flows (under development and planned to be delivered in next release)

### Screen Flow prerequisites

1. To support CPO Card actions, existing or newly created Screen Flow must have **String type:** recordId variable defined

New Resource

\* Resource Type  
Variable

\* API Name  
recordId

Description

\* Data Type  
Text ☐ Allow multiple values (collection) ⓘ

Default Value  
Enter value or search resources...

Availability Outside the Flow  
☒ Available for input  
☐ Available for output

Cancel Done

which will be filled by CPO App with **record ID** behind CPO Card (link from title of the card)

**Gebäude & Inhalt**

 **Elektronik** 

|                                      |            |
|--------------------------------------|------------|
| 7 Sighilde Erhard                    | 310,62 €   |
| Needs Analysis                       | x 4        |
| Existing Custom...                   | 1.242,49 € |
| Finalize pricing                     | 20 %       |
| Needs Analysis Existing Customer ... |            |

2. CPO App does not come with any Screen Flow therefore to use CPO Card Actions, screen flows must be **created and activated** on org and can be started from CPO App via configured CPO Card Actions.





# NEOMATIC

build your business with NEO experience

3. To run specific flow make sure to grant user access to it

By default, users with any of the following permissions can run this flow.

- Run Flow or Manage Flow in their profile or permission set
- Flow User in their user record

☐ Override default behavior and restrict access to enabled profiles or permission sets.

Recommended is to add all needed flows into CPO Permission Sets: CPO Admin / CPO User

4. Custom metadata configuration

| Custom Metadata Type                 | Field                      |
|--------------------------------------|----------------------------|
| neo_cpo__CPO_ReadingPathSetting__mdt | neo_cpo__CardActionFlow__c |

Card Action Flow

Test\_Flow, Test Flow Label  
FLOW\_API\_NAME\_1, Label 1

Flow\_Api\_Name, Flow label  
(comma separated key, value  
pairs) available as CPO Card  
Quick Action. Separate with new  
line for multiple flows

a. Provide API Names with labels in this field with **coma separated key value pairs**:  
API\_NAME\_1, Label 1

b. For multiple actions, separate coma separated key value pars **with new lines**:

My\_Awsome\_Flow, Label no need to match with real flow label. Can be customized here ANOTHER\_API\_NAME\_2, Any Label here but recommended is to use the same like flow label ONE\_MORE\_API\_NAME\_3

This will activate on all CPO Cards related to the specific Reading Path where the field was populated, dedicated CPO Card Actions.

c. Api Name is **obligatory**.

d. Label is **not obligatory**. If not provided, Api Name will be used only.

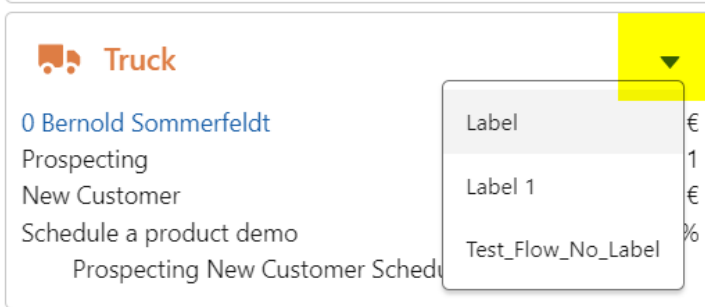


# NEOMATIC

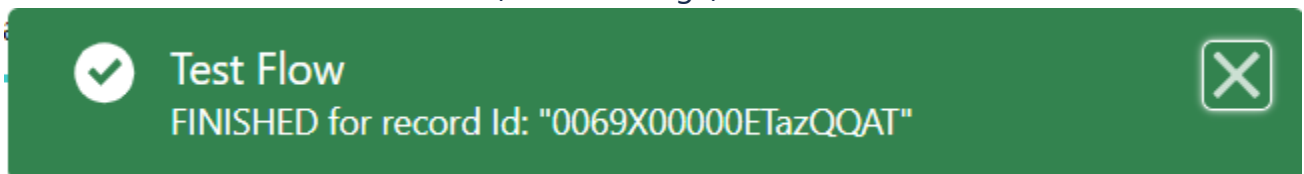
build your business with NEO experience

## 5. Running CPO Card Actions

- Click on the arrow icon in top right of CPO Card and simply select from list an action to run:



- Validate that flow was successful (Toast message)



## 6. Error reporting

Make sure to provide valid **API Names** of the **Active** flows otherwise user will get error

❌ Flow "Test\_Flow\_1" is not found or doesn't have an active version. Contact your admin for more information.

and following Toast Error Message:

❌ Something went wrong with executing flow: "Test\_Flow\_1" for record Id: "0069X00000ETazQQAT" Cannot read properties of undefined (reading 'flow')

Above Toast Error Message will also be shown in case flow was not finished successfully .



