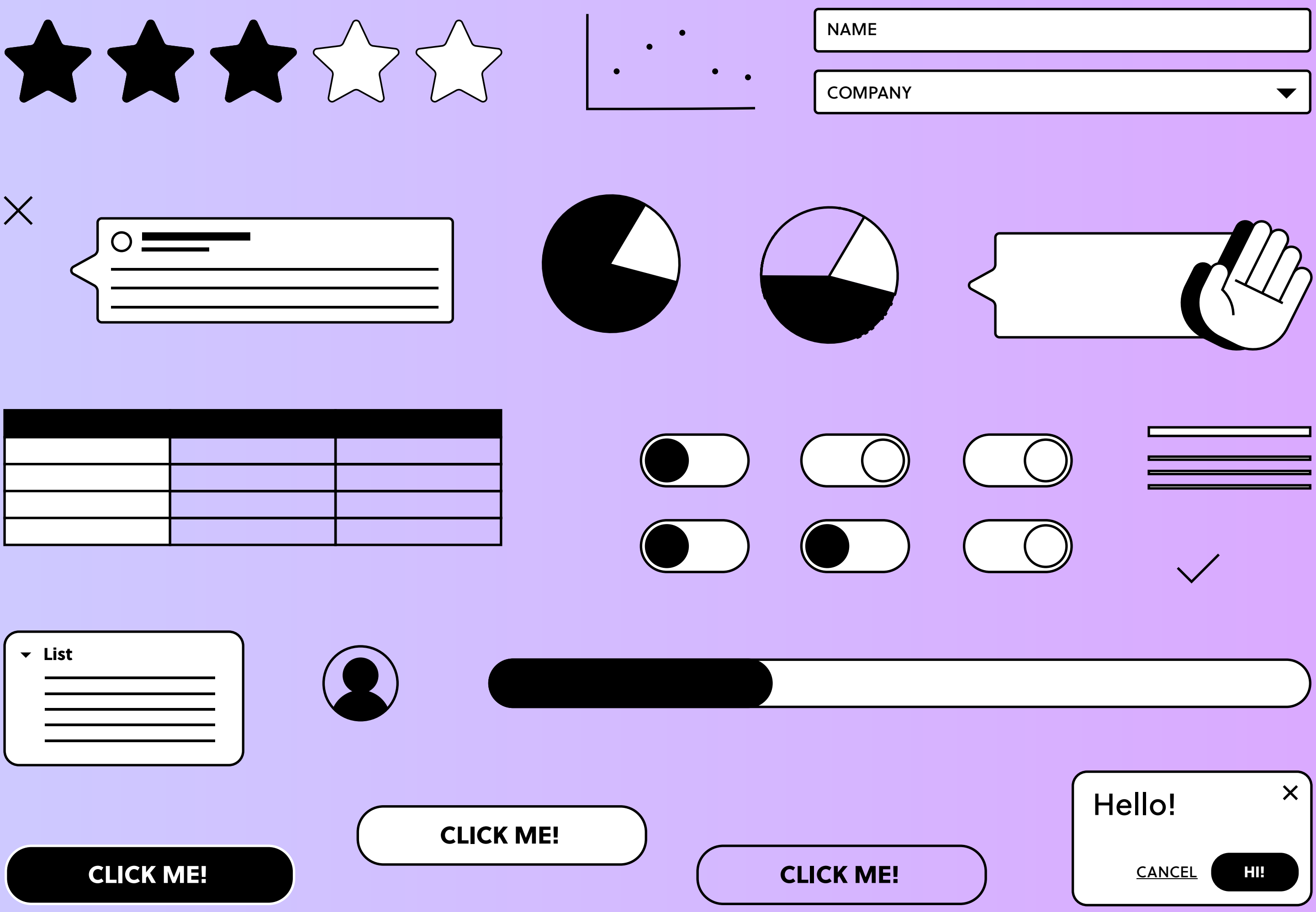
UXPin {M} merge

Interaction Design Patterns for Enterprises



UXPin {M} merge

Interaction Design Patterns for Enterprises

UXPin merge

Create prototypes that are 1:1 reflection of the final product

UXPin allows you to design with fully interactive components that your devs build products with.Now, you can prototype with no room for errors and release products faster. Scale your design process with UXPin Merge technology.

Discover Merge

Build with ready building blocks

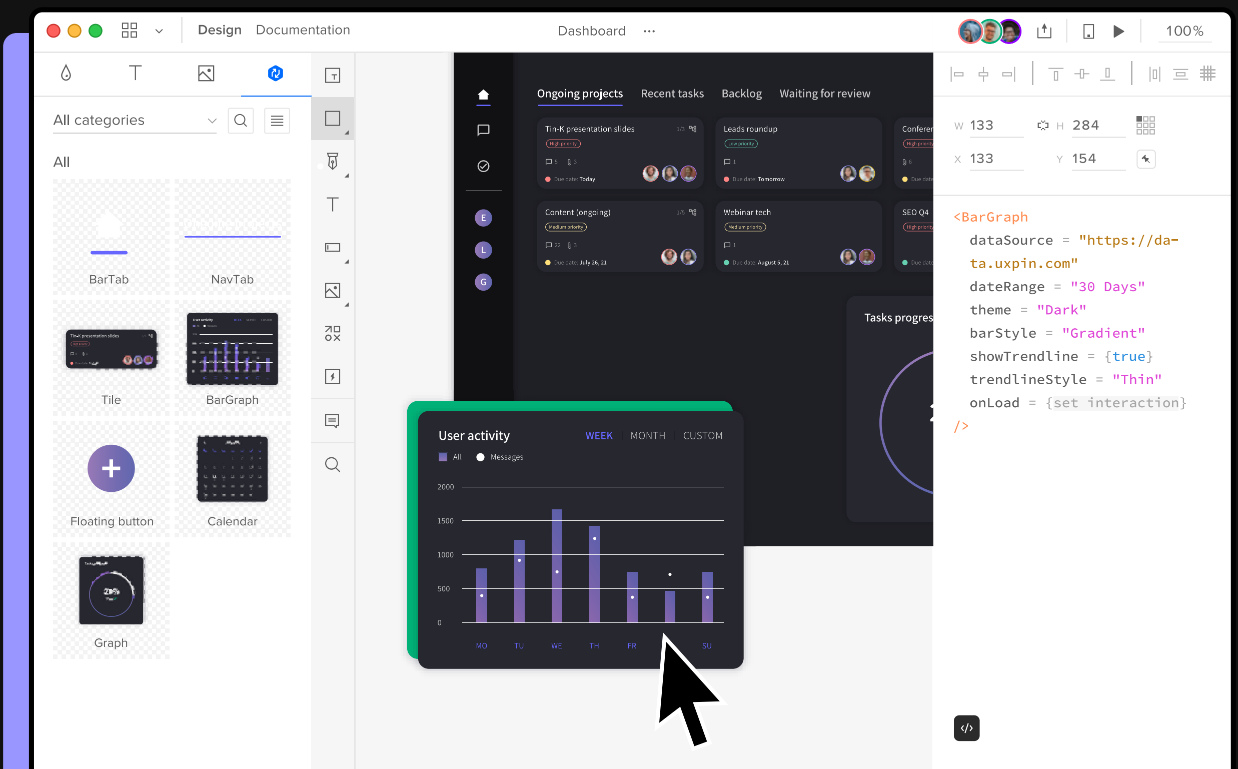
Build prototypes that feel like the end product with hundreds of built-in elements and your code components.

Test real user experience

Design with coded components and enjoy their built-in interactivity that lets you mimic the actual user experience.

Release products faster

Speed up your design process and share ready prototypes with auto-generated specs andproduction-ready code.



```
<BarGraph
  dataSource = "https://da-
  ta.uxpin.com"
  dateRange = "30 Days"
  theme = "Dark"
  barStyle = "Gradient"
  showTrendline = {true}
  trendlineStyle = "Thin"
  onLoad = {set interaction}
/>
```

INDEX

00 INTRODUCTION 5

01 USER INPUT PATTERNS 9

02 DATA DISPLAY PATTERNS 29

03 PROGRESS TRACKING PATTERNS 49

04 NAVIGATION PATTERNS 65

05 ACCOUNT MANAGEMENT PATTERNS 86

06 ALERT PATTERNS 103

07 OUTRO 125

INTRODUCTION

00

Interaction design is everywhere. What makes using a website, app, or any other digital product easy is a series of patterns that create a seamless experience that doesn't require heavy thought.

Interaction design patterns are standardized solutions to common design problems. They provide best practices that help improve usability, enhance user experience, and ensure application consistency. Understanding and using these patterns for enterprise teams can streamline design processes and foster collaboration among cross-functional teams.

Patterns provide a consistent and predictable framework for user interactions, reducing the cognitive load on users and streamlining their journeys. A well-designed pattern makes digital elements feel familiar and reliable, so a user interacts almost without thinking about it.

Design patterns are everywhere, whether you plan for use or not. The goal is to create and replicate common user patterns to make your products work smoothly.

In this eBook, we'll explore a variety of interaction design patterns including:

1. **User input patterns**
2. **Data display patterns**
3. **Progress tracking patterns**

4 **Navigation patterns**

5 **Account management patterns**

6 **Alert patterns**

Each chapter will include a deep dive into each set of patterns, including the most common user interface elements. We'll look at each of these patterns at a high level for your enterprise applications and dig into details for each of the specific user interface elements, including usage, best practices for design, and what interactions and patterns you should use.

How to use this eBook

This eBook is designed to provide chunks of information about specific user patterns and components. While we'd love for you to read this guide cover to cover, it may be more practical to thumb through interactions in your current design workflow to help ensure you are on the right path.

Come back to this guide as needed when you need to support a design decision or get an idea for solving a design pattern problem.

Finally, consider this eBook in context with UXPin's code-backed prototyping. Everything you see in this guide can be created, saved, and reused with UXPin. Our tool is loved by teams who need interactive prototyping with consistent design patterns. Design faster with ready and documented UI elements and stop building from scratch every time.

See how to design 8.6x faster by reusing your coded component library or leveraging an open source. Build a realistic prototype with components with built-in interactivity and clean code, ready to be copied off the design.

We're excited to discuss UXPin Merge with you and show you how it can fit into your workflow.

