

UX Plugins User Manual
Oracle Banking Digital Experience
Patchset Release 22.2.5.0.0

Part No. F72987-01

October 2024

ORACLE®

UX Plugins User Manual

October 2024

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax: +91 22 6718 3001

www.oracle.com/financialservices/

Copyright © 2006, 2024, Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Table of Contents

1. Preface	1-1
1.1 Intended Audience.....	1-1
1.2 Documentation Accessibility.....	1-1
1.3 Access to Oracle Support.....	1-1
1.4 Structure	1-1
1.5 Related Information Sources	1-1
2. The UX Plugin.....	2-1
3. Plugin Setup	3-3
4. List of Components	4-5
5. List of Templates	5-10
6. Create New	6-16
7. Generate From Existing	7-18
7.1 Convert to Transaction	7-18
7.2 Add Grid.....	7-21
7.3 Add Conditions	7-24
7.4 Add Logic.....	7-30
8. Export to Toolkit	8-32
8.1 Individual Page	8-34
8.2 Transaction	8-37
8.3 Widget.....	8-38
8.4 After completing configurations	8-41

1. Preface

1.1 Intended Audience

This document is intended for the following audience:

- Customers
- Partners

1.2 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

1.3 Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

1.4 Structure

This manual is organized into the following categories:

Preface gives information on the intended audience. It also describes the overall structure of the User Manual.

Introduction provides brief information on the overall functionality covered in the User Manual.

The subsequent chapters provide information on transactions covered in the User Manual.

Each transaction is explained in the following manner:

- Introduction to the transaction
- Screenshots of the transaction
- The images of screens used in this user manual are for illustrative purpose only, to provide improved understanding of the functionality; actual screens that appear in the application may vary based on selected browser, theme, and mobile devices.
- Procedure containing steps to complete the transaction- The mandatory and conditional fields of the transaction are explained in the procedure. If a transaction contains multiple procedures, each procedure is explained. If some functionality is present in many transactions, this functionality is explained separately.

1.5 Related Information Sources

For more information on Oracle Banking Digital Experience Patchset Release 22.2.5.0.0, refer to the following documents:

- Oracle Banking Digital Experience Licensing Guide
- Oracle Banking Digital Experience Installation Manuals

2. The UX Plugin

The UX plugin is a Figma-based tool designed to facilitate the seamless conversion of Figma designs into Oracle JET-compliant code. This plugin helps ensure consistency between design and development by eliminating discrepancies between the visual design and the final product.

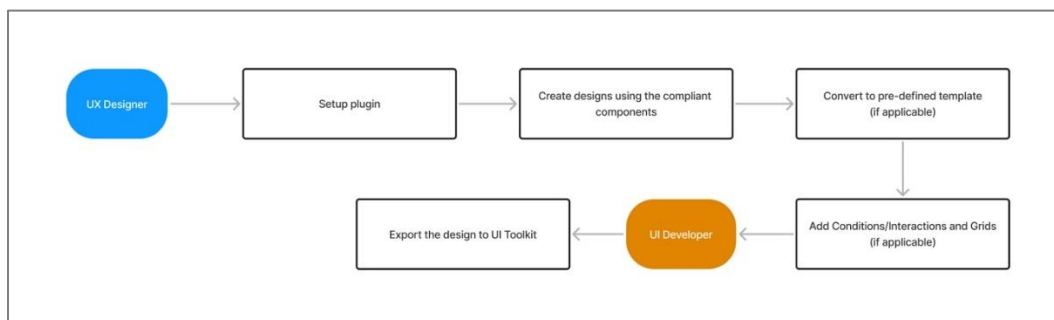
By automating this translation process, the tool significantly reduces the time and effort required to develop user interfaces, accelerating the overall UI implementation.

Pre-requisites:

- Basic understanding of Figma
- Basic understanding of Oracle's Redwood Design System
- Basic understanding of Oracle JET
- Basic understanding of the UI Toolkit
- An active Figma license
- UI Toolkit installed and running on the system (latest version)

Workflow

The process to create a screen and export the design in the UI toolkit is as follows:



Step 1: Setup Plugin

The plugin can be easily setup on your system by following the instructions outlined in the **'Plugin Setup'** section. Once done you can run the plugin and start with your design journey.

Step 2: Create designs using compliant components

Designers can either drag and drop the compliant components to create their designs on Figma or can explore the **'Create New'** (link to the Create New feature to be provided here) feature of the plugin to get started.

Step 3: Convert to pre-defined template

In case if the designer wants to structure their designs into a pre-defined template, they can do so using the **'Generate from Existing'** (link to the Generate from Existing feature to be provided here)

Step 4: Add Conditions/Interactions

In case the designer wants to add an **'On Click'** or **'On Change'** functionality on components they can do so by either using the Figma's prototyping feature or using the **'Add Conditions'** feature available in the **'Generate from Existing'** option.

Step 5: Export to UI Toolkit

At this stage the UX Designer hands over the design to the UI Developer, who will then take over the procedure to export this design into the toolkit. The UI developer will have to ensure the toolkit is up and running in the background before starting the export procedure.

Note: The workflow outlined above provides an overview of the plugin usage.

A detailed explanation of each feature is provided in the sections below.

Usage Guidelines

- Use components exclusively from the Oracle JET or Redwood Library
- Ensure that the UI toolkit is successfully installed with the latest version
- Ensure that the Figma components are not detached as those components may not be recognised by the plugin
- Create business components only if they will be reused multiple times across your screens. Avoid creating excessive business components to maintain efficiency and manageability.
- Ensure that the UI toolkit is up and running in the background while exporting to avoid any issues.
- For optimal plugin functioning, keep the original layer names of components unchanged, as modifying them may result in recognition errors.

3. Plugin Setup

1. Install the latest version of [Node.js](#) supported for your device specifications, before downloading the plugin zip file.

(Note: This is a one-time step, need not to be repeated.)

2. Download the zip file and extract it in the desired destination folder.
3. Upon extraction, there will be two folders:

- a. Backend:

- i. Open the command terminal and navigate to the Backend Folder.
- ii. In the terminal, enter the following command:

npm install

(For Mac Users, append 'sudo' before the command')

- iii. Now, enter the following command in the same terminal:

npm run build

- b. UI:

- i. Open the command terminal and navigate to the UI Folder.
- ii. In the terminal, enter the following command:

npm install

(For Mac Users, append 'sudo' before the command')

- iii. Now, enter the following command in the same terminal:

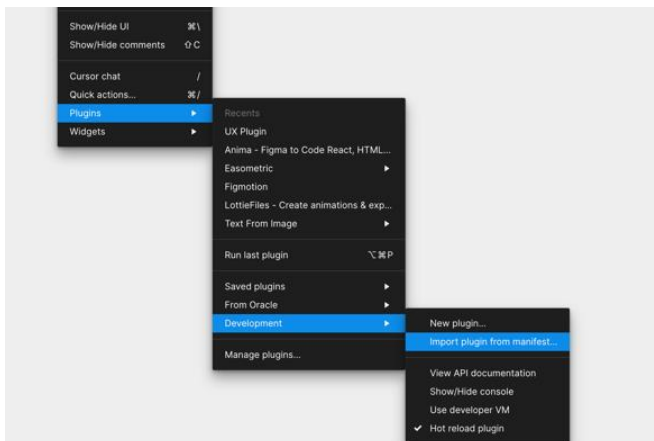
npm link './Backend/'

Note: The './Backend/' is the path to the Backend folder.

- iv. Enter the following command to build the UI of the plugin:

ojet build

4. Move to Figma, right click on any page, Plugin > Development > Import plugin from manifest



5. Navigate to the UI folder, and click on the 'manifest.json' file inside the UI folder, that needs to be imported.
6. The plugin is successfully imported inside Figma and is ready to use.

4. List of Components

The plugin supports a range of components that streamline the design and development process. These include components from Oracle's Redwood Design System (RDS) that are directly compatible, as well as custom components created to meet specific business requirements.

Below is the list of all RDS supported components:

1. Avatar
2. Badge
3. Button
4. Checkbox set
5. File Picker
6. Input Date
7. Input Number
8. Input Password
9. Input Select Single (supports Figma interaction in case of static dropdown values)
10. Input Text
11. Input Text Area
12. Meter Bar
13. Radio Button
14. Rating Gauge
15. Slider
16. Switch
17. Tab Bar
18. Table
19. Toggle Button

Following are the components created to cater to specific needs:

1. **Account Input**

This is a custom input field created specifically to select an account from a list of accounts fetched from a pre-configured API. It has two display modes: dropdown view and card view.

Following are the properties of this component:

- Display: Dropdown, Card
- Label: Text Input
- Read Only: Boolean
- Hide Label: Boolean
- Value: Boolean

2. Action Card (JET Version)

An Action Card is an actionable container rendering related information. An action card renders children in a styled rectangular area and has support for an action event when clicked.

It has a single property of instance swap that allows the designers to place their content in the body slot of the card.

[Link to JET Component](#)

3. Address

The Address component is designed for entering address information within a form. It includes two elements: address input and address selection.

Following are the properties of this component:

- Read Only – Boolean
- Type – Saved Address, Branch Near Me, Custom
- Address Enabled – Boolean

4. Amount Input

The Amount Input component is specifically designed for entering monetary values in a form. It supports input fields for currency formatting and precise value entry.

Following are the properties of this component:

- Label Hidden – Boolean
- Read Only – Boolean
- Currency List – Boolean
- Mandatory – Boolean
- Label – Text Input
- View Limits – Boolean

5. Collapsible (JET Version)

A collapsible displays a header that can be expanded to show its content.

Following are the properties of this component:

- Variant – Basic, Horizontal Rule
- Header – Instance Swap
- Body – Instance Swap

[Link to JET Component](#)

6. Column Share

The Column Share component is designed to allocate and display data in a column-based format. It enables the division of content into proportional sections for better layout and organization.

Following are the properties of this component:

- Columns – 2,3
- Field Size – 25-75, Equal, 40-60, 75-25, 60-40

Disclaimer: To create more than 3 column layout the designer can combine two form layouts and achieve the desired column structure.

7. Drawer (JET Version)

A Drawer Popup is a panel that slides into the viewport. It can be placed at the 'start', 'end' or '**bottom**' edge and it always overlays the page.

Following are the properties of this component:

- Edge – Start, End
- Body – Instance Swap
- Header – Boolean
- Header Text – Text Input
- Sub Header – Boolean
- Sub Header Text – Text Input
- Hide Close Icon – Boolean
- Show Footer – Boolean

[Link to JET Component](#)

8. Icon

This component was created specifically to identify the correct icon from the Redwood icon repository. It is recommended that designers use this component when incorporating icons independently.