# NEOMATIC

build your business with NEO experience

## Enhanced Customization for NEO.CPO

The App does provide a more powerful degree of customization.

By respecting certain aspects you can create a full custom solution for the

- Partner Feature
- Persona Feature
- Reading Paths

### Custom Partner Feature

A fully customized class would require to call some methods from the App. Therefore we are providing an example that uses the Account field Parent for the partner feature:

```
global with sharing class CPO_Partner implements Callable {

    global List<neo_cpo.PartnerWrapper> call(String param, Map<String, Object> args) {
        Set<String> accIds = args.keySet(); // Account Ids to retrieve partners from database
        List<neo_cpo.PartnerWrapper> partnerWrapperList = getData(accIds);
        return partnerWrapperList;
    }

    private List<neo_cpo.PartnerWrapper> getData(Set<String> accIds) {
        //get Accounts (also named as rootAccounts of CPO context from database via bulkified mode
        //List<Account> rootAccounts = [SELECT Id, Name, RecordType.Name, RecordTypeId, ParentId, Parent.Name, Parent.RecordType.Name FROM Account WHERE
        ID IN :accIds]; //-> use this if custom persona feature (with our example) is activated
        List<Account> rootAccounts = [SELECT Id, Name, Type, ParentId, Parent.Name, Parent.Type FROM Account WHERE ID IN :accIds]; //-> else use this

        //the custom CPO Partner example is about to get children and parent of rootAccount

        //get the children of the rootAccounts
        //List<Account> childAccounts = [SELECT Id, Name, RecordType.Name, RecordTypeId, ParentId FROM Account WHERE ParentId IN :accIds]; //-> use this
        if custom persona feature (with our example) is activated
        List<Account> childAccounts = [SELECT Id, Name, Type, ParentId FROM Account WHERE ParentId IN :accIds]; //-> else use this

        List<neo_cpo.PartnerWrapper> pwList = new List<neo_cpo.PartnerWrapper>();

        //loop the rootAccounts to build partner view
        for (Account acc : rootAccounts) {
            neo_cpo.PartnerWrapper pw = new neo_cpo.PartnerWrapper();

            //accFrom is always the root Account of the CPO context
            pw.accFromId = acc.Id;
            pw.accFromName = acc.Name;

            //provide persona information to partnerObj if you use the persona feature - do not provide if you do not use
            //pw.rootPersona = acc.RecordType.Name; //-> use this if custom persona feature (with our example) is activated
            //pw.rootPersona = acc.Type; //-> use this if standard persona feature is activated with field type

            //get children as a partner
            pw.partners = getChildren(acc.Id, childAccounts);

            //add the parent as a partner
            if(acc.ParentId != null) {

                neo_cpo.PartnerWrapper.PartnerObj partnerObj = new neo_cpo.PartnerWrapper.PartnerObj();
                partnerObj.id = acc.ParentId;
                partnerObj.name = acc.Parent.Name;
                partnerObj.role = 'Parent';

                //provide persona information to partnerObj if you use the persona feature - do not provide if you do not use
                //partnerObj.partnerPersona = acc.Parent.RecordType.Name; //-> use this if custom persona feature (with our example) is activated
                //partnerObj.partnerPersona = acc.Parent.Type; //-> use this if standard persona feature is activated with field type

                pw.partners.add(partnerObj);
            }

            pwList.add(pw);
        }
        return pwList;
    }

    private List<neo_cpo.PartnerWrapper.PartnerObj> getChildren(Id rootAccId, List<Account> children) {
```



```

List<neo_cpo.PartnerWrapper.PartnerObj> partnersList = new List<neo_cpo.PartnerWrapper.PartnerObj>();

//loop the children Accounts to find to add where rootAccount is parent
for (Account childAcc : children) {
    if (childAcc.parentId == rootAccId) {
        neo_cpo.PartnerWrapper.PartnerObj partnerObj = new neo_cpo.PartnerWrapper.PartnerObj();
        partnerObj.id = childAcc.Id;
        partnerObj.name = childAcc.Name;
        partnerObj.role = 'Child';

        //provide persona information to partnerObj if you use the persona feature - do not provide if you do not use
        //partnerObj.partnerPersona = childAcc.RecordType.Name; //-> use this if custom persona feature (with our example) is activated
        //partnerObj.partnerPersona = childAcc.Type; //-> use this if standard persona feature is activated with field type

        partnersList.add(partnerObj);
    }
}

return partnersList;
}
}

```

Note: You still need to populate class name in the global settings.