Request access to UXPin Merge →

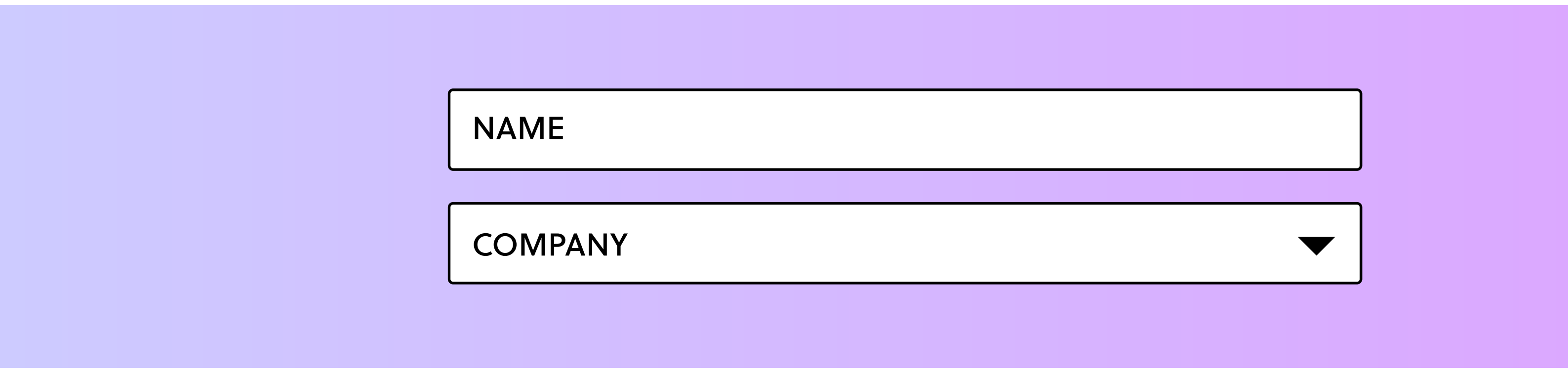
USER INPUT PATTERNS

01

A user input pattern is a recurring design element that helps users interact efficiently and effectively.

These inputs include entering data into form fields, leaving feedback through rating systems, participating in discussions via commenting features, or interacting with buttons. Well-designed user input patterns eliminate friction, making actions easy to understand and follow while ensuring accessibility.

The importance of user interactions cannot be overstated. Every click, tap, or submission is an opportunity to create a positive experience, build trust, and maintain engagement. Poorly designed interactions can lead to frustration, increased drop-off rates, or even lost conversions.



PATTERN 1: FORM FIELDS

A form field is any structured input area where users provide information. It can be used to facilitate signups, logins, or anything that requires standardized information such as name, email address, and phone number. Form field user inputs are one of the most common UI elements found on websites.

What Forms Fields Are Used For

1. Collecting information from users: This can include almost any input, including feedback, surveys, or transactions. A good form field is one that a user can interact with without having to think about how to complete the task at hand.
2. Accessing websites that require accounts or logins: They are used for everything from the initial login or password reset to updating personal details, changing account settings, or modifying any other account information.

Best Practices for Form Fields

FOCUS ON EASE OF USE: The best form fields are easy to use, include plain and clear instructions, and are accessible for all users. Additionally, strong forms only ask for information that is absolutely necessary and use smart components where possible. (Think about how annoyed you get when a checkout form asks for your credit card type—Visa, Mastercard, etc.—when you know the first four digits denote this anyway.

USE CLEAR LABELING: The best form fields include labels that identify each field's purpose. **Design form field** labels to provide clear instructions, ensuring that users know exactly what information to input.

PROVIDE ERROR FEEDBACK: Immediate feedback is important when information is entered incorrectly. Improve user experience with error messages that show in real time when information isn't as directed. This helps reduce user frustration and form abandonment.

PROVIDE HINTS THAT EXPLAIN FORMATTING: Hints help users fill out forms correctly the first time. This might include the pattern for a phone number or date or auto-filling for an address when a ZIP code is entered. Input hints help contribute to a more frictionless overall user experience.

What Interactions to Use on a Form

Form interactions should be simple. This is not the place to dazzle website users with your design and coding skills. Every interaction should aim to help guide users through form completion with ease and confidence.

FEEDBACK: Implement real-time validation to notify users of mistakes as they type, such as a missing .com at the end of an email address. This will keep users happy and reduce submission errors.

ACTION: Carry over user-provided data from screen to screen for a more personalized experience, such as keeping their name present on login. More personalized experiences can enhance user engagement and encourage additional activity.

ACTION: Use smart defaults and auto-fill where possible. If users have auto-fill enabled in their browsers, this can make form fields even quicker, almost ensuring form completion.

FEEDBACK: Consider highlights or form input visual cues to show users where they are in their completion journey. This helps ensure that users won't lose their place or forget to fill in necessary fields.

BUTTON: Use concise but helpful microcopy to ensure that users fill everything out quickly and correctly the first time. This includes instructions such as "Enter a valid email address" or "Password must be at least 8 characters."

ACTION: Include prefilled, sample text inside form fields to illustrate what you are looking for with each form input.

FEEDBACK: Show success indicators such as checkmarks for fields that are completed correctly. These small wins can make users feel good about their progress and keep going.

ACTION: Make sure every form and input works equally well across device types. Test on desktop, mobile, and tablet devices.

BUTTON: Use calls to action with clear labels to complete the form submission. A button is a clear and common user pattern that signifies the end of a user interaction.

FEEDBACK: Don't forget the completion confirmation. Once the form is submitted successfully, display a clear success message or confirmation page with next steps or a thank you.

How to Create a Form Field

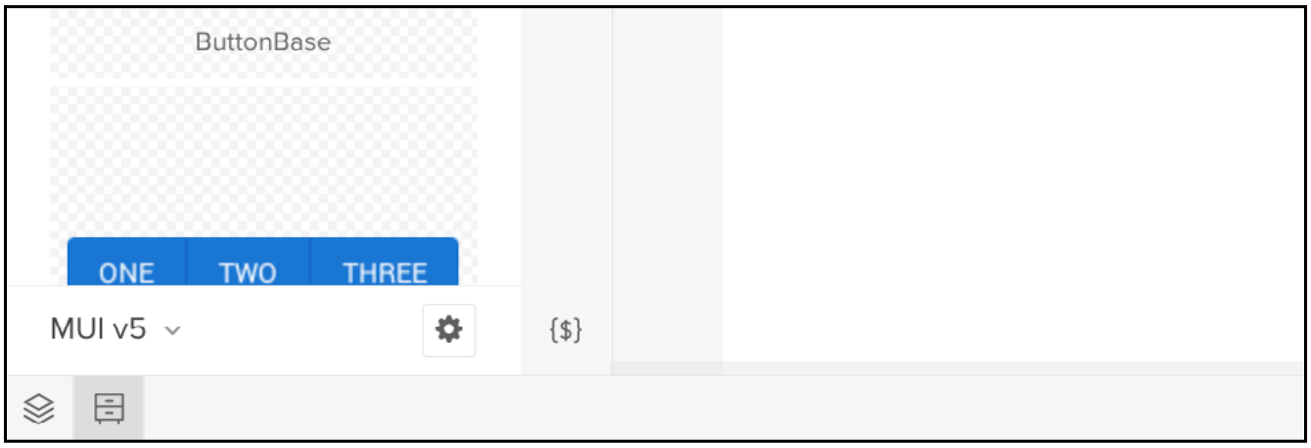
Using UXPin

To create form fields in UXPin, **read this tutorial** or follow these steps:

1. Add Form Elements: Use the Quick Tools panel under "Forms" to add elements like input fields, radio buttons, and dropdowns.
2. Set Up States: Select an input field, click "Add State," and create states like "Base" or "Empty" for email and password fields.
3. Add Interactions: In the Properties Panel, add conditions to trigger error messages when fields are empty or gain focus, and clear errors on focus.
4. Button Interactions: Set up the button to validate the inputs and show a confirmation screen if all conditions are met.
5. Validation: Use regex to validate email format and password length through conditions in the Properties Panel.
6. Confirmation Screen: Display the email on the confirmation screen after successful validation.

Using UXPin Merge

In UXPin, you can easily access form components built with coded elements from Merge libraries, offering inputs like radio buttons, checkboxes, and text fields.

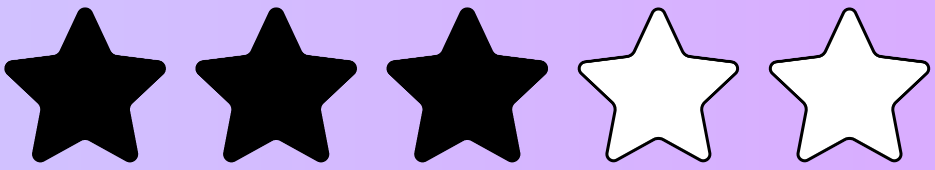


1 Access Form Components: Go to the Design System Libraries in the bottom-left corner of the editor. Choose a Merge library, like MUIv5 or Ant Design v5, and search for components such as "Button," "Radio Button," or "Form."

2 Drag and Drop: Simply drag and drop the components onto your canvas, then customize their properties in the right panel. You can adjust specific properties for individual elements like radio buttons and checkboxes directly from this panel.

3 AI Component Creator: Use the AI Component Creator in the Quick Tools panel to generate full forms from libraries like MUI, Ant Design, Bootstrap, or Tailwind CSS based on your design needs.

TIP: UXPin Merge with Merge AI plan is the fastest way to build a design system for your company, ensuring consistency in looks and behavior. Pick a library, customize its components, and save them for reuse.



PATTERN 2: RATING SYSTEMS

A rating system is a user input design pattern that allows users to provide feedback or express their opinions on content, products, or services. This is a widely used and popular design element, especially for shopping and ecommerce sites.

What Ratings Are Used For

1. Gather feedback: Collect and quantify user feedback or opinions about a product, service, or experience. It allows users to express their level of satisfaction, agreement, or preference in a structured and simple way.
2. Assess quality: Ratings can tell you if users are happy with a product, by offering ratings or recommendations that may encourage others to buy it as well. Ratings can help a business determine the quality of a product or service they provide.
3. Compare items: Rating systems allow users to compare products against one another when making a purchase decision. This can be a strong indicator for some users as they go through the decision-making process for anything from buying a single option to booking travel.
4. Social proof: Rating systems provide an element of social proof for users, showing them that others have had a good (or not-so-good) experience with the things that's being rated. In this instance, ratings can build trust, improve overall user experience, and help generate greater website conversions.

Best Practices for Rating Systems

MAKE IT FUN AND INFORMATIONAL: Most interactions include only a single step for providing feedback, such as denoting a number of stars. More complex systems may also allow commenting or additional feedback on specific line items within the overall rating.

DESIGN IT FOR UNDERSTANDING AND COMPREHENSION: That includes using familiar symbols, such as stars, a thumbs-up/thumbs-down, or clear numeric scale for ratings.