It has only one property of instance swap for selecting the correct icon.

9. Internal Account Input

The Internal Account Input component is designed for entering account numbers in a format commonly used in the banking domain. It consists of two input fields: a primary field for the account number and a secondary field for confirming the account number.

Following are the properties of this component:

- Required – Boolean
- Mask – Boolean
- State – Default, Filled
- Read Only – Boolean
- Confirm Style Account – Boolean (for the secondary account number field)
- Confirm Label – Text Input
- Label – Text Input

10. Link

The Link component is used to create clickable text elements that navigate users to external URLs, internal pages, or trigger specific actions within the application.

Following are the properties of this component:

- State – Default, Primary, Secondary, Disabled
- Type – Standalone, Embedded
- Label – Text Input

Disclaimer: Use Figma's native prototyping feature for links that trigger components within the same page. For links that redirect to separate pages or external locations, handle them using the UI Toolkit.

[Link to JET Component](#)

11. List Item Layout (JET Version)

A List Item Layout represents layout used for list view item elements. To create a list view combine all list items and add them in a frame.

Following are the properties of this component:

- Selector – Boolean & Instance Swap
- Leading – Boolean & Instance Swap
- Default – Instance Swap
- Secondary – Boolean & Instance Swap
- Tertiary – Boolean & Instance Swap
- Overline – Boolean & Instance Swap
- Metadata – Boolean & Instance Swap
- Trailing – Boolean & Instance Swap
- Action – Boolean & Instance Swap
- Bottom Slot – Boolean
 - Quaternary – Boolean & Instance Swap
 - Navigation – Boolean & Instance Swap

Disclaimer: Designers are recommended to use this component to make list views as this has been re-created specifically for the purpose of making it more compatible with the plugin.

[Link to JET Component](#)

12. Modal Dialog (JET Version)

A dialog displays a popup window that provides information and gathers input from the application user. Modal dialogs require interaction before control can be returned to the outer window.

Following are the properties of this component:

- Hide Header – Boolean

- Body – Instance Swap
- Close Button – Boolean
- Footer – 3 buttons, 2 buttons, 1 button

[Link to JET Component](#)

13. Page Section

The "Page Section" component creates distinct sections on a page, allowing for clear organization and layout. It is especially useful for structuring forms, such as those needed for transaction screens.

Following are the properties of this component:

- Page Heading – Boolean
- Header Template – Boolean (for hiding the controls present on the header)

14. Quick Action Button

The "Quick Action Button" provides a shortcut to specific tasks or actions related to particular modules or transactions, streamlining user interaction. It consists of an avatar, primary text and a custom slot.

Following are the properties of this component:

- Primary Text – Text Input
- Custom Slot – Boolean & Instance Swap
- Avatar – Dedicated Component Properties (defined by Redwood)

15. Row

The "Row" component displays read-only data in a label-value format, allowing designers to customize how the information is presented.

Following are the properties of this component:

- Type – Primary, Secondary, Disabled, Success, Warning, Danger
- Size – 2XS, XS, SM, MD, LG, XL
- Label – Text Input
- Value – Text Input

16. Text

The "Text" component defines typography styles and tags, helping designers maintain consistency in text hierarchy and application throughout the design.

Following are the properties of this component:

- Type – H1, H2, H3, H4, H5, Label, Custom
- Text input for each of the above types

Disclaimer: The plugin reads the typography styles defined in Figma only if 'Custom' type is used. For the rest, it follows the standards as defined in the application framework.

5. List of Templates

UI Templates are pre-designed layouts that provide a structured framework for creating user interfaces. They help streamline the design process by offering standardized, reusable components and styles, ensuring consistency and efficiency in the development of applications.

Predefined templates of OBDX Application that are supported by the plugin include:

- Transaction Template
- Widgets

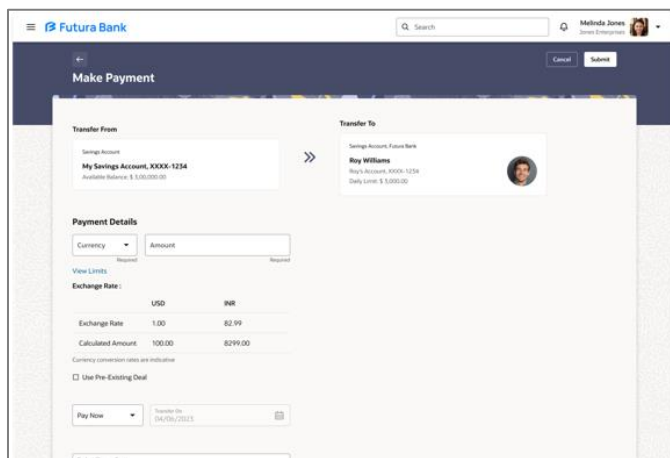
Transaction Template

A streamlined three-step process designed for accuracy and ease of use, consisting of an initiation screen, a review screen, and a confirmation screen.



Initiation Screen:

The starting point of the process, where users input or select initial information to begin the workflow.



Following are the properties of the initiation page:

- Title – Text Input
- Context Switcher – Boolean
- Show Subtitle – Boolean
- Action Slot – Boolean
 - Secondary Button Type – Default, Menu Button
 - Primary Button – Boolean
 - Secondary Button – Boolean
 - Tertiary Button – Boolean
- Footer Tabs – Boolean

How to create Transaction template using the UX Plugin:

- a. Use the '**Create New**' feature (link to Create New section) to create a series of page sections and make the desired changes to the input fields.
- b. Once the form is ready, convert the page sections into the transaction template, using '**Generate from Existing – Transaction**' feature (link to Generate from Existing – Transaction section)
- c. Make the necessary changes to the template and proceed towards exporting the template (link to Export – Transaction) section

Widgets

The dashboard page enables users to create and display widgets, offering graphical representations of information for quick insights and a personalized experience.



Following are the properties of widgets:

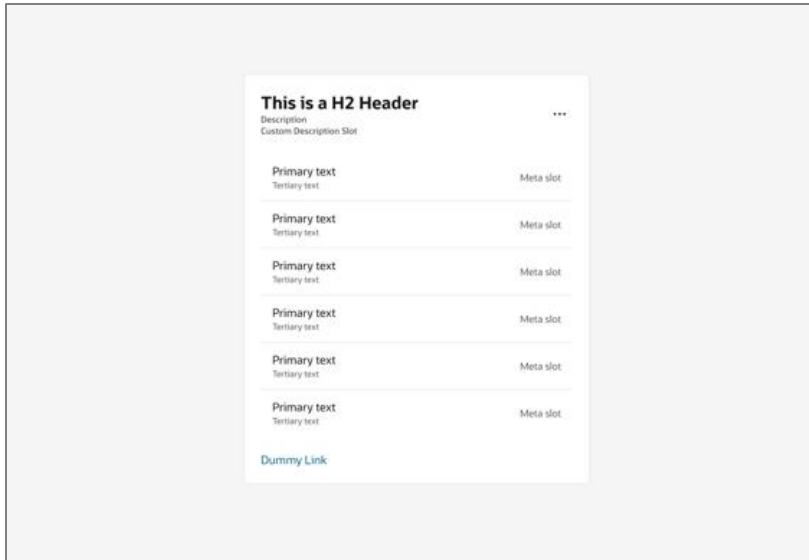
- Full Body – Boolean (a property that eliminates the header and footer)
- Body – List View, Chart, Custom
- Body Slot – Instance Swap (applicable on from '**Custom**' Body)
- Heading – Boolean & Text Input
- Description – Boolean & Text Input
- Action Slot – Boolean & Instance Swap
- Custom Description Slot – Boolean & Instance Swap
- Footer – Boolean & Instance Swap

How to create widgets using the UX Plugin:

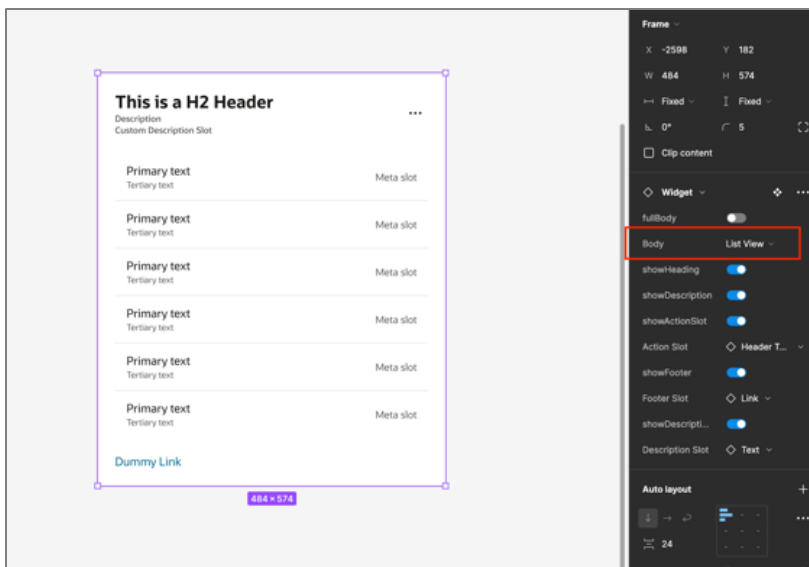
The widget component provides three body types: List View, Chart and Custom.