**SETUP** Custom Metadata Types

### CPO Global Setting

**CPO Global Setting Detail**

| CPO Global Setting Name | Global |
|-------------------------|--------|
| Label                   | Global |

▼ Colors

Empty Card Color 000000

▼ Partner View Standard

Partner Object Name Partner

Partner Role Field Role

▼ Partner View Custom

Partner View Class **CPO\_PartnerView**

▼ Persona Feature Standard

Custom Apex class name to overwrite standard Partner View Settings

Personas

## Custom Persona Feature

A fully customized class would require to call some methods from the App. Therefore we are providing an example that uses the Account Record Type to differ CPO dashboard views:

```

global with sharing class CPO_Persona implements Callable {
    global Object call(String param, Map<String, Object> accIds) {
        // param == 'CPO_PERSONA_CUSTOM'
        List<Account> accountsToGetPersonas = [
            SELECT RecordType.Name // <- sample persona field
            FROM Account
            WHERE Id IN :accIds.keySet()
        ];
        List<String> personas = new List<String>();
        for (Account acc : accountsToGetPersonas) {
            personas.add(acc.RecordType.Name);
        }
        return personas;
    }
}

```



# NEOMATIC

build your business with NEO experience

}

Note: You still need to populate class name in the global settings.

Persona Feature Custom

Persona Feature Class

Custom Apex class name to overwrite standard Persona feature  
(leave empty to deactivate Custom Persona globally - SSH - > Persona field is ignored)

## Custom Reading Path

NEO CPO application supports up to **9 Active Reading Paths (RP)**. It is limited via configuration schema **Reading Path** field. Validation will not allow entering number above MAX 9.

Reading Path

Field for defining Reading Path number

Error: Number is too large.

If standard reading paths are not enough, it is possible to configure custom reading paths via custom Apex classes. Both standard and custom RP can work simultaneously.

Custom Reading Path Considerations:

## SOQL

To render date or number fields in NEO CPO App v2 on Cards in user specific locale, use in SOQL query:

```
SELECT
  FORMAT(DateField) AliasForDateField,
  FORMAT(NumberFieldName) AliasForNumberFieldName,
  ...
```

[https://developer.salesforce.com/docs/atlas.en-us.soql\\_sosl.meta/soql\\_sosl/sforce\\_api\\_calls\\_soql\\_select\\_format.htm](https://developer.salesforce.com/docs/atlas.en-us.soql_sosl.meta/soql_sosl/sforce_api_calls_soql_select_format.htm)

In Organizations with multicurrency, to render currency field in NEO CPO App v2 on Cards in user specific locale, use in SOQL query.

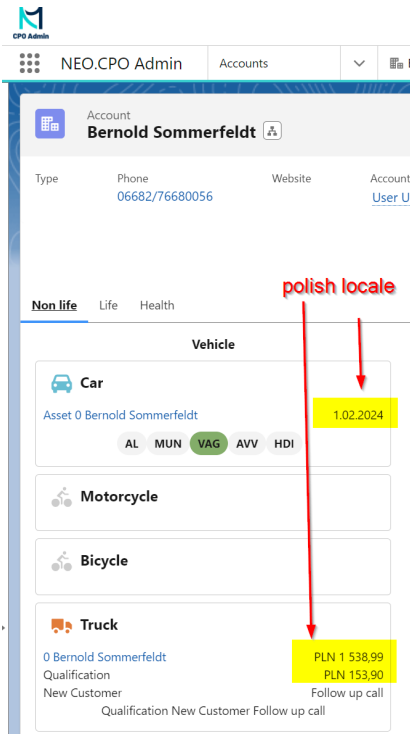
```
SELECT
  FORMAT(convertCurrency(CurrencyField)) AliasForCurrencyField,
  ...
```

[https://developer.salesforce.com/docs/atlas.en-us.soql\\_sosl.meta/soql\\_sosl/sforce\\_api\\_calls\\_soql\\_querying\\_currency\\_fields.htm](https://developer.salesforce.com/docs/atlas.en-us.soql_sosl.meta/soql_sosl/sforce_api_calls_soql_querying_currency_fields.htm)