HELP USERS: Include information to guide users in making the right choice. This is especially important with scales; note an instruction such as 1 star = Poor, 5 stars = Excellent.

ENSURE ACCESSIBILITY: Make it accessible with a rating system that is navigable via keyboard and labels that include both images and text, where icons or images are used.

What Interactions to Use with Rating Systems

The best rating systems are understandable at a glance and often work with a single user interaction. If there's a next step, the user gets an instruction after the initial input is received, such as noting a number of stars and then being asked if they would like to leave a comment.

FEEDBACK: Provide visual feedback when users hover or click on rating elements, such as highlighting, color change, or animation.

FEEDBACK: Update the rating display in real-time as users interact.

ACTION: For more granular rating systems, such as a wide scale (0 to 100), use a slider or drag-and-drop functionality. Users can slide to their rating or drag items in a preferred order.

ACTION: Allow users to change their rating until it is submitted.

FEEDBACK: To provide engaging feedback, incorporate subtle animations, such as stars gradually filling up or a thumbs-up animating on click.

FEEDBACK: If the user is not engaging in a manner that will create a result, show a clear error message.

ACTION: Design responsive rating elements that are touch-friendly for mobile devices, ensuring that users can interact anywhere.

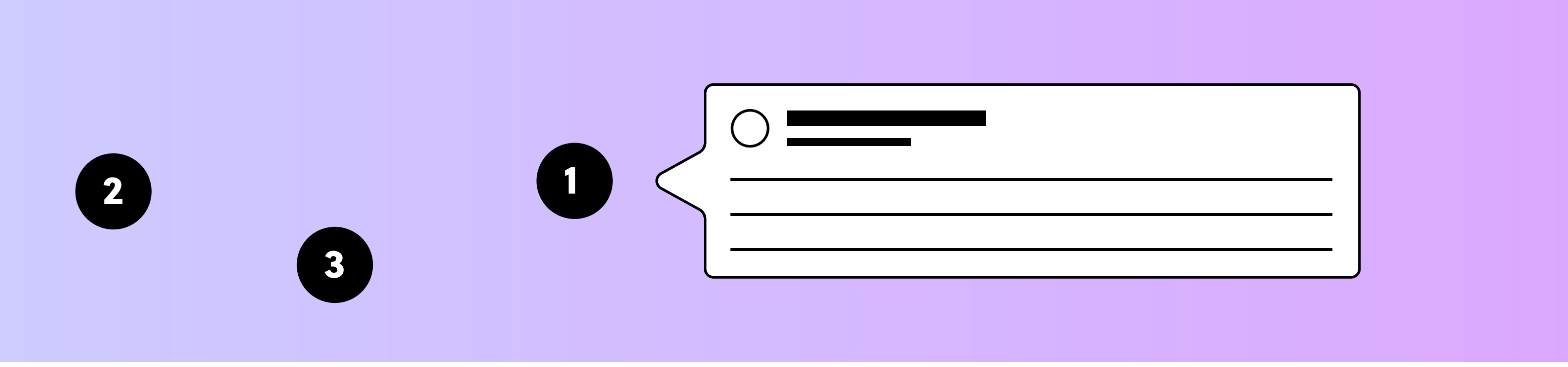
FEEDBACK: Show a confirmation after the rating is submitted.

CONTENT MANAGEMENT: Display a summary of ratings or averages in real-time, including the average rating or total number of ratings received.

How to Create a Rating System

In UXPin, creating custom rating systems and form fields is simple with the pre-built components available in Merge libraries.

1. Access Components: Navigate to the Design System Libraries in the bottom-left corner of the editor or press Cmd/Control + F to search for components. Type in "Rating" or "Form" to find elements like radio buttons, checkboxes, and star ratings from libraries such as MUIv5, Ant Design v5, or React Bootstrap v5.
2. Drag and Drop: Select the desired component and either click or drag it onto your canvas. Once on the canvas, customize it using the right-hand panel.
3. Customize: For form components, adjust properties like input type, labels, and states, while rating components allow control over icon styles, maximum rating, size, and value. You can also modify styles using the sx property with JSON5 syntax, customize with JSX code, or override default styles with classes.
4. Consistency with Code: All components are synced with code, ensuring seamless collaboration between design and development.



PATTERN 3: COMMENTS

Comments are everywhere from social media feeds to blogs to reviews. Commenting is one of the oldest forms of user input online and an expected method of feedback for many users.

What It Is Used For

1 Gather feedback: Commenting is a direct feedback channel. It is used by internal teams and by the general public. Internally, commenting can help in the design and development process with markup to help in the creation of a digital product. Externally, it is a way to generate feedback from users.

2 Rating systems: While commenting isn't a rating system, it can be an element of one. Whether it's on a news article, blog post, or e-commerce site, comments allow users to express their opinions, share experiences, and offer suggestions.

3 Interacting with other users: Commenting is also a way to help build community online. Comments foster discussions and allow users to interact with one another, often sparking debates, sharing ideas, and creating long-lasting engagement.

4 Get support: Users can engage in commenting as a method of support. Question/comment and answer formats are especially beneficial on product pages or tech forums where users look for troubleshooting tips.

Best Practices for Comments

PUT CONSTRAINTS OVER INPUTS: Commenting user inputs need clear constraints, such as the size of the comment input field, submission button, and instructions or guidelines.

TAKE CARE OF EASE OF USE: The most important user pattern for commenting is ease of use. Commenting fields and modules should look like what a user would expect. Don't try to create something new and different here.

PRIORITIZE ACCESSIBILITY: Commenting should be an accessible feature, meaning the commenting system is fully accessible via keyboard navigation and compatible with screen readers. Use ARIA labels and roles.

PUT RESTRAINTS INTO INPUTS: Establish a plan for filtering and moderation to remove spam, offensive language, and inappropriate content.

What Interactions to Use with Comments

Subtle user cues are the norm when planning your design for a commenting system and related user patterns. Because the focus is on text input here—sometimes in a long-form manner—east of text entry should be the primary focus.

FEEDBACK: Provide feedback when users hover over interactive elements, such as comment buttons or reply links. This could include changes in color, highlighting, or animations.

ACTION: Update new comments in real-time, without requiring a page refresh.

ACTION: Allow users to edit their comments by placing an "Edit" button or link next to them.

BUTTON: Include a "Reply" button on each comment with visual cues to indicate the hierarchical structure of replies.

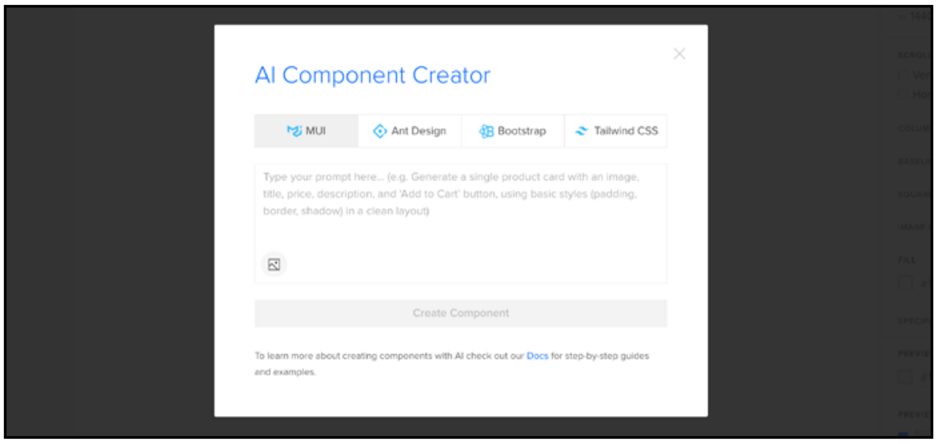
FEEDBACK: If you have a character or limit on comments, display an indicator to inform users how much space remains. Provide real-time feedback if it exceeds the limit.

CONTENT MANAGEMENT: Use rich text formatting for commenters, so they can include elements such as bold, italics, etc.

BUTTON: Design a clear submit button for comments as well as comment-received feedback.

How to Create a Comment Section

To generate a comment section use the AI Component Creator in UXPin. Follow these steps:



Type your prompt here... (e.g. Generate a single product card with an image, title, price, description, and 'Add to Cart' button, using basic styles (padding, border, shadow) in a clean layout)

To learn more about creating components with AI check out our Docs for step-by-step guides and examples.

1. Open the AI Component Creator: Navigate to the Quick Tools panel on the left-hand side of the editor. Click on the AI Component Creator option.
2. Specify Component Type: In the AI Component Creator window, describe the component you want to generate. For a comment section, input a description like: "Generate a comment section with user avatars, comment text, and reply buttons."
3. Choose a Library: Select the code library you prefer, such as MUI, Ant Design, Bootstrap, or Tailwind CSS. This will ensure the generated comment section follows the design system you are working with.
4. Generate the Component: Click the "Generate" button, and the AI will create a coded comment section based on your input.
5. Customize the Comment Section: Once the component is placed on the canvas, you can customize its properties using the right-hand panel. Adjust things like avatar size, comment styling, spacing, and other visual elements to fit your design.

PATTERN 4: BUTTONS

A button may be the most basic user input–click or tap and something happens almost instantly. The perfect button is identifiable, findable, and clear. That means every button "looks like a button," is easy to see on the screen with ample space to click or tap, and includes direct instructions that say exactly what happens upon interaction.

Button design is often similar throughout a website, even if buttons lead to a different user interaction. From a design perspective, common button styles include text, ghost, raised, and toggle.

What Is a Button Used For

1. Call to action: A button is a graphical element typically appearing as a clickable area on a digital interface. Its primary purpose is to convey a specific call to action, and communicate what users can do next.
2. Complete or continue action: Buttons can convey action, guide users to something else (internal or external) or provide feedback. Button usage should be designed in a tactile manner so that the digital button seems to mimic the look, feel, and interaction of a button in the real world.

Best Practices for Buttons

There's one golden rule when it comes to button design—buttons should look like buttons.

There are a few design patterns that users associate with what a button should look like:

SMALL FILLED BOXES WITH SQUARE OR ROUNDED CORNERS, often in an accent color on your website with microcopy (standard button).

SMALL FILLED BOXES WITH A DROP SHADOW AND MICROCOPY (raised button).

CLEAR FILLED BOXES WITH AN OUTLINE and micro-copy (ghost button).

AN OUTLINE ITEM, usually pill-shaped with a circle inside to denote on/off (toggle button).

A TEXT ELEMENT IN A STANDALONE SPACE that takes users somewhere else on the site, such as in main, footer, or sidebar navigation (text button).