For List View and Chart

- Place the widget component from the asset library on the design board

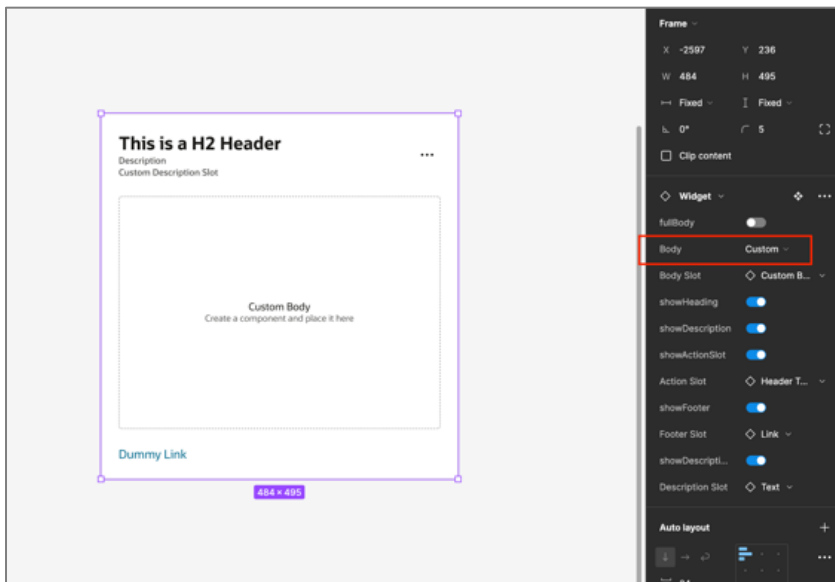


- Select the required body display from the properties and proceed towards exporting the widget (link to Export – Widget)

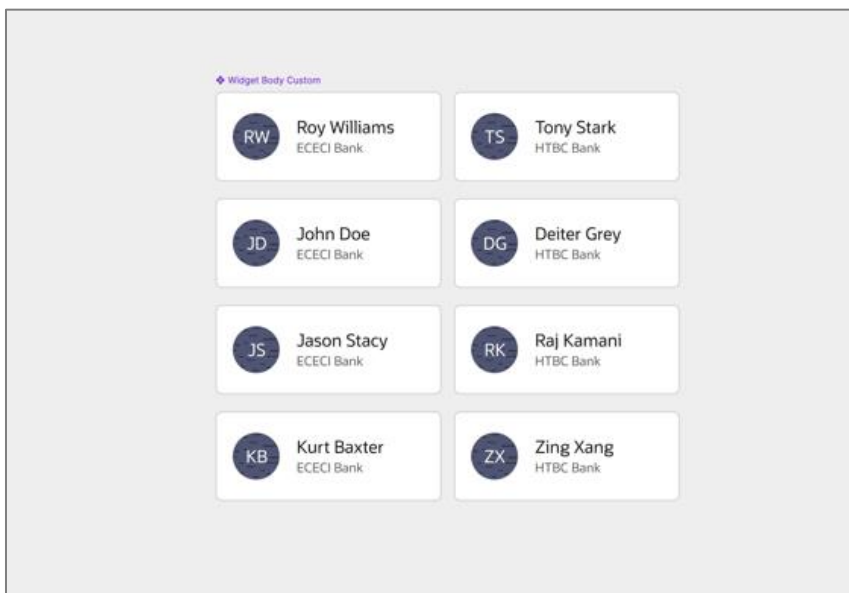


For Custom Widget

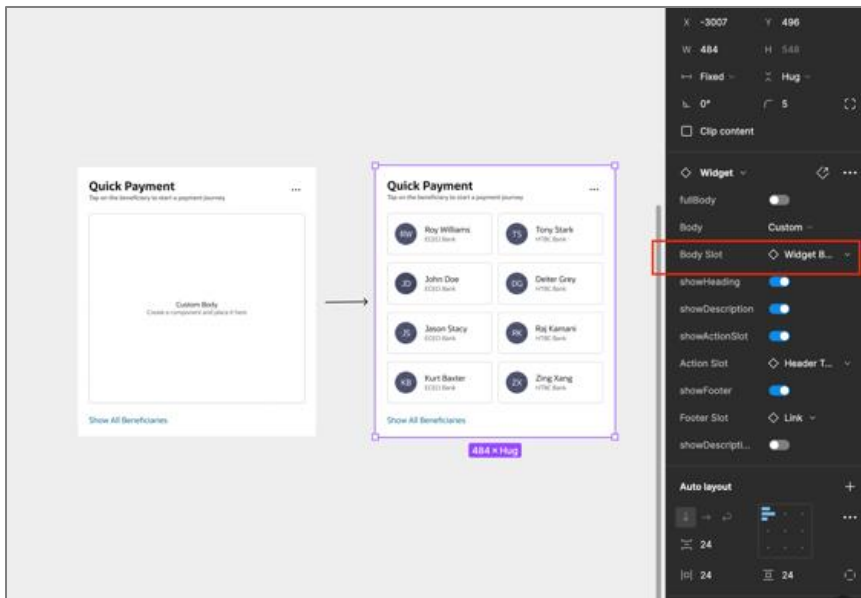
- Place the widget component from the asset library on the design board and choose the **“Custom”** body property



- Design the custom body using the supported components and place them in a single frame. Convert that frame into a component.



- c. Once the component is ready, swap that component with the dummy custom body component using the Instance Swap property of the body slot.



- d. Once the widget is ready proceed towards exporting the widget (link to Export – Widget).

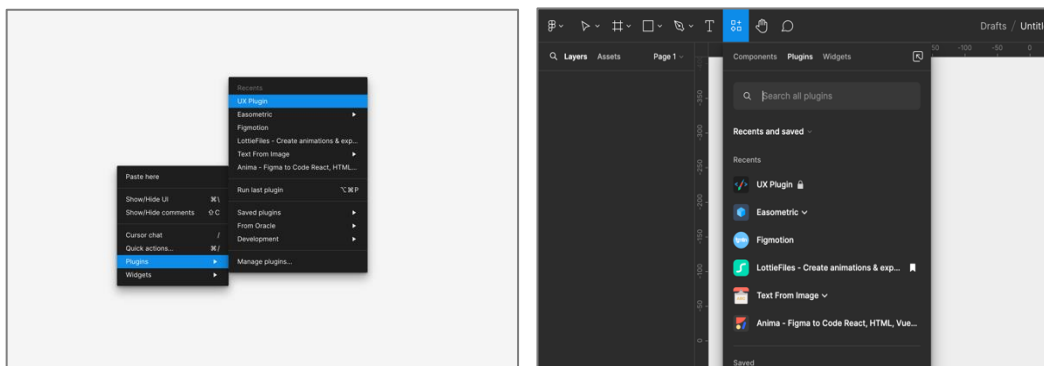
6. Create New

This feature allows for the automatic creation of design components in Figma based on predefined specifications. It streamlines the early stages of the design process by providing standardized elements, enabling designers to efficiently begin their work with consistent and compliant components.

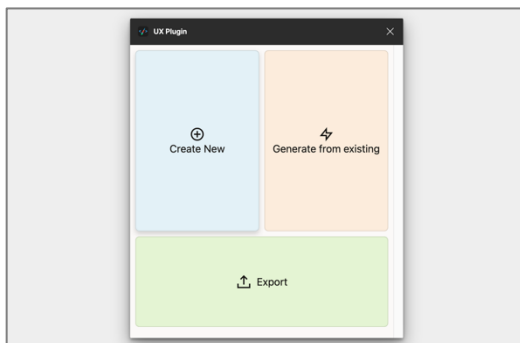
For example:

To generate a series of 'Page Sections' to start the journey of designing a form:

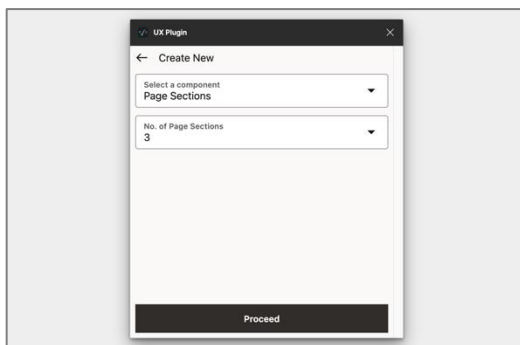
- a. Run the plugin by right clicking on the screen > Plugins > Locate the **'UX Plugin'** or find your recent plugins through the **'Resources'** icon located on the toolbar



- b. Once the plugin window appears, click on the option **'Create New'**.



- c. Select the **'Page Section'** component and specify the number of page sections required.



- d. When clicked on **'Proceed'** the plugin will place the desired series of page sections on the design board. The designer can then start replacing the input components and remove the unrequired ones.

The screenshot displays a design board with a central component titled "TransactionForm". This form is organized into three distinct sections. Each section begins with a header labeled "This is a H2 Header", followed by four input fields, each labeled "Label". The sections are stacked vertically, and each section's header and input fields are enclosed in a light gray border. The entire form is set against a light gray background.

7. Generate From Existing

This feature enables designers to refine and enhance existing components, simplifying the process of adjusting components to meet specific design and interaction requirements efficiently.

This feature has 3 capabilities:

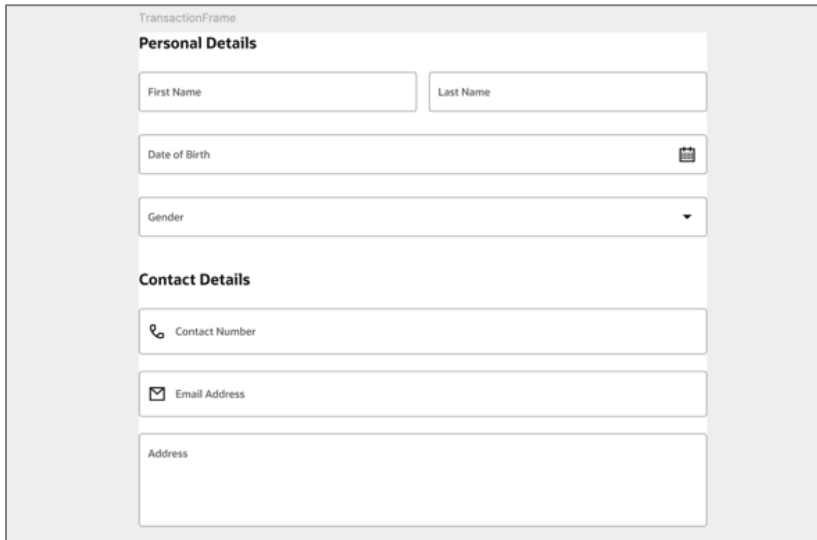
- Convert to Transaction
- Add Grid
- Add Conditions

7.1 Convert to Transaction

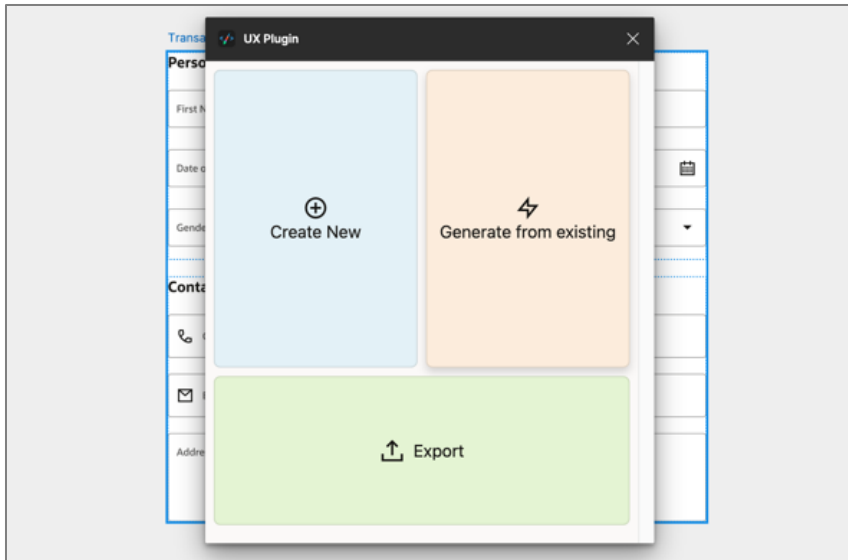
This capability converts a series of pages sections into a 3-step transaction template. For more details about the Transaction Template refer to the Templates section (provide link to the transaction template section).