# NEOMATIC

build your business with NEO experience



For a custom reading path you need to consider to call some App methods. Therefore we are providing a class example for a custom reading path with mocked data.

```
global class CPO_CustomRP implements Callable {
    public static final String VALID_PATH_NUMBER = '4', VALID_ROOT_OBJ_NAME = 'Account';
    private static final Set<Id> accIds = new Set<Id>();
    private static final Map<Id, neo_cpo__Sales_Service_Hierarchy__c> sshById = new Map<Id, neo_cpo__Sales_Service_Hierarchy__c>();

    public Object call(String pathNumber, Map<String, Object> args) {
        neo_cpo.ReadingPathHelper.validate(pathNumber, args, VALID_PATH_NUMBER, VALID_ROOT_OBJ_NAME);
        neo_cpo.ReadingPathHelper.extractArguments(args, VALID_ROOT_OBJ_NAME, accIds, sshById); //validated: accIds (rootAccount + partnerAccount) &
        sshById properties filled here
        return getCards();
    }

    private static List<neo_cpo.CpoWrapper.Card> getCards( ) {
        List<Case> data = getData();
        List<neo_cpo.CpoWrapper.Card> cards = new List<neo_cpo.CpoWrapper.Card>();
        for (Case cs : data) {
            cards.add(buildCard(cs.AccountId, cs));
        }
        return cards;
    }

    private static List<SObject> getData( ) {
        // Mocked Case data here for demo purpose but should be retrieved from org for all 'accIds'
        // both standard and custom objects are supported

        List<Case> data = new List<Case>();

        for ( id accId : accIds) {

            for (neo_cpo__Sales_Service_Hierarchy__c crdSsh : neo_cpo.UtilSelector.cardsLevel3Sshs.values()) {
                Boolean rnd = Math.mod(Integer.valueOf(Math.floor(Math.random() * 10)), 6) == 0;
                if (rnd) {
                    Case newCase = new Case(AccountId = accId, Description = crdSsh.Name, SuppliedName = crdSsh.Id, Comments = 'Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam');
                    data.add(newCase);
                    data.add(newCase.clone());
                }
            }
        }
    }
}
```



# NEOMATIC

build your business with NEO experience

```
}
return data;
}

private static neo_cpo.CpoWrapper.Card buildCard(Id accId, SObject sObj) {
    // specify product - cardSsh lookup
    String crdSshId = (String) sObj?.get('SuppliedName');
    // use package method to build initial card as a mandatory step
    neo_cpo.CpoWrapper.Card card = neo_cpo.ReadingPathHelper.buildInitialCard(accId, crdSshId, sshById, Integer.valueOf(VALID_PATH_NUMBER));

    // fill up to 1-12 custom fields on card:
    card.custom1 = new neo_cpo.CpoWrapper.CustomField('Account ' + accId, 'Account', (String) accId); // -> use package CustomField constructor to
    build an internal link with (label, objectName, id)
    card.custom2 = (String) sObj?.get('Description');
    card.custom3 = 'Custom 3';
    card.custom4 = 'Custom 4';
    card.custom5 = new neo_cpo.CpoWrapper.CustomField('Google Link', 'http://www.google.de'); // -> use package CustomField constructor to build an
    external link with (label, url)
    card.custom6 = 'Custom 6';
    card.custom7 = 'Custom 7';
    card.custom8 = 'Custom 8';
    card.custom9 = (String) sObj?.get('Comments');
    card.custom10 = 'Custom 10';
    card.custom11 = 'Custom 11';
    card.custom12 = 'Custom 12';

    fillBadgesForCard(card);
    return card;
}

private static void fillBadgesForCard(neo_cpo.CpoWrapper.Card card) {
    if (card?.badges == null || card.badges.isEmpty()) {
        return;
    }
    Set<String> sshNames = neo_cpo.UtilSelector.badgesLevel4SshsByName.keySet();
    activateBadges(card);
}

private static void activateBadges(neo_cpo.CpoWrapper.Card card) {
    integer counter = 0;
    for (neo_cpo.CpoWrapper.Badge badge : card.badges) {
        counter++;
        if (counter > 1) {
            badge.active = true;
            badge.description = 'Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore
            magna aliquyam';
        }
    }
}
}
```

Note: You still need to create a new record in custom metadata settings.

The screenshot shows the Neomatic Setup interface. The top navigation bar includes a search bar and various utility icons. The left sidebar has a search bar and a menu with 'Setup', 'Home', and 'Object Manager'. The main content area is titled 'Custom Metadata Types' and displays 'CPO Reading Path Settings'. Below this, there is a table with columns for 'Action', 'Label', 'CPO Reading Path Setting Name', and 'Namespace Prefix'. The table lists three settings: 'Assets', 'Opportunities', and 'Orders', all with the namespace 'neo\_cpo'. A 'New' button is located above the table. A message at the bottom left states: 'Didn't find what you're looking for? Try using Global Search.'

| Action | Label         | CPO Reading Path Setting Name | Namespace Prefix |
|--------|---------------|-------------------------------|------------------|
| Edit   | Assets        | ReadingPath3                  | neo_cpo          |
| Edit   | Opportunities | ReadingPath1                  | neo_cpo          |
| Edit   | Orders        | ReadingPath2                  | neo_cpo          |

Click New



# NEOMATIC

build your business with NEO experience

### CPO Reading Path Setting

Help for this Page

CPO Reading Path Setting Edit

Save Save & New Cancel

Information

= Required Information

Label

CPO Reading Path Setting Name

Root Object Account

Reading Path 0

Color 000000

Protected Component

Is Active

Class Name

Badges Settings

Badge Fill Color 42d7f5

Badge Border Color 42d7f5

Badge Text Color 000000

Badge Active

Save Save & New Cancel

All mandatory fields need to be populated with information based on the class that you have created.