What Interactions to Use with Buttons

Every action that happens from a button interaction should be clear and direct.

BUTTON: Microcopy should state exactly what will happen next and every button should have a label.

IDENTIFICATION: Icons should be commonly used and easily identifiable. Most everyone knows what the "play" triangle looks like, but there's no common icon for "read more."

IDENTIFICATION: Buttons should have multiple visual signifiers of action, such as a hover state with a color change and underline.

ACTION: Clearly distinguish disabled buttons from active ones using visual cues such as reduced opacity or a grayed-out color to indicate that the button is not active.

FEEDBACK: For buttons that trigger actions with a delay, such as a form submission, include a loading indicator to show users the action is being processed. Don't activate a button until all requirements to submit have been met.

ACCESSIBILITY: Use ARIA attributes to enhance accessibility for screen readers for buttons with icons or ambiguous text.

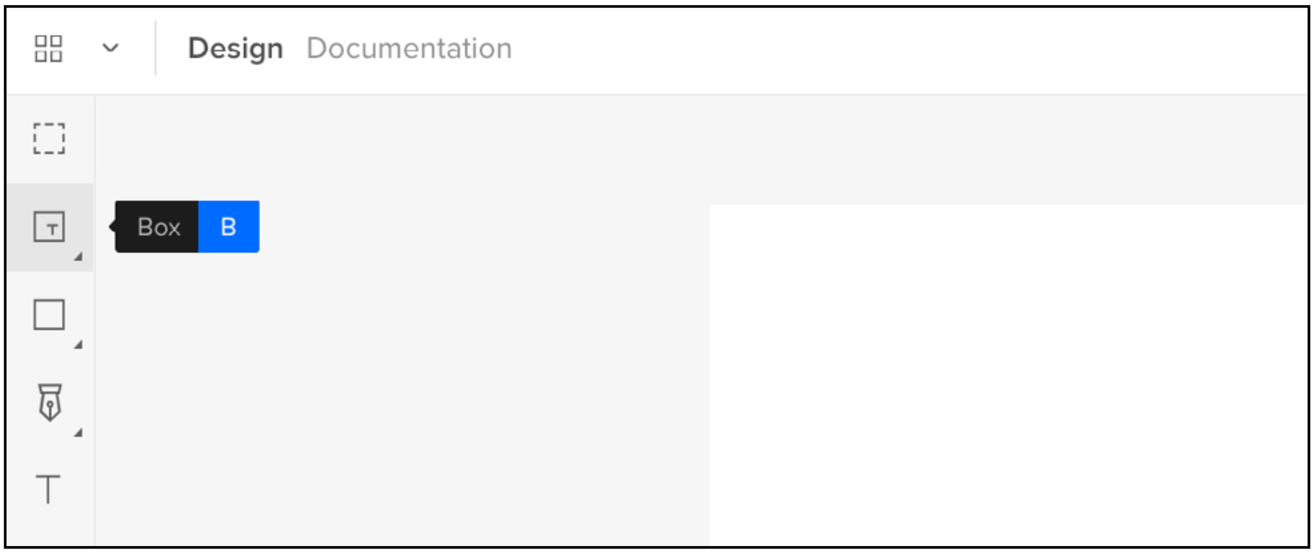
ACTION: Distinguish between primary and secondary actions using different button styles or placements. Primary actions should be more prominent.

FEEDBACK: Provide feedback when a button action is successfully completed, such as a success message for buttons for forms. A button that clicks to a new URL or location provides innate feedback.

How to Create Buttons

Buttons play a crucial role in driving actions like submissions, downloads, triggering modals, navigating pages, and confirming decisions.

Create a Button Using the Box Tool

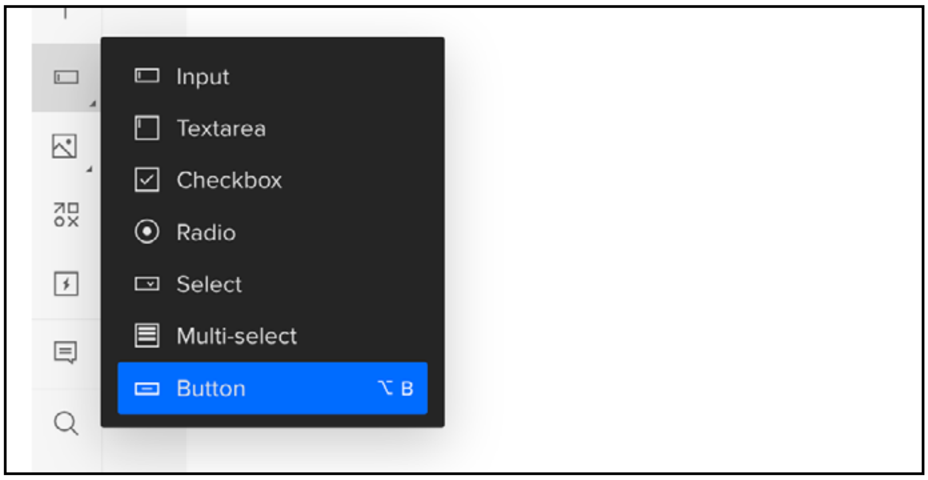


1 Select the Box tool from the Quick Tools panel or press "B."

2 Click and drag on the canvas to draw your box, then double-click inside to add text.

3 Customize the button's properties in the right-hand panel, such as font size, colors, font family, and more.

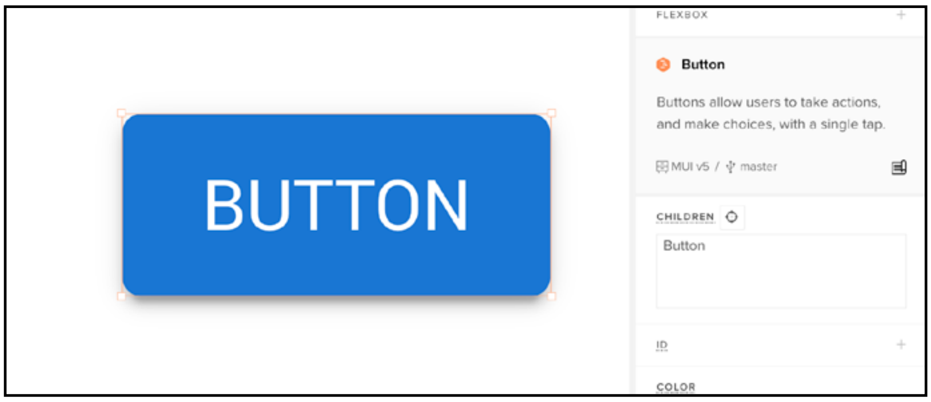
Add a Form Button



1 From the Quick Tools panel, navigate to **Form** and drag the **Button** component onto your canvas.

2 Adjust its properties and behavior in the right panel, and add interactions to enhance its functionality, such as form submissions or triggering actions.

Using button components in UXPin Merge



Buttons allow users to take actions, and make choices, with a single tap.

1 Navigate to the Design System Libraries in the bottom-left corner of the editor.

2 Choose your Merge library (e.g., MUI), then search for the "Button" component.

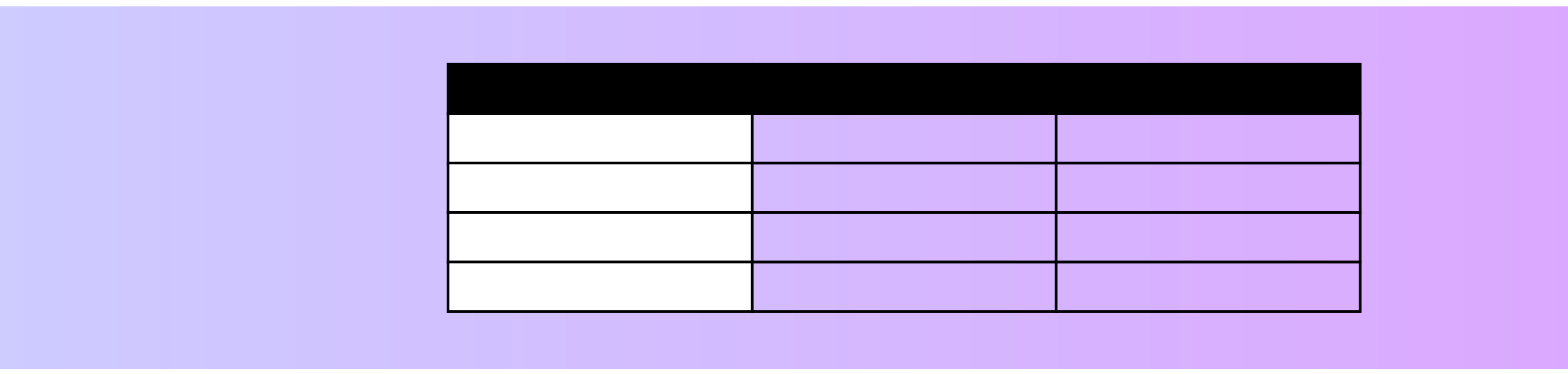
3 Drag it onto the canvas and customize it in the right panel to suit your design needs.

DATA DISPLAY PATTERNS

02

Enterprise companies often have a lot of data. Moreover, that data is used to enhance the user experience for all types of customers, from personalizing to showing trends, patterns, outliers, and insights.

Most data displays have some common elements that encourage user action including a strong visual hierarchy to make comprehension easier, interactive elements to encourage engagement, real-time data to information has a sense of urgency and time, and accessibility to that data is understood by all.



PATTERN 1: DATA TABLES

Data tables make it easy to interpret large amounts of information quickly and in a more organized manner. Most companies use a data table for everyday inventory or list management functions. Data tables have a distinct structure for displaying information, including a header, rows, and columns, often in a gridded element.

Data tables can be internal-facing elements of your website or intranet, such as dashboards and tools that make your company run smoother. On the other hand, data tables can be public-facing elements that help users find information or make a choice about your business or product.

What Data Tables Are Used For

1. Show and interact with data: While data table design is important, function is imperative. The key uses for data tables are often linked to business operations, meaning they have to show data and allow users to interact with it flawlessly every time.
2. Compare data: Data tables are a practical way to compare data using rows and columns – often with sorting features – to better understand large amounts of information, including financial records, product lists or inventory, and employee data. These tables also often allow for the entry and editing of information.
3. Track data in real-time: Tracking functionality is another key feature of data tables. It allows you to track things in real time, from sales to support inquiries.
4. Reporting: Data tables are the most common methodology for reporting complex data. Each row or column represents different bits of information that can be compared for performance and exported to a spreadsheet or other format for offline work or distribution.