For example:

To export the below user details form, the designer should ensure:



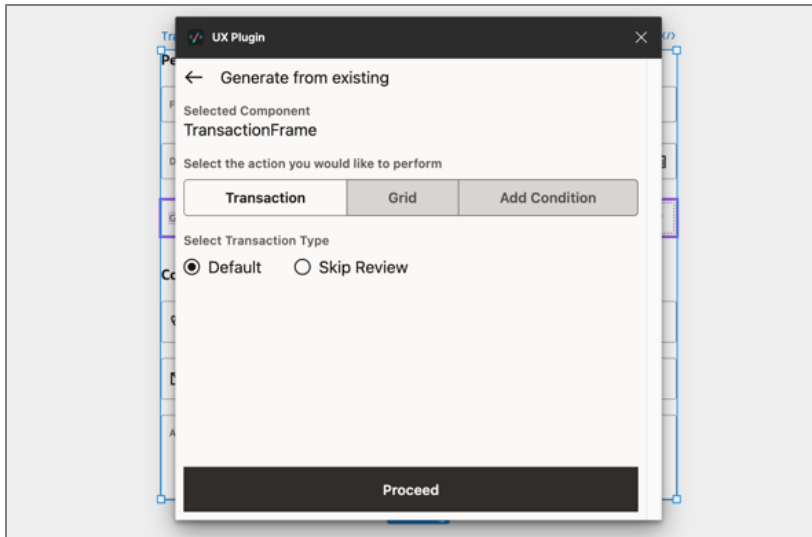
- Once the form that contains the series of page sections is ready, run the plugin and select the **'Generate from Existing'** option.



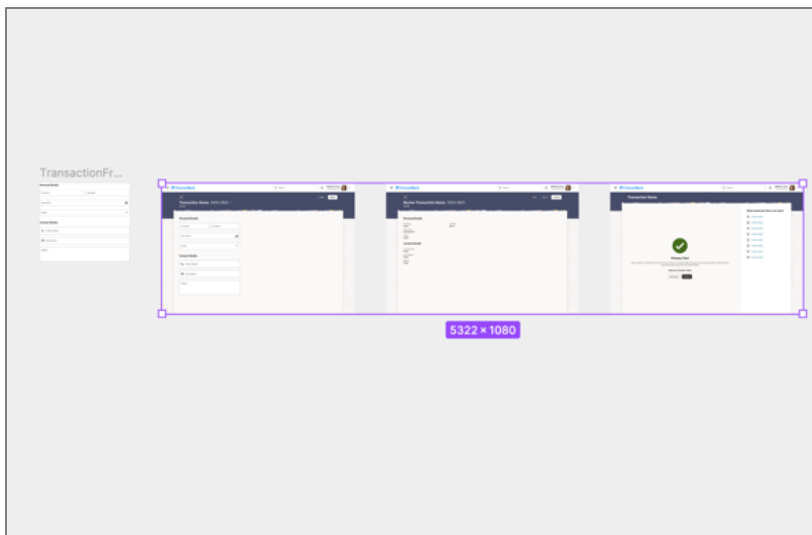
- b. Select the frame that contains the series of page sections as prompted on the plugin window.



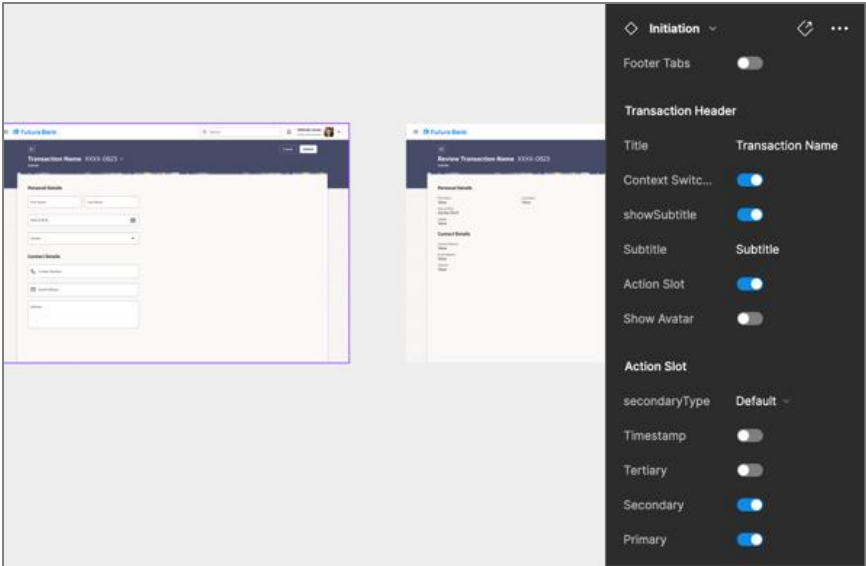
- c. After selecting the frame, choose the **'Transaction'** option, specify the type of Review type needed and click on **'Proceed'**.



- d. After clicking on '**Proceed**' the plugin will generate the place the form within the '**Initiation**' page of the template, auto generating the review and confirmation page.



- e. Once the transaction template is generated, the designer can change to the template such as updating the page title, subtitle etc. and begin the export journey.



7.2 Add Grid

This feature allows the designers to ensure the correct grid structure will be implemented to the designed components when the screens are exported into the UI toolkit.

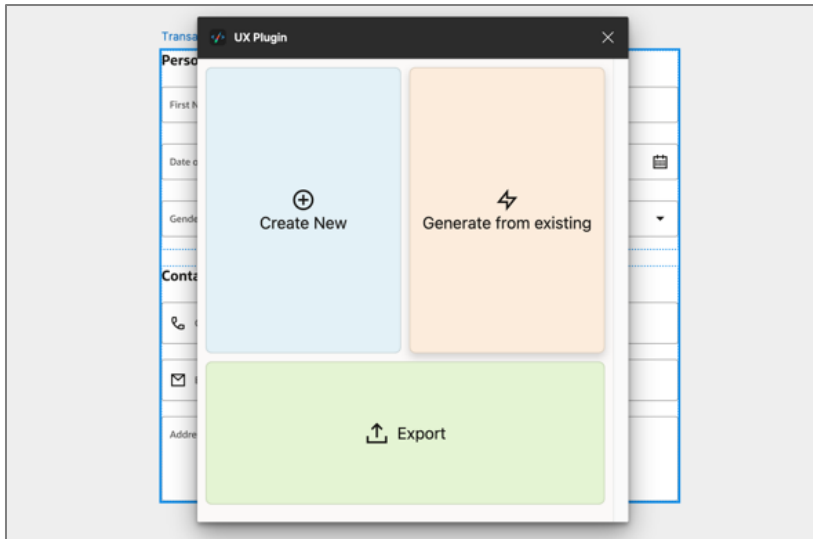
For example:

If designers want to ensure the grid specification of the below design is translated identically,



they should:

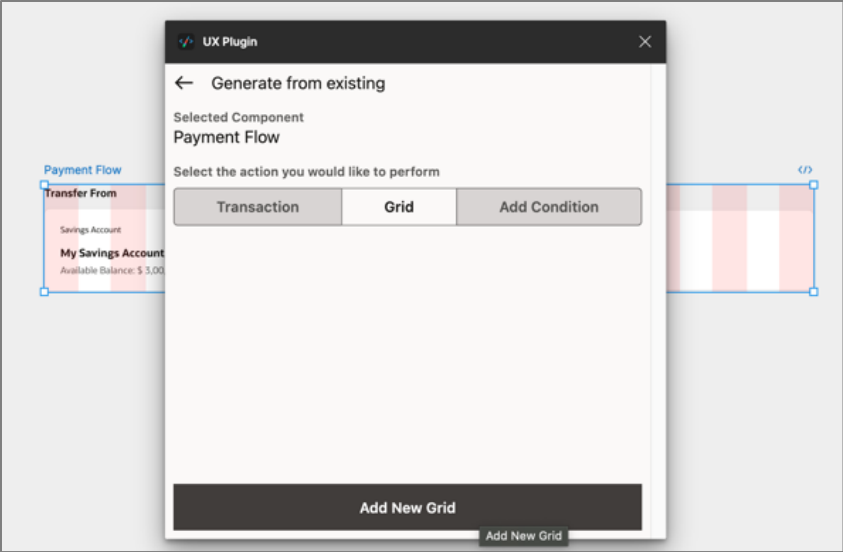
- Run the plugin and choose the **'Generate from Existing'** option.



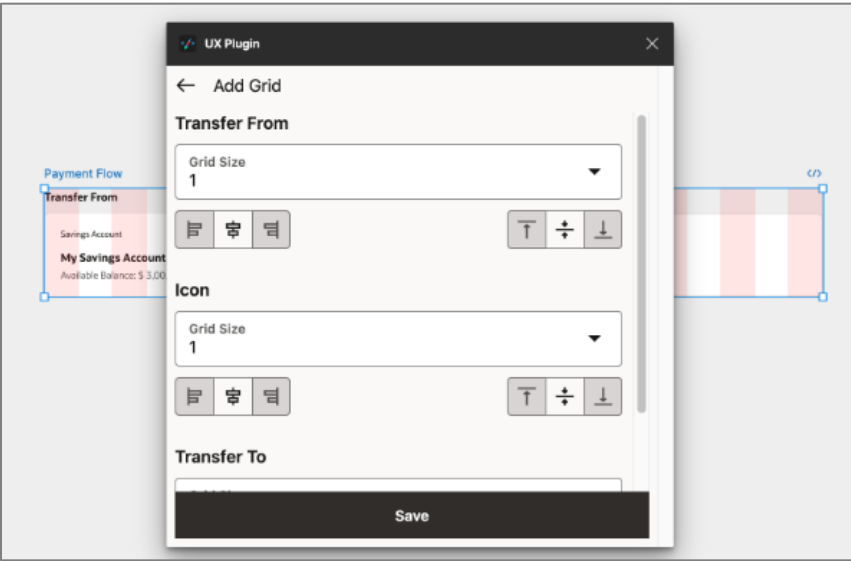
- b. Select the frame on which the grid specifications need to be added.



- c. After selecting the frame, select the '**Grid**' option and click on the '**Add New Grid**' button at the bottom of the window.



- d. On the 'Add Grid' page, the plugin will automatically detect and list the child components of selected frame. The designer just needs to specify the grid size, its alignment and click 'Save' to save the specifications.



7.3 Add Conditions

This capability allows designers to define complex interaction handling for supported components. It establishes a relationship between two components:

- **Variable component:** The dependent component that changes based on input from the controlling component.
- **Controlling component:** The component that determines what is displayed or how the variable component behaves.

This feature simplifies managing interactions within a particular component ensuring dynamic design behavior.

This capability has two use cases:

- Show/Hide of variable component based on the explicit value, state or selection of the controlling component
- Show/Hide of variable component based on the defined properties of the controlling component

Disclaimers:

> For cases where there are multiple variable components controlled by a single controlling component, place all the variable components in a single frame.

> For cases where the interaction leads an external component, please use Figma's native prototyping feature.

Case 1: Show/Hide of variable component based on the explicit value, state or selection of the controlling component

For example:

To hide and show the relevant fields on the click of the below radio button:

The image displays two variations of a form titled "Investment Period".

Top Variation: The "Specify Tenure" radio button is selected. Below it are two input fields: "Years" and "Months". Both fields are marked as "Required".

Bottom Variation: The "Specify End Date" radio button is selected. Below it is a single input field labeled "End Date" with a calendar icon to its right. This field is also marked as "Required".

- a. Ensure that all the components are present in one single frame.

Investment Period

Investment Period

☐ Specify Tenure ☐ Specify End Date

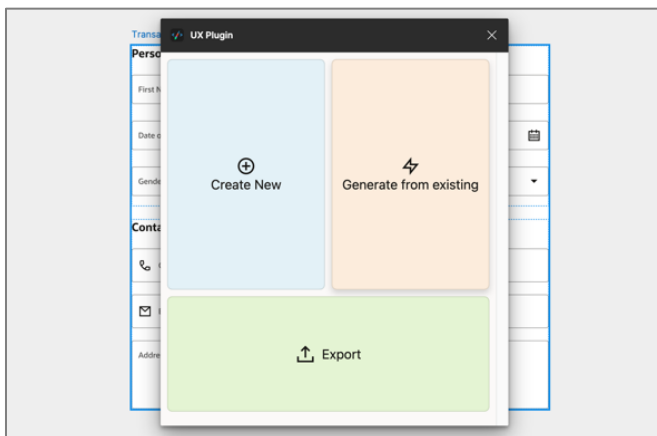
Years Months

Required Required

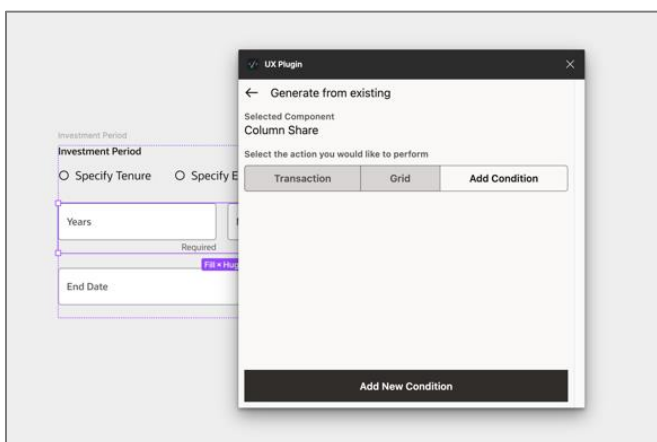
End Date

Required

- b. Select the designed frame, run the plugin and select the **'Generate from Existing'** option.



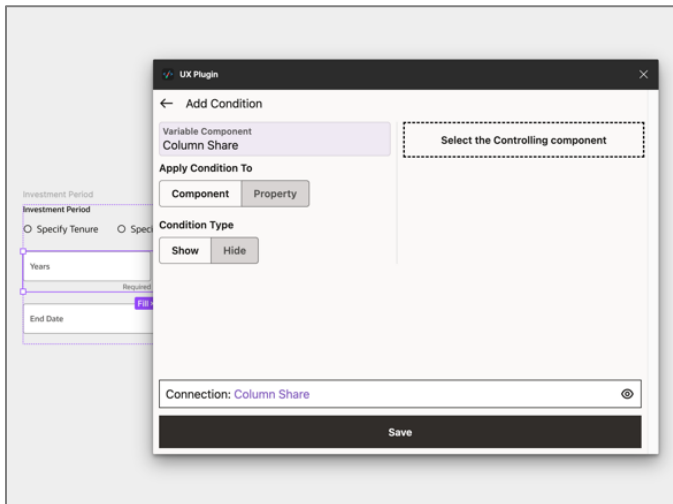
- c. Select your variable component (in this case, the Input fields), choose the **'Add Condition'** option and click on the **'Add New Condition'** button at the bottom of the window.



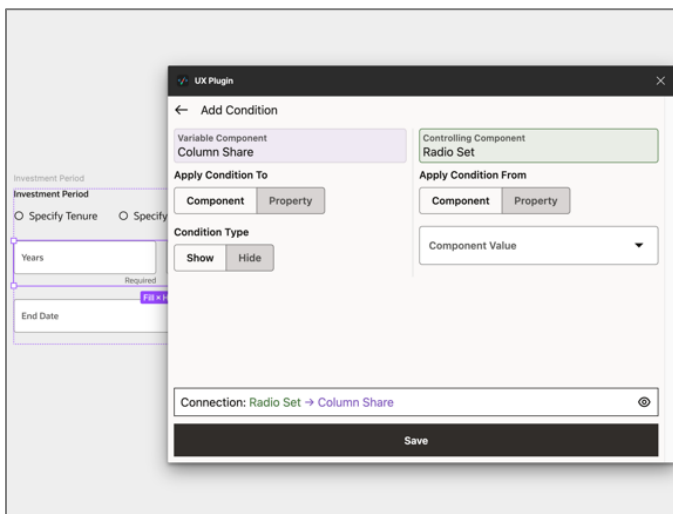
- d. On the next page, choose what aspect of the variable component will be controlled by the controlling component.
- The entire variable component,

OR

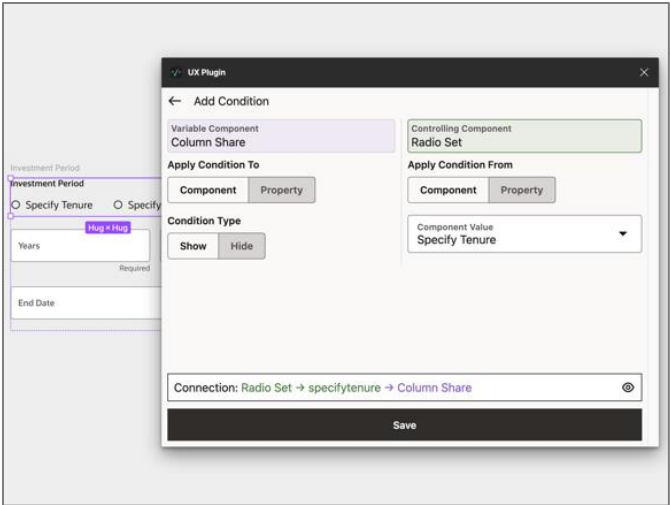
- A property of the variable component
- In this case, show the entire variable component.



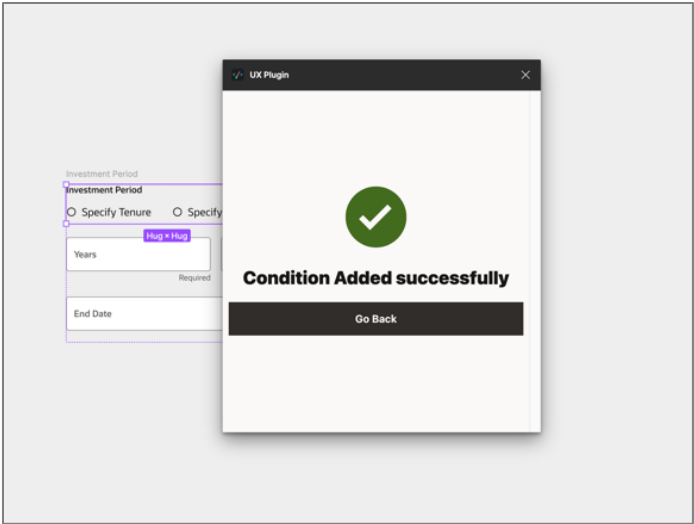
- e. Tap on the button **'Select the Controlling Component'** and choose the controlling component **'Radio Set'**.



- f. Select the controlling radio button **Specify Tenure** from the **Component Value** dropdown to control the visibility of the input fields on tap of the **Specify Tenure** radio button



- g. Review the condition through the connection flow present at the bottom of the window and click **'Save'** to save the condition.



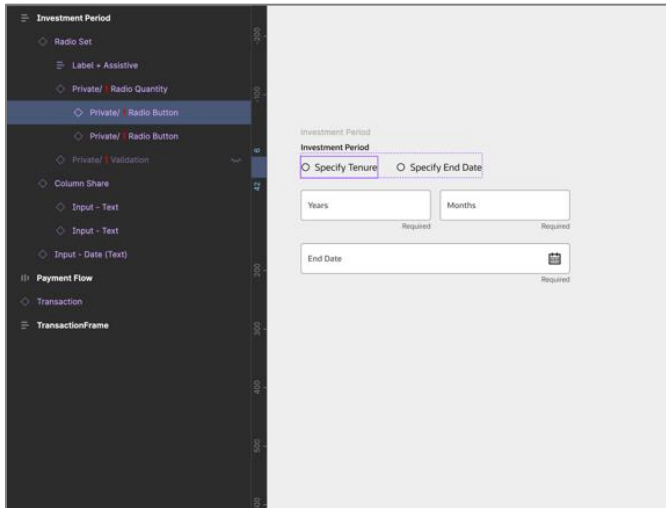
- h. Repeat the same procedure for the other radio button **'Specify Duration'** and ensure that the connection reads:

Connection: Radio Set → specifyenddate → Input - Date (Text)