Best Practices for Data Tables

ALLOW SORTING AND FILTERING: Allow users to sort by column or row, such as clicking the header to reorder alphabetically, numerically, or by date. Include filters—often in the form of a dropdown—for common actions such as filtering by date range, status, or category.

CONSIDER SEARCH: A search bar that filters across all table fields can make large amounts of data more manageable.

MAKE IT RESPONSIVE: Data tables must work well on every device size, and responding tables can be tricky. Consider allowing users to scroll horizontally or collapse less important columns on smaller screens.

CONSIDER EDIT FUNCTIONALITY: If users need to change data or make updates, edit functionality is important. Inline editing functions are often reserved for internal data tables and are not public-facing.

DESIGN A USABLE EXPORT: Many enterprise users need to export data from tables in CSV, Excel, or PDF formats. An easy export feature ensures a seamless transition to external or offline tools.

What Interactions to Use with Data Tables

Interaction design is focused on function and data analysis with data tables more than anything else. Every user might look at information differently, therefore needing a different set of interactions to complete their tasks.

ACTION: Click or tap a column header to sort the table in ascending or descending order.

FEEDBACK: Allow users to hover to highlight rows or columns to improve readability.

ACTION: Users should be able to expand rows for additional details in tables that include a lot of information or have nested data.

ACTION: Dragging column headers to rearrange the order of the table.

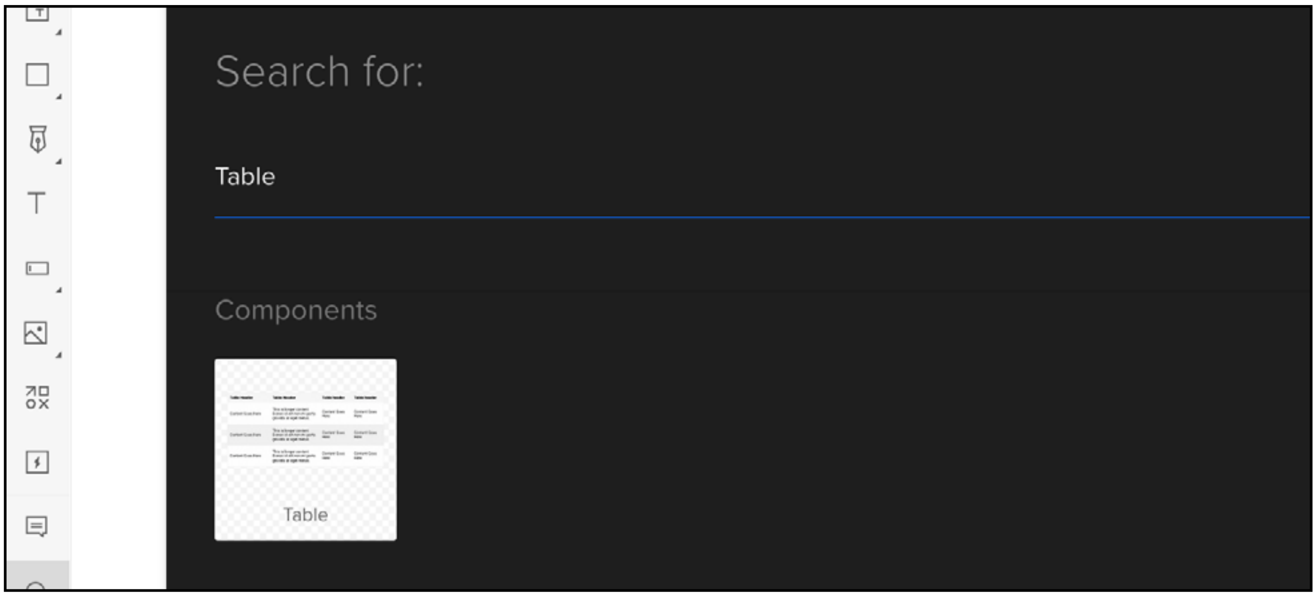
ACTION: Users can select multiple rows simultaneously with checkboxes for actions like deleting or exporting data.

ACTION: For inline editing, users can double-click a cell to edit its content.

How to Create a Data Table

To create tables in UXPin, you have two powerful methods: using table presets or leveraging UXPin Merge components.

Using Table Presets



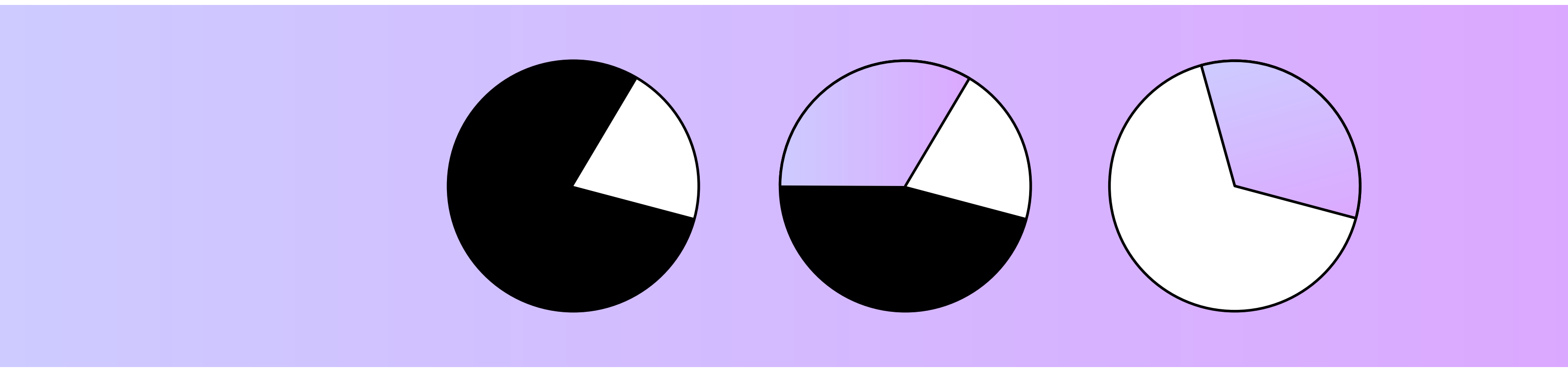
1 Click on **All Assets** in the Quick Tools panel or press Cmd/Control + F to open the search bar. Type in "Table" to access table presets.

2 Drag and drop one of the table presets onto your canvas. Once placed, customize its properties, such as color, cell padding, and borders, in the right-hand panel. These tables are already responsive, with scroll bars appearing automatically on smaller screens for easier navigation through larger datasets.

3 For more control over the design, use **Power Duplicate** and **Smart Grid** features to create custom table elements. Power Duplicate allows you to replicate elements consistently, while Smart Grid helps manage spacing and arrangement in grid layouts efficiently.

Using UXPin Merge Components

1. Navigate to the **Design System Libraries** in the bottom-left corner of the editor, select a Merge library like MUI or React Bootstrap v5, and search for "Table."
2. Drag and drop the table component onto your canvas. In the right panel, you can control elements like columns, cells, and table variants. Customize settings like padding, size, sort direction, and apply advanced styling using the **sx** property with JSON5 syntax, giving you full control over background color, borders, typography, and more.



PATTERN 2: PIE CHARTS

Pie charts are one of the easiest data visualizations. All of the pieces are part of the whole, totaling 100% of a specific data point. They are often used for high-level summaries and quick visual comparisons.

Data in pie charts can be shown as percentages of a whole, with everything equaling 100% or whole numbers that add up. Many pie charts include both percentages and whole numbers to aid in comprehension.

From a design perspective, one of the most important considerations in a pie chart might be color. You need plenty of contrast in color and pattern for each "slice of pie" so that the chart is easily understood. This gets more complex with charts that have more pieces to account for.

What Pie Charts Are Used For

1. Quick comparisons: Pie charts are a method for making quick comparisons. You see them everywhere, from nonprofit websites that show how donations are used to internal dashboards that explain budget allocations to highlighting demographics or where users come from.
2. Chunking data: Data types that make pie charts function well include financial reports, segmentation reporting, and analytics. (You've probably seen plenty of pie charts in your website analytics!) A pie chart might be used when drilling down a bit of information into smaller data segments.
3. Comparing: A pie chart is ideal for showing how individual parts contribute to a whole.
4. Comprehension: They can help create a better understanding of how different categories of things, from products to budget allocation, compare.

Best Practices for Creating Pie Charts

LIMIT THE PIECES OF PIE: This style is best with limited data segments. Try to limit to less than eight slices.

USE CLEAR LABELS: Every segment should include a clear label indicating the type of data displayed and how it contributes to the overall data set (such as percentage). If necessary, you can use a key or legend.

ENSURE PIE SLICES ARE PROPORTIONATE AND ORDERED BY SIZE: Slices should be proportional to their actual values, meaning a 50% value should have a piece that's as big as half of the overall pie (circle). Arrange slices in descending order of size, starting from the top and moving clockwise.

USE CONTRASTING COLORS AND PATTERNS: Use distinct colors and/or patterns for each slice and avoid using similar colors or patterns close to one another. (This is a lot easier to do if you have limited the number of slices.)

What Interactions to Use with Pie Charts

Simple pie charts won't have as much functionality as other types of data you display in your interfaces. Generally, these are simple, with just a few common interactions.

FEEDBACK: When users hover over a slice, it should include a visual interaction such as a color change, expansion, or highlight that allows users to better see that element.

ACTION: Clicking a slice should show additional information when applicable, encouraging users to spend more time with the chart.

CONTENT MANAGEMENT: Charts with real-time data should update dynamically without losing context.

ACTION: If present, a key or legend should connect to the data displayed. If you interact with a certain label on the legend, that data activates in the chart (see the first interaction).

ACTION: Use simple and smooth animation to help create greater understanding, such as a transition or smooth blink if data updates or refreshes.