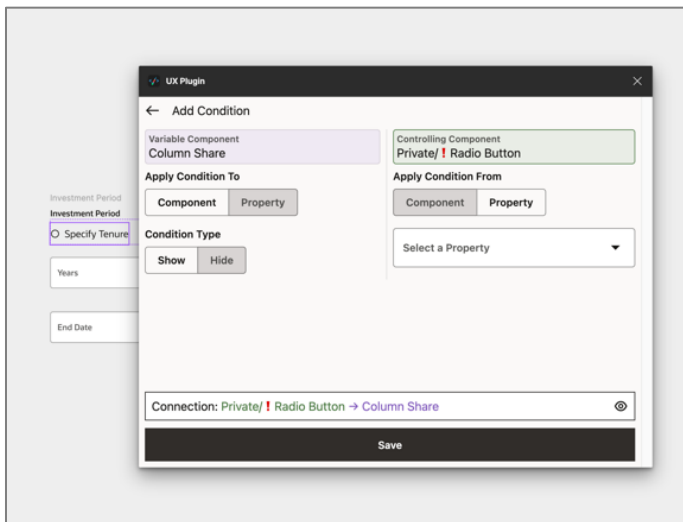
Case 2: Show/Hide of variable component based on the defined properties of the controlling component

Repeat the steps of Case 1 from (a) to (d)

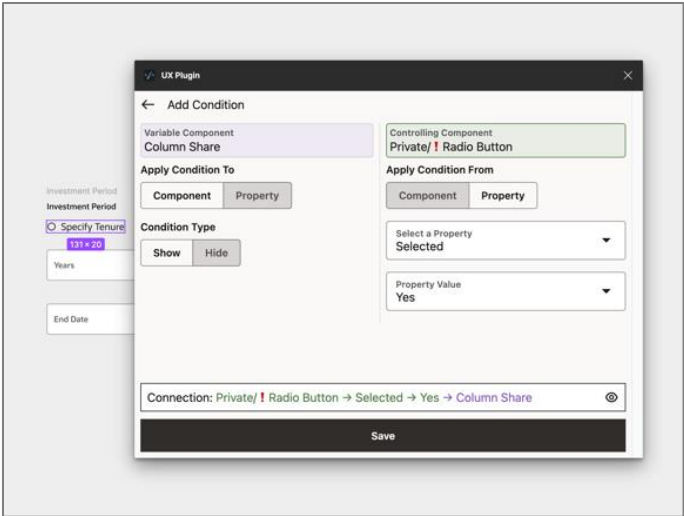
1. After the variable component is set, to provide control to the 'Selected' property of the '**Specify Tenure**' radio button, the exact layer of that radio button needs to be selected. For convenience use Figma's layer panel to select that specific layer



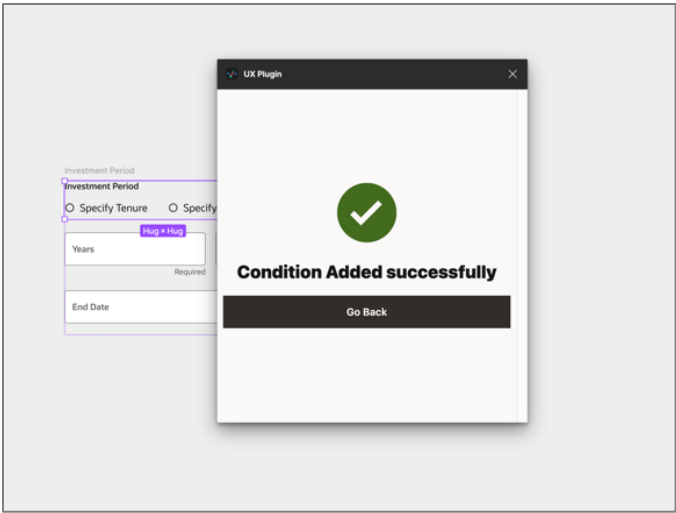
2. Once the right layer has been selected, get back to the plugin and apply the condition from '**Property**'.



3. Choose the 'Selected' property followed by the 'Yes' property value from their respective dropdowns.



4. Review the condition through the connection flow present at the bottom of the window and click 'Save' to save the condition.



5. Repeat the same procedure for the other radio button 'Specify Duration' and ensure that the connection reads:

Connection: Private/ ! Radio Button → Selected → Yes → Input - Date (Text)

7.4 Add Logic

This capability allows designers to enhance conditions by adding "and/or" logic, enabling the creation of multiple, complex conditions for supported components. It provides greater flexibility in controlling component interactions and behavior.

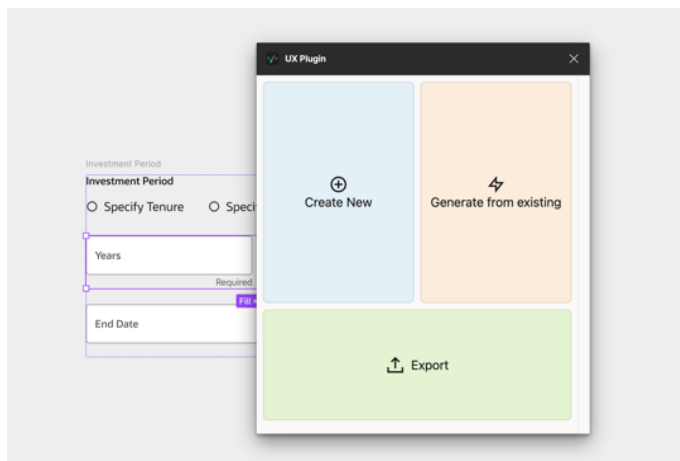
There are two types of Logics that can be added:

- OR Logic: The case is successful if it meets at least one of the conditions.
- AND Logic: The case is successful only if it meets all conditions.

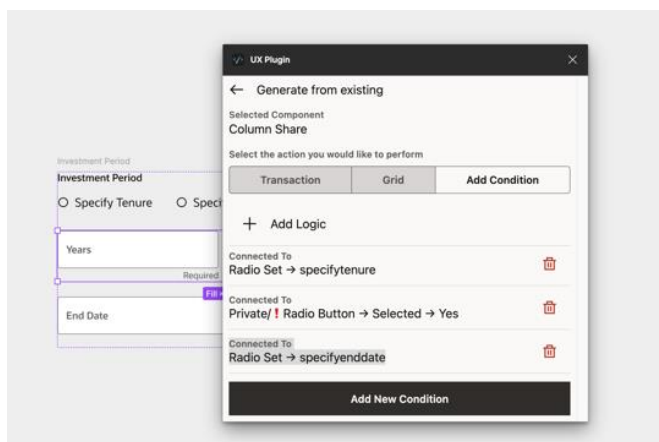
If there are multiple conditions added to a component, they are considered to be in an 'OR' logic.

To add conditions in an 'AND' logic:

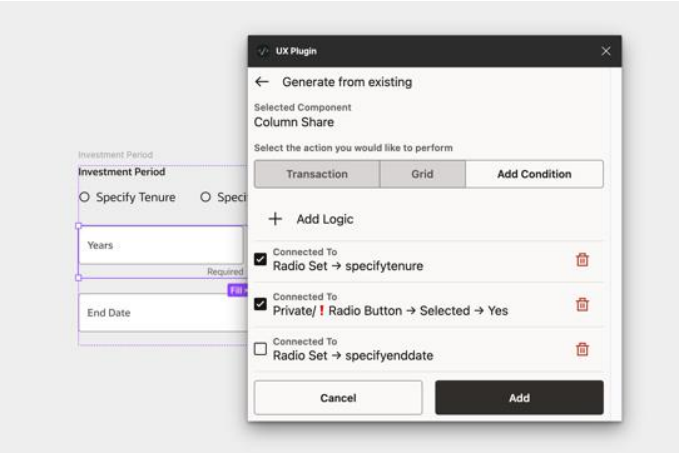
1. Select the component containing multiple conditions, run the plugin, and click on **Generate from Existing**.



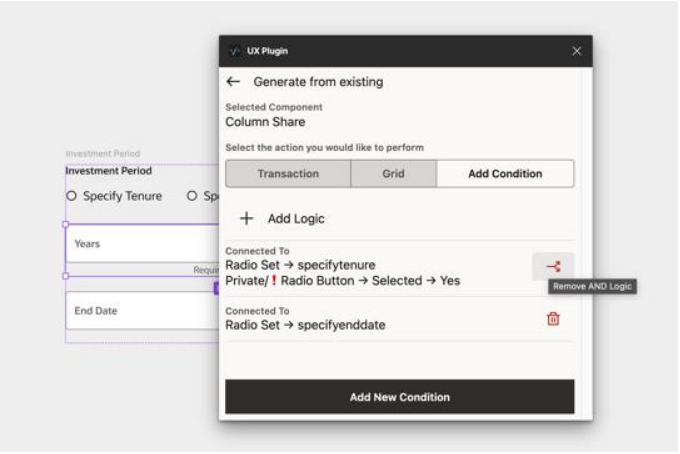
2. Select the **Conditions** button where the multiple conditions are listed, and tap on **Add Logic**.



3. Select the conditions where the **AND** logic is to be applied and hit **Add**.



4. Removing the **AND** logic will default the logic to **OR**.



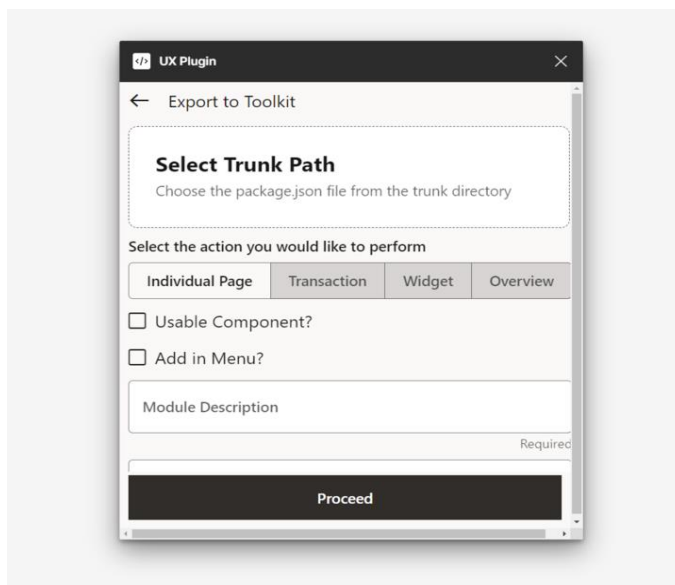
8. Export to Toolkit

This feature allows you to seamlessly convert your Figma design into a code-readable JSON structure. This JSON file contains all the component properties and layout details, making it easy for developers to use the design directly in their toolkit for further development and integration. Once, the design is complete, the designer should handover the design to the developer for exporting.

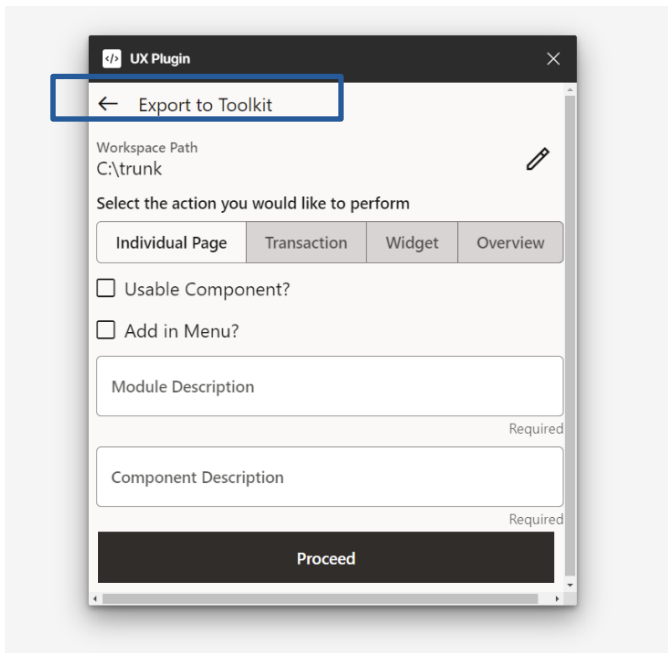
Disclaimer: Ensure that the toolkit is running in the background before initiating the export process.

Steps:

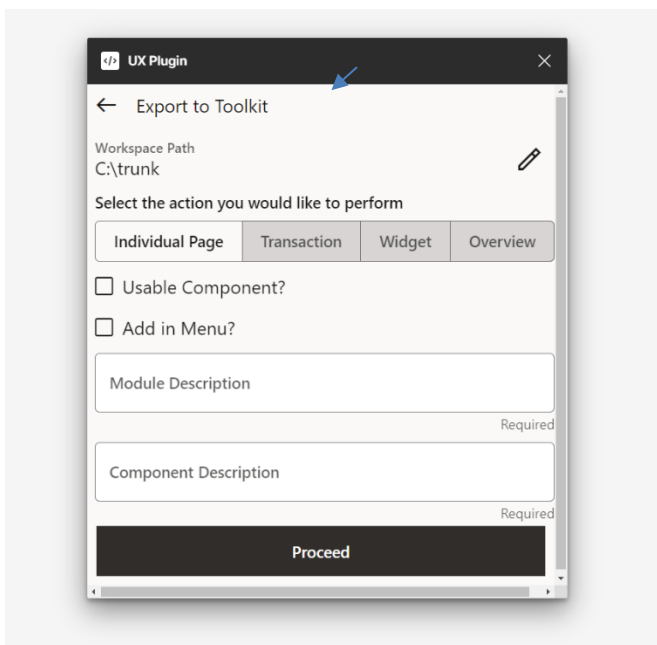
1. Start by clicking the **Drag and Drop** file picker to open the file explorer. Navigate to the folder where your main application files are located.



2. Within the app folder, locate and select the package .json file, then click **OK**. This action sets the file path for the new JSON export.



3. The selected path will now be displayed in the input field. You can manually edit this path if any adjustments are needed.



4. Once the path is set, choose the export mode that best fits your design requirements. There are four different modes to choose from, each serving a unique purpose:
 - a. Individual Page
 - b. Transaction
 - c. Widget

8.1 Individual Page

Use this mode to export a single page of components or specific individual components from your design. It is ideal for exporting standalone pages, specific UI components or reusable business components.

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction Widget Overview

☐ Usable Component?

☐ Add in Menu?

Module Description
Required

Component Description
Required

Proceed

Steps to Export Individual Page:

1. Usable Component?:

Please check this box if the component being exported is intended for reuse.

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction Widget Overview

☒ Usable Component? ☐ Read Only?

☐ Add in Menu?

Module Description
Required

Component Description
Required

Proceed

2. Read Only?:

This checkbox represent if the selected component will have a read-only variant exported as well

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction Widget Overview

☒ Usable Component? ☒ Read Only?

☐ Add in Menu?

Module Description Required

Component Description Required

Proceed

3. Add to Menu?:

- a. This option prompts you to decide whether to add the exported component or page to an existing menu within the application.
- b. If selected, you can choose one or more of the following menus to include your component in:
 - v. Retail
 - vi. Corporate
 - vii. Admin
- c. In the subsequent pages user can place this page in the desired location of the application's menu (explained in Step 7)

UX Plugin

← Export to Toolkit

Individual Page Transaction Widget Overview

☒ Usable Component? ☒ Read Only?

☒ Add in Menu?

Select type of user

retail

corporate

admin

Proceed

d. Module Description:

- i. Enter a folder name where the JSON file will be stored. The base path for this folder is already defined in Step 1.
- ii. This folder will act as a container for the exported JSON, helping organize the exported components or pages systematically within the defined path.

e. Component Description:

- i. Provide a file name for the JSON that will be generated and exported. This name should be descriptive and relevant to the component or page being exported.
- ii. The file name, along with the module folder, will form the complete path for the JSON file, ensuring it's easy to locate and reference in the toolkit.

Note: The component name is used to search the file in the toolkit.

8.2 Transaction

This mode is used for exporting transaction template. Ensure that the entire transaction frame must be selected later.

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction Widget Overview

☐ Add in Menu?

Transaction Description

Required

Proceed

Steps to Export Transaction Template:

a. Add to Menu?:

1. This option prompts you to decide whether to add the exported component or page to an existing menu within the application.
2. If selected, you can choose one or more of the following menus to include your component:
 - i. Retail
 - ii. Corporate
 - iii. Admin
3. In the subsequent pages user can place this page in the desired location of the application's menu (explained in Step 7)

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction Widget Overview

☒ Add in Menu?

Select type of user
retail ×

Transaction Description
File Name

Transaction Name
file-name

Proceed

b. Component Description:

1. Provide a file name for the JSON that will be generated and exported. This name should be descriptive and relevant to the component or page being exported.

Note: The component name is used to search the file in the toolkit.

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction **Widget** Overview

☒ Add in Menu?

Select type of user
retail X

Transaction Description
File Name

Transaction Name
file-name

Proceed

8.3 Widget

Choose this mode to export a set of components that function as widgets, typically used in dashboards or modular interfaces.

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction **Widget** Overview

☐ Usable Component?

☐ Add in Menu?

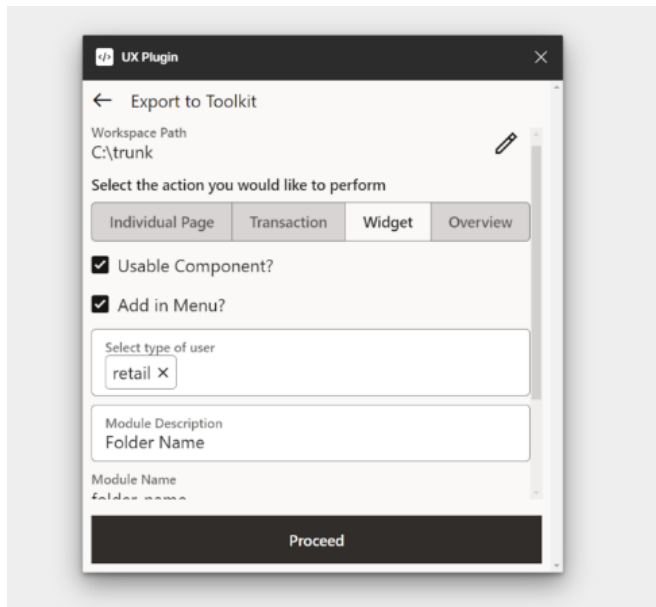
Module Description

Component Description

Proceed

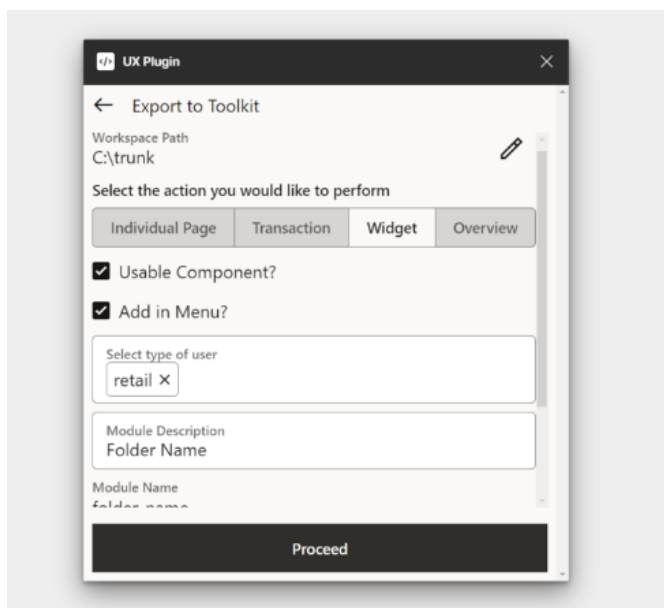
a. Usable Component:

Please check this box if the component being exported is intended for reuse.



b. Add to Menu:

1. This option prompts you to decide whether to add the exported component or page to an existing menu within the application.
2. If selected, you can choose one or more of the following menus to include your component:
 - i. Retail
 - ii. Corporate
 - iii. Admin
3. In the subsequent pages user can place this page in the desired location of the application's menu (explained in Step 7)



c. Module Description:

1. Enter a folder name where the JSON file will be stored. The base path for this folder is already defined in Step 1.
2. This folder will act as a container for the exported JSON, helping organize the exported components or pages systematically within the defined path.

UX Plugin

← Export to Toolkit

Workspace Path
C:\trunk

Select the action you would like to perform

Individual Page Transaction **Widget** Overview

☒ Usable Component?

☒ Add in Menu?

Select type of user
retail ×

Module Description
Folder Name

Module Name
folder_name

Proceed

d. Component Description:

1. Provide a file name for the JSON that will be generated and exported. This name should be descriptive and relevant to the component or page being exported.
2. The file name, along with the module folder, will form the complete path for the JSON file, ensuring it's easy to locate and reference in the toolkit.

Note: The component name is used to search the file in the toolkit.

UX Plugin

← Export to Toolkit

☒ Usable Component?

☒ Add in Menu?