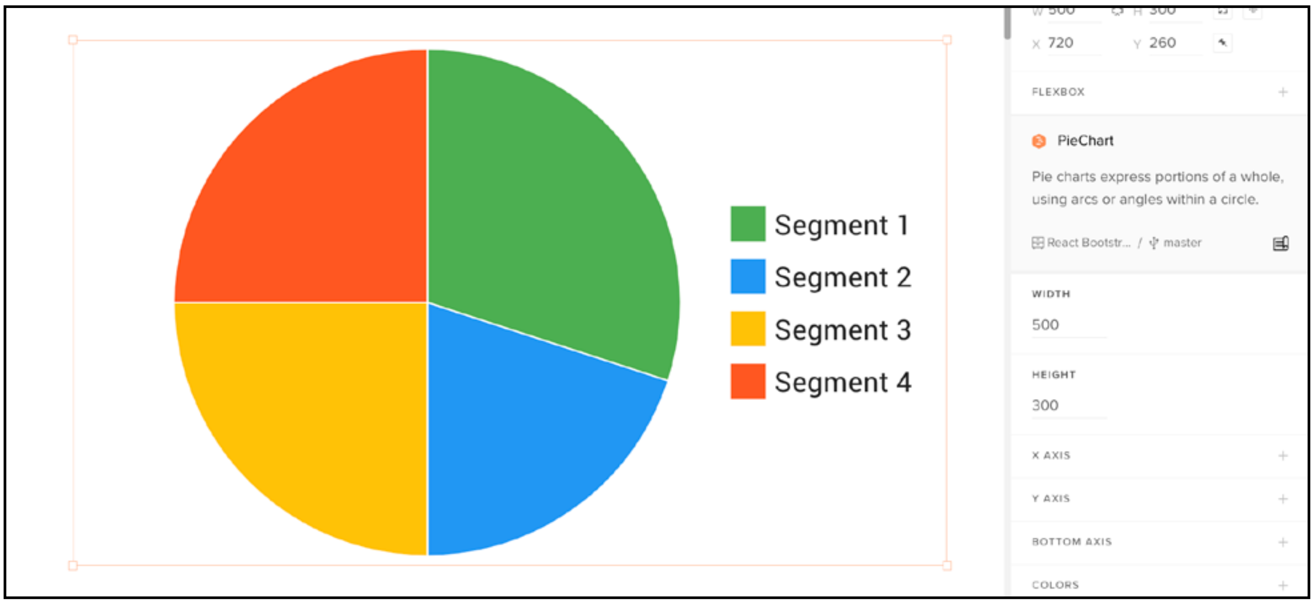
BUTTON: Include a download or export option.

How to Create a Pie Chart

To generate a pie chart using UXPin's AI Component Creator, follow these steps:

1. Open the AI Component Creator: Go to the Quick Tools panel on the left-hand side of the editor and click on the AI Component Creator.
2. Describe the Component: In the input field, provide a detailed description such as: "Generate a pie chart with customizable data segments, colors, and labels."
3. Choose a Library: Select a code library like MUI, Ant Design, or Bootstrap from the dropdown to ensure the chart aligns with your design system.

4 Generate the Chart: Click the "Generate" button. The AI will create a pie chart based on your input and place it on the canvas.

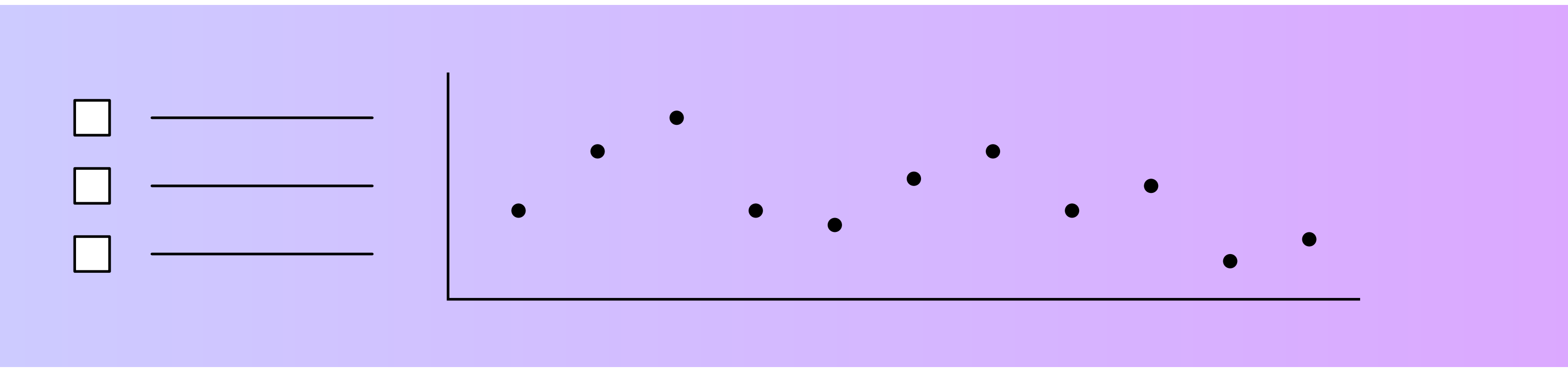


Pie charts express portions of a whole, using arcs or angles within a circle.

WIDTH 500

HEIGHT 300

5 Customize the Pie Chart: Once the pie chart is generated, you can modify its properties in the right-hand panel. Adjust data segments, colors, labels, and other visual properties to match your design needs.



PATTERN 3: GRAPHS

Graphs are a core component of data visualization in enterprise UX, used to communicate trends, comparisons, relationships, and patterns across large datasets.

Graphs have a wide range of practical applications and are flexible in that they can be used for simple or complex data sets. Unlike pie charts, which display one type of data, graphs have varying formats for different methods of organizing information.

Graphs are so commonly used because they help users digest complex data quickly, facilitating the decision-making process for everything from comparing two pairs of shoes to purchasing to choosing the right set of product features.

What Graphs Are Used For

1. Track change: Line charts are used to show trends and change over time. The most well-known application is graphs showing how the stock market has fared over time.
2. Track progress: Bar or column charts often compare different categories or sets of data or data sets with a comparison to time. Stacked bar charts or area graphs can show how individual components contribute to a whole. Histograms display information across a range and can be used, for example, to segment an email list by customers and sales.
3. Present relationship: Scatter or bubble charts are valuable for showing relationships between variables. Larger bubbles and more scatter points show more of a specific element versus smaller or fewer dots.

Best Practices for Graph Design

USE THE RIGHT KIND OF CHART: The graph style and type should match the type of information you need to display, making information easier to understand.

LABEL WITH PRECISION: Label the x-axis and y-axis, including units of measurement, and give each chart a clear title and subtitle, if needed. Show data labels on lines or bars if it aids comprehension in an open or hover state.

USE PROPORTIONATE DISPLAY AND SCALE: Always use proportional axes and avoid uneven scaling. Graphs should present data clearly and fairly.

USE GRIDLINES TO ENHANCE COMPREHENSION: Small lines can make tracking data across a graph easier. The design should be subtle so the graph does not look cluttered.

CONSIDER TOOLTIPS FOR EXTRA DETAIL: Provide extra information, such as exact values or additional context, when users hover over specific data points.

What Interactions to Use with Graphs

Because of the varying nature of chart types available to create here, there are a lot of different ways to interact with them. While most of these interactions will work on almost any graph type, some are better suited for certain styles than others.

FEEDBACK: When users hover over a data point, line, or bar, it should highlight or expand to indicate interactivity.

ACTION: Clicks can allow access to additional information based on the data point selected.

CONTENT MANAGEMENT: Filtering and search can make it easier for users to display exactly what they are looking for.

ACTION: A legend or key should be clickable or tappable and include an immediate interaction and connection to that data point on the graph.

ACTION: Allow users to zoom and pan for larger charts or those that include a lot of information.

CONTENT MANAGEMENT: Provide an animation to show that data has been refreshed or updated in real time.

CONTENT MANAGEMENT: For time-series or comparative graphs, offer options to display trends that show patterns or forecasting.

ACTION: Use hover states for tooltips or data labels.

BUTTON: Include a download or export option.

How to Create a Graph

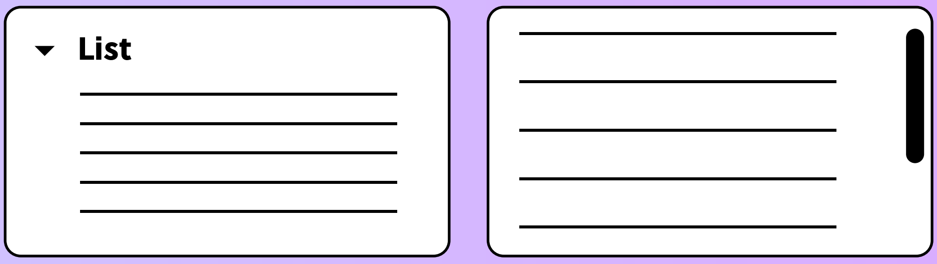
To create graphs in UXPin, you have two methods: using chart presets or coded components from libraries like MUIv5.

Using Chart Presets

1. Click on All Assets in the left-side Quick Tools panel or press Cmd/Ctrl + F to search. Type the name of the chart you're looking for, such as "Pie Chart," "Line Chart," "Bubble Chart," or simply "Chart" to browse all available presets.
2. Drag and drop the chart onto your canvas. Once on the canvas, customize it in the right-hand panel by adjusting properties like colors, shadows, and more to make the chart your own.

Using Coded Components (MUIv5)

1. Open the **Design System Libraries** in the bottom-left corner of the editor. Select the **MUIv5** library from the dropdown and search for graph components like "LineChart" or "BarChart."
2. Drag and drop the component onto your canvas, and use the right panel to adjust properties such as colors, labels, animations, and more. You can also add custom props to control the appearance and behavior of the graph elements.



PATTERN 4: LISTS

Lists are used to organize and display items, tasks, options, or datasets. They help create structure for almost any type of data or text entry. In enterprise applications, lists are critical for managing workflows, presenting filtered data, and allowing users to interact with items efficiently and intuitively.

Lists are a common device in website design because they send readability and scanability signals to search engines that your copy is clear and organized. Users can often read and comprehend this information quickly, contributing to a positive overall experience with your digital product.

Lists can make it easier to show blocks of text in a grouping for organization purposes, show products or items, indicate or help filter a database, or even trigger a notification system (such as checkboxes to note something is complete).

What a List is Used For

1. Make information organized: Lists are valuable in UX design because they organize information in a way that's easy to scan and digest. Whether a

simple to-do list or a complex product display, well-designed lists enhance usability and improve the overall user experience.

2 Give structure: A list differs from a data table in that tables have a specific structure, including a header, rows, and columns with sorting and filters to find and manipulate data. A list does not have a set structure and is often little more than a small header with text elements following it.

3 Help sort data: There are three types of **list design**: Text, Image, Card. These different types of list design may be used to display items, manage tasks, filter and sort data, select information from a larger dataset, or even facilitate navigation.

Best Practices for List Design

PRIORITIZE USER NEEDS: List design should provide appropriate fields and ordering for content and respond well on different devices.

FOLLOW DESIGN BEST PRACTICES: Lists should be organized logically, include actionable information that's easy to identify, and use a consistent layout for text, icons, and interactions.

MAKE LISTS SCANNABLE: Users should be able to read lists at a glance.

LEVERAGE VISUAL HIERARCHY: Use principles of strong UX design, including typography, color, spacing, and images, to ensure that the list has a logical flow.

ENSURE ACCESSIBILITY: Use a heading with an unordered or ordered list in conjunction with image-based lists so that screen readers can understand the content.

What Interactions to Use with Lists

The most common interactions with lists include subtle animations for feedback or interactivity that allow users to better understand more complex lists.

FEEDBACK: Lists should respond to hover events, especially in desktop applications. Highlight on hover to show interactivity.

ACTION: Include expandable/collapsible sections to manage content for long or complex lists.

ACTION: Checkboxes and radio buttons allow users to select and act on list items. Checkboxes can be used to select multiple items, and radios can be used to make a single selection.

ACTION: Scrolling and swiping allow users to perform actions, such as removing or selecting something from a list.

CONTENT MANAGEMENT: Reordering list items, often using drag and drop, gives users control over how they prioritize and experience data.

Filtering and search are similar functions that allow users to find and reorganize information.

ACTION: When dealing with large datasets, use infinite scrolling to keep the interface fast and responsive. New items load dynamically as the user scrolls.

BUTTON: Where appropriate, include action buttons (e.g., edit, delete, view details) or offer a context menu (usually opened by right-clicking or tapping).

FEEDBACK: When a user performs an action, such as marking a task as complete, provide immediate visual feedback, such as a checkbox, removal of the item, or strikethrough.

How to Create a List

To create lists in UXPin, you can use list presets or coded components from Merge libraries.

Using List Presets

1 Click on **All Assets** in the Quick Tools panel or press Cmd/Control + F to open the search bar. Type "List" to view all available list components like Contact List, Basic List Group, or Ordered List.

2 Drag and drop the list component onto your canvas. Once placed, customize the list in the right-hand panel by adjusting properties such as colors, fonts, and spacing to suit your design needs.

Using Coded Components (Merge Libraries)

1 Open the **Design System Libraries** in the bottom-left corner of the editor. Select a Merge library like MUIv5, React Bootstrap, or Ant Design.

2 Search for "List" in the selected library and drag the desired list component onto your canvas.

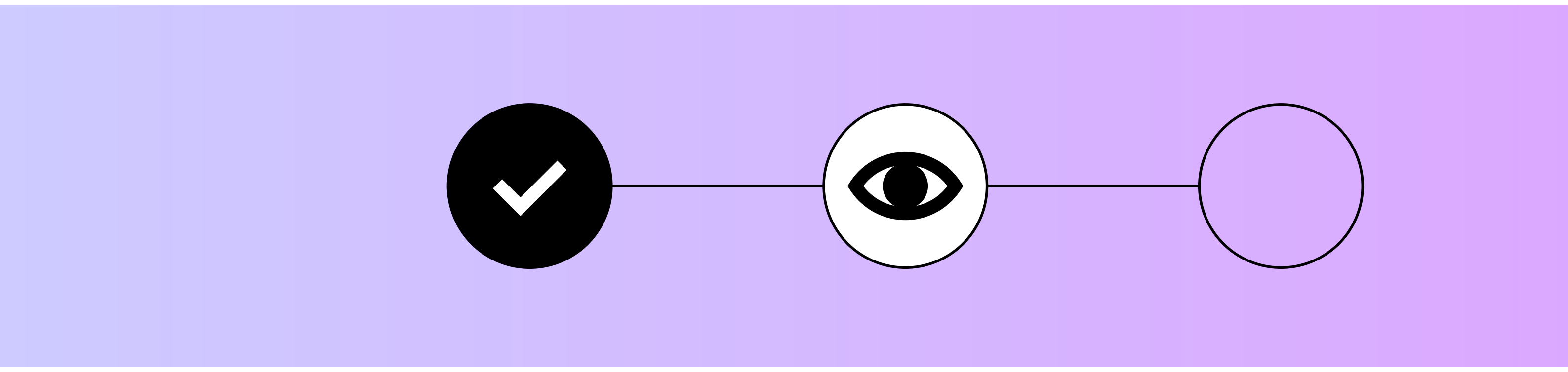
3 Customize the list's properties in the right-hand panel, adjusting options such as formatting, spacing, and additional behaviors to organize and display information effectively.

PROGRESS TRACKING PATTERNS

03

Progress trackers are design patterns that display the number of steps, the user's current status, and overall progress in completing a task. Progress trackers are particularly useful for completing forms with multiple stages, like eCommerce checkouts, insurance/medical onboarding, visa/passport applications, and other instances that require lots of data or inputs.

This interaction design pattern is different from a progress indicator in that a progress tracker is a step-by-step guide showing users where they are and what is remaining, while indicators are animated user interface patterns that indicate something is happening. A progress indicator is one example of a progress-tracking pattern.



PATTERN 1: WIZARD

Any guided workflow to complete tasks is known as a wizard. This name indicates a smart, or knowing, progress tracking element that can encourage completion or motivate users to finish steps in a task journey.

Wizard is used in enterprise applications to guide users through a complex or multi-stage process, such as on-boarding, form completion, or configuration workflows. By breaking such processes into manageable chunks, wizards provide clear progress tracking and help users focus, reducing cognitive load, preventing mistakes, and ensuring efficient completion.

What a Wizard Is Used For

1. Organize information: Wizards are powerful tools because they can simplify complex information. Their structured format is understandable, lets users know where they are on the path to task completion, and helps minimize confusion or abandonment.
2. Put order into chaos: In task management or project workflows, wizards guide users through tasks that must be completed in a specific order, such as setting up a new account or project. You've probably come across an onboarding wizard at some point from a tool that helped you access software or download and manage a digital product.
3. Guide users: Wizards are often part of a greater data entry or configuration setup using forms or multiple input fields. Use them to divide and explain complicated configurations into simple, step-by-step tasks.

Best Practices for Wizard Design

MAKE STEPS SIMPLE AND CLEAR: Each step should only ask for necessary information or actions. Label each step with a number and purpose.

USE A VISUAL PROGRESS INDICATOR: Show a progress bar or step indicator to let users know how far along they are in the process and how many steps remain.

ALLOW NAVIGATION BETWEEN STEPS: Wizards are designed to work in order, but a user may need to go back to change information.

DESIGN CONSISTENT LABELS AND BUTTONS: Create a style for labels and buttons and use them throughout the wizard. Common user interface elements for wizards include next, back, cancel, and submit.

SAVE PROGRESS: Display an obvious save button or enable auto-save functionality so users don't lose progress.

CREATE HELP DOCUMENTATION: Because wizards guide through complex steps, include a link to your help center or documentation for users who need additional information.

PLAN FOR USER ERROR: Display error messages inline with relevant fields, with instructions on how to fix it.

OFFER A FINAL REVIEW: Summarize all the information provided with the option to edit before final submission.

What Interactions to Use with Wizards

While wizards are more information design than interaction design, there are devices you can use to help ensure the completion of wizard tools. Almost every wizard interaction is a function that should lead to successfully completing a task.

ACTION: Users should be able to move between steps. Ensure that navigation logic is intuitive, with clear transitions between steps and a persistent progress indicator that updates dynamically.

CONTENT MANAGEMENT: Wizards should adapt based on user input. Use conditional logic to display or hide steps as needed.

FEEDBACK: Provide real-time validation, ensuring users can correct mistakes immediately before progressing to the next step. Highlight errors and provide suggestions for correction.

ACTION: Autosave to ensure users do not have to start over with complex tasks.

FEEDBACK: Provide a completion message that clearly indicates the process is finished. This can include the next steps or a summary of what was accomplished. Consider a fun animation to highlight the user's success.

How to Create a Wizard

To create a wizard in UXPin, you can break tasks into simple steps using separate pages for each stage. Here's how to do it:

Create a Wizard with Separate Pages

1. First, design your progress indicator or stepper to show the steps of the wizard.
2. Place the stepper on every page of the wizard to ensure consistency. As users progress through each step, update the highlighted step on the stepper to show their current position.
3. Add buttons to navigate between steps. In the Properties panel, click the + icon to add an interaction for each button. Set the trigger to "click," the action to "Go to page," and select the appropriate page to navigate to.
4. For efficiency, you can right-click a button, select **Copy interaction**, and paste it onto other buttons, adjusting the target page for each one.

Clickable Stepper Indicators

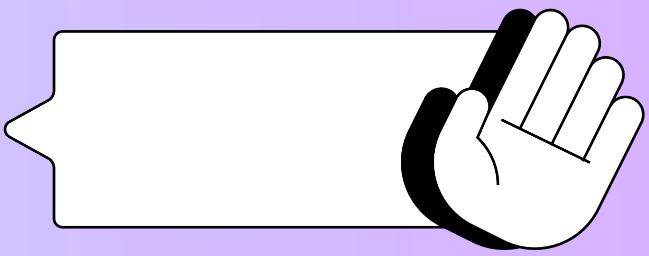
1. If you want the stepper itself to be clickable, follow the same process. Add interactions that allow users to jump to different steps by clicking the step icons.

Use the AI Component Creator for Speed

1. To streamline your wizard creation, use the **AI Component Creator** from the Quick Tools panel. Select your preferred library (e.g., MUI, Ant Design) and

craft a prompt like "Generate a multi-step wizard with navigation buttons and a progress bar."

2. Once generated, customize the pages and components in the right-hand panel. Don't forget to add interactions to each component to guide users through the process.



PATTERN 2: HELPBOX

Keep users on track with a helpbox component that offers support in real time, addressing confusion, and preventing errors. Generally, there are two types of helpboxes as progress tracking devices – static which is always present on the screen, or dynamic which appears when triggered by user actions or needs.

The more complex your data entry or user interaction, the more likely it is that a helpbox feature is needed as part of the design. When designed well, with a pattern that users understand–such as a ? icon or entering a search query–helpboxes can be an oft-used element.

Helpboxes are primarily paired with other user engagement tools, such as forms or wizards, and for elements requiring a lot of input, such as task management, project setup, system configurations, or onboarding.

What Helpboxes Are Used For

1. Guide users: A helpbox is exactly what you think it is—a tool to provide contextual assistance when a user gets stuck. They are often located and provide relevant information or explanations at every stage of a process, ensuring users understand requirements for each step.
2. Reduce cognitive load: At its core, a helpbox reduces cognitive load for a user, making an interface more approachable, while guiding users through complex processes. The key value is in that helpboxes provide immediate guidance without needing them to exit the task.
3. Clarify information: Helpboxes answer common questions and can help clarify instructions or prevent confusion in the user journey. Think of this feature as an embedded set of frequently asked questions that mitigates user mistakes. Additionally, helpboxes can be used to clarify terms, wording, or jargon specific to your product. In this use case, helpboxes may even look similar to tooltips.