Select type of user
retail ×

Module Description
Folder_Name

Module Name
folder_name

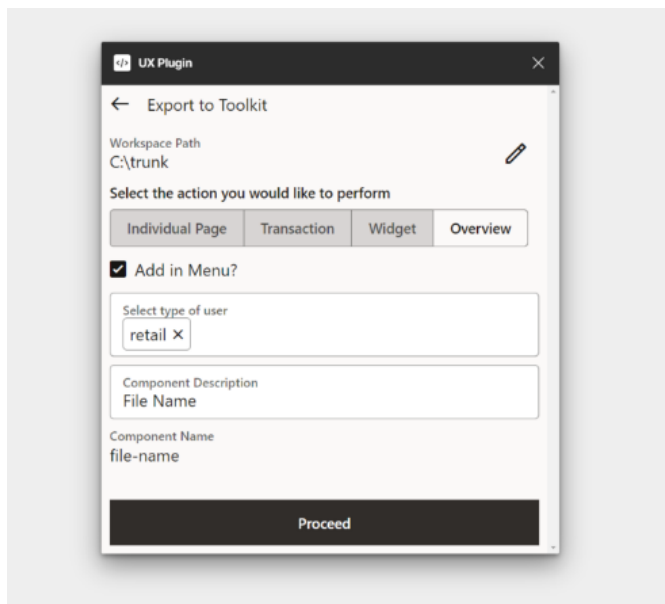
Component Description
File_Name

Component Name
file_name

Proceed

8.4 After completing configurations

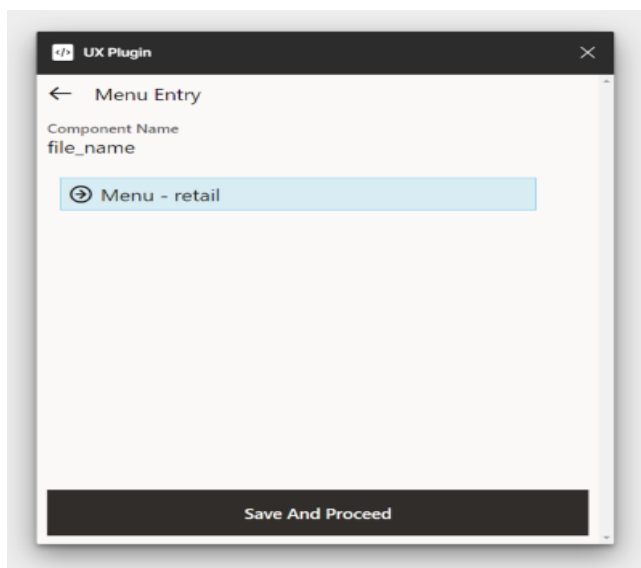
1. After completing all the above steps for your selected export mode, click on the **Proceed** button. This action will finalize the export options and allow the user to either choose the path for the component from the listed **Menu** option (if available) or directly choose the designed component for exporting it to the toolkit.



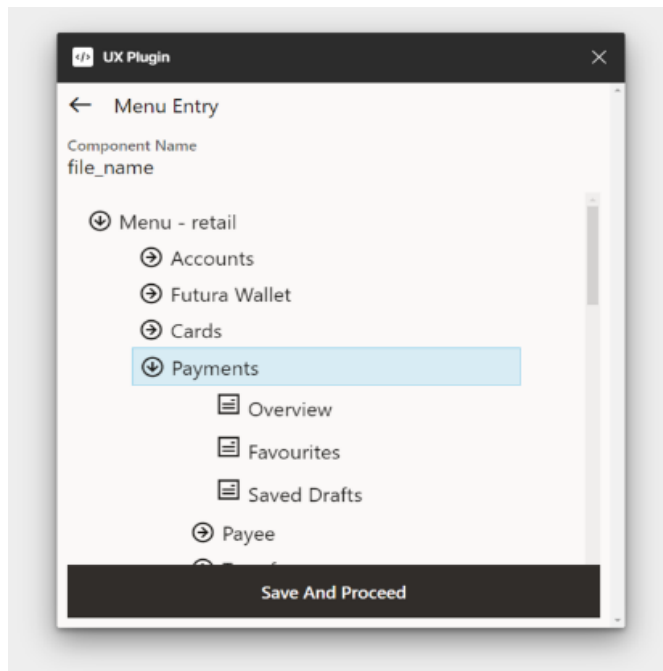
Upon proceeding further, the following two paths are possible:

a. If “Add to Menu?” was selected:

1. A page will open displaying a tree structure of the application's menu. Select the specific path within the chosen menu (Retail, Corporate, or Admin) where you want to add the component. This will define its location within the application's menu structure.

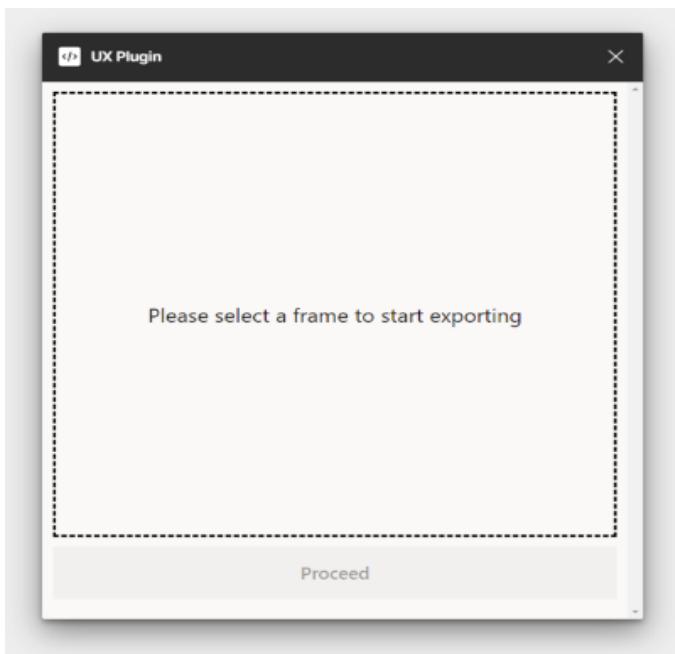


2. After selecting the menu path, click on **Save and Proceed** to direct the plugin to the next page to choose the specific frame or component in your Figma design that has to be exported to the toolkit.

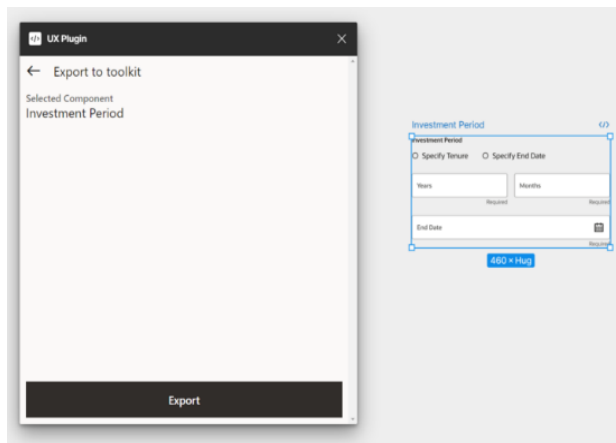


b. If “Add to Menu?” was not selected:

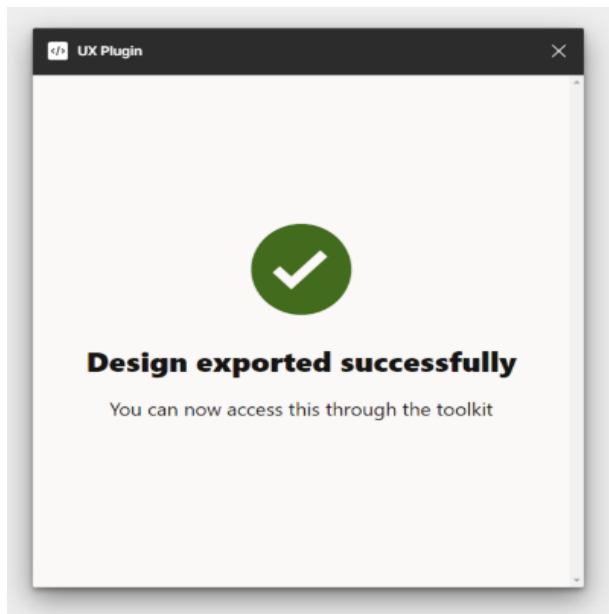
The plugin will be redirected to the page where you can select the frame or component in your Figma design to be exported to the toolkit.



2. Select the specific frame or component in your Figma design that you want to export. This frame will be converted into a JSON structure and integrated based on the configurations you have set in the previous steps. Click on **Export** to export the design into the toolkit.



3. The design is now successfully exported to the toolkit based on the chosen configurations.



FAQs

1. What are the prerequisites to use the UX plugin?

You'll need a basic understanding of Figma, Oracle's Redwood Design System, Oracle JET, and the UI Toolkit. Additionally, an active Figma license and the latest version of the UI Toolkit installed on your system are required.

2. How do I set up the plugin?

Follow the steps outlined in the "Plugin Setup" section of this manual to install and configure the plugin. Once set up, you can start designing with the plugin.

3. Can I create new components using the UX Plugin?

Yes, the plugin allows designers to kick-start their designs by automatically generating components based on the inputs provided. This is done through the "Create New" feature, which streamlines the process of setting up design components according to specified requirements.

4. How do I enhance existing components?

The "Generate from Existing" feature allows you to refine and enhance existing components. This includes capabilities such as converting to transactions, adding grids, and adding conditions.

5. What does the "Convert to Transaction" feature do?

This feature converts a series of page sections into a 3-step transaction template, which can then be further customized by the designer.

6. How do I add a grid to my design components?

Use the "Add Grid" capability under the "Generate from Existing" option to apply a grid structure to your components. Simply select the frame, specify the grid size and alignment, and save the specifications.

7. What is the purpose of the "Add Conditions" feature?

The "Add Conditions" feature allows designers to manage interactions between components by defining how one component (controlling component) affects another (variable component), such as showing or hiding elements based on user inputs.

8. What if I need to apply multiple conditions to a component?

You can use the "Add Logic" feature to apply "and/or" logic to your conditions, enabling the creation of more complex interaction handling.

9. Can I modify layer names in Figma?

It is recommended to keep the original layer names of components unchanged, as modifying them may result in recognition errors by the plugin.

10. What components are supported by the plugin?

The plugin supports components exclusively from the Oracle JET and Redwood libraries. Other components are not recognized by the plugin.

11. How do I export my design to the UI Toolkit?

Once the design is complete, the UI developer can export the design to the UI Toolkit. Ensure that the toolkit is running in the background during the export process to avoid any issues.

12. Can I update an already exported design?

Yes, the plugin allows you to overwrite or update an existing design when exporting to the UI Toolkit. Details on this process can be found in the “Export to Toolkit” section.