Best Practices for Creating a Helpbox

USE CONTEXTUAL PLACEMENTS: Helpbox content should evolve as the user progresses through different stages of the task and positioning of the helpbox should pertain to the action where help is needed.

PROVIDE ACTIONABLE INSTRUCTION: Content should be short and actionable, with examples if needed.

PICK A DISPLAY TYPE: One display style—persistent or trigger-based—should be used throughout the interaction. Don't switch helpbox display types within a product.

USE A NON-INTRUSIVE PLACEMENT: While location of the helpbox should be clear, it should not cover or encroach on other user interface elements. Pop-up or floating helpboxes that hover near the user's action area are ideal.

KEEP IT CONSISTENT THROUGHOUT EACH STEP: The helpbox should be in the same location and styled similarly at each step of the process.

What Interactions to Use with Helpboxes

The most important interactivity with a helpbox is being there when users need it.

ACTION: When using a trigger-based option, users should be able to hover over an info icon or click on a specific element to reveal the helpbox.

BUTTON: For persistent helpboxes, use a common icon (?), label (help), or button to show location.

CONTENT MANAGEMENT: With multiple steps, helpbox content should dynamically update with user progress.

ACTION: Use tooltips for brief, targeted information.

ACTION: Interactive elements, such as clickable links to external guides or expandable sections for more detailed explanations, can improve user experience.

FEEDBACK: If a user makes an error or enters invalid data, the helpbox should tell users how to fix the issue.

ACTION: Allow users to close or hide helpboxes when they don't need them.

How to Create a Helpbox

To create a helpbox, you may want to design it in a traditional way or speed things up with AI:

Using the Box Tool

1. Select the Box tool from the Quick Tools panel or press the "B" shortcut.
2. Click and drag on the canvas to create your box. Once placed, double-click inside to add your message.

3 Customize the help box's properties in the right-hand panel. You can adjust font size, fill color, text color, and font family, and enable auto-size for a perfect fit.

Using Merge AI

4 Open the AI Component Creator from the Quick Tools panel.

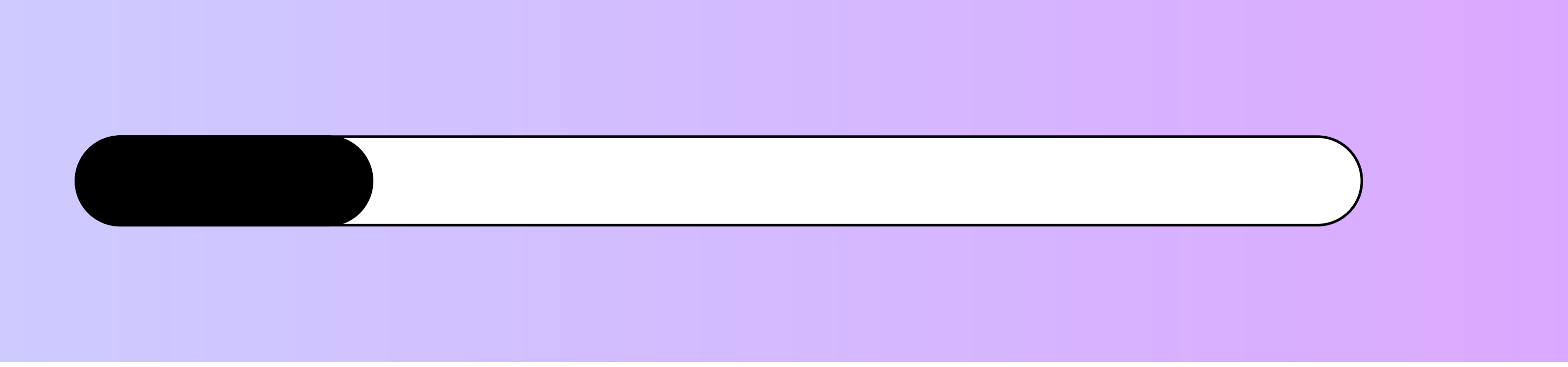
1 Input a prompt like, "Create a help box that provides tips for form completion with a title and descriptive text."

2 The AI will generate a basic help box component. Customize it in UXPin by adjusting size, colors, and text formatting in the right-hand panel.

Adding Interactions

1 Set up interactions for when and how the help box appears. For example, make it appear when a user hovers over an icon, and add a close button or set it to close when clicked outside the box.

TIP: Did you know that you can turn the components that you drew using Box Tool into code? You just need to enable AI Component Creator. Then, right-click on the component and transform it into code-backed one.



PATTERN 3: PROGRESS INDICATOR

While progress tracking isn't necessarily always in the form of a progress indicator, a progress indicator is always a progress tracking user design pattern.

Progress indicators are often lines or circles that show users the status of an ongoing event, such as loading a website or app, completing a form, or saving information. They communicate both the state of the app and what actions are currently available, such as continuing on a path, saving, or navigating away from a screen.

Progress indicators are a solid way to manage user expectations within a product interaction and maintain clear feedback about user status.

What Progress Indicators Used For

1. Create intuitive workflow: Progress indicators are essential in enterprise UX for guiding users through complex or multi-step processes by providing clear, real-time feedback on their status. By visualizing both completed and upcoming steps, progress indicators reduce uncertainty and create a more intuitive workflow. Properly integrating them can improve task completion rates.

2. Guide and motivate users: Progress indicators are used in wizards, forms, onboarding, and checkout processes to give users a sense of progression. They are not found in simple or single-step applications. No matter the shape—linear or curved—a progress indicator reduces user uncertainty while providing a sense of accomplishment, motivating users to continue.
3. Make tasks easy to follow: With tasks that include multiple stages, progress indicators help users understand and prepare for the steps ahead.

Best Practices for Progress Indicator Design

USE ANIMATION WITH CARE: Looped animations can work well for fast actions, but a percent-based or progress bar without motion is better for longer processes.

SEGMENT THE PROGRESS INDICATOR: Show clear stages or steps the system is completing or that the user is completing.

VISUALIZE COMPLETE, CURRENT, AND UPCOMING STEPS: Use color or another visual marker to note the difference between completed steps, current progress, and what happens next.

USE PROGRESS BARS FOR LINEAR TASKS: With forms, wizards, or other workflows that involve sequential steps, give users a continuous sense of progress.

USE STEP INDICATORS FOR NON-LINEAR TASKS: If users can jump between steps, show individual steps (rather than a continuous bar). Each step can be clicked to navigate between steps.

PROVIDE A TIMELINE FOR COMPLETION: Consider offering an indication of the effort required, such as "5 steps left" or "50% complete."

What Interactions to Use with Progress Indicators

Most of the interaction design associated with a progress indicator comes from the interface. While interactions result from user progress, they are not generally a direct engagement tool.

FEEDBACK: Progress indicators should provide real-time updates. As users move from one step to the next, the indicator should reflect the new progress, such as a color bar filling up.

ACTION: Include clickable steps that allow users to move back and forth along the progress indicator and to specific parts of the process.

ACTION: Add tooltips that detail each step as a hover state when users land on the progress indicator.

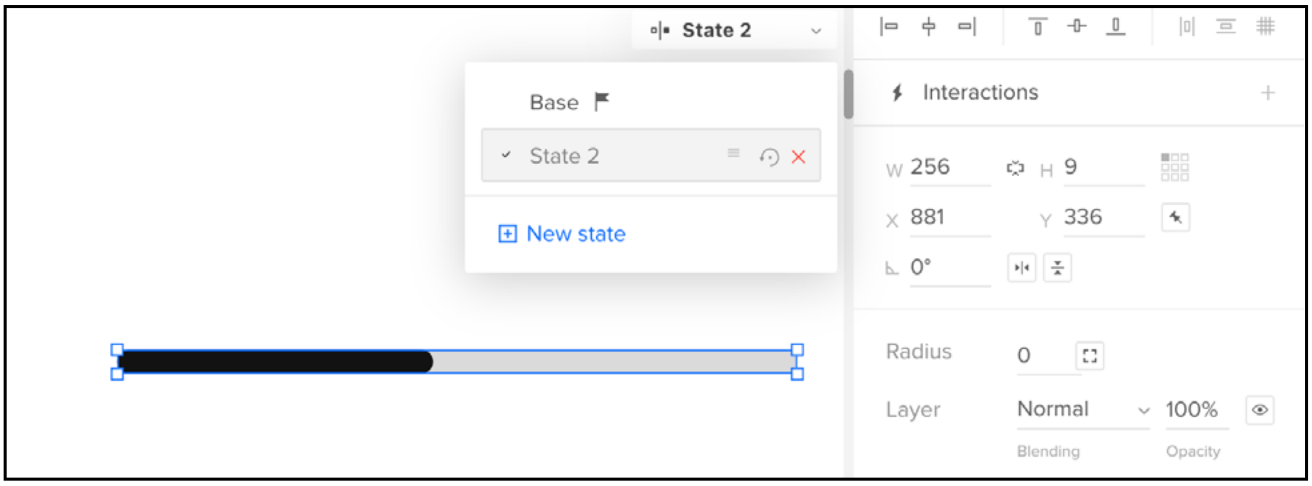
FEEDBACK: Provide a completion or submission message when the task is complete.

How to Create a Progress Indicator

Here's how to create progress indicators in UXPin, using two methods:

Using Shapes and States

1. Navigate to **Shapes** in the Quick Tools panel and select **Rounded**. Click and drag on the canvas to create the shape that will serve as the progress bar background.
2. Create another rounded shape in a different color to represent the progress.
3. Group the two shapes using the "G" shortcut or by selecting Group from the top bar.
4. With the group selected, click **Add State** or press Cmd + Shift + S. Create **State 1** as the base and **State 2** to represent 100% progress by extending the second shape.



5. To simulate progression, add an interaction. In the Properties panel, click the + button to add a new interaction, set the trigger to **On Page Load**, and choose **Set State** to transition between states. Customize the delay and duration, and select **Easing: Linear** for smooth progress.

Using Coded Progress Bar Components:

1. Access coded progress bars from design system libraries like **MUI v5** or **React Bootstrap v2**.
2. Open **Design System Librarie**s from the bottom-left corner of the editor, select your desired library, and search for **ProgressBar**.
3. Drag and drop the **ProgressBar** component onto the canvas.
4. Customize its properties, such as the **variant** (linear or circular), color, and classes. Use the **sx** property to apply custom styles with JSON5 syntax, adjusting elements like spacing, colors, and more.

NAVIGATION PATTERNS

04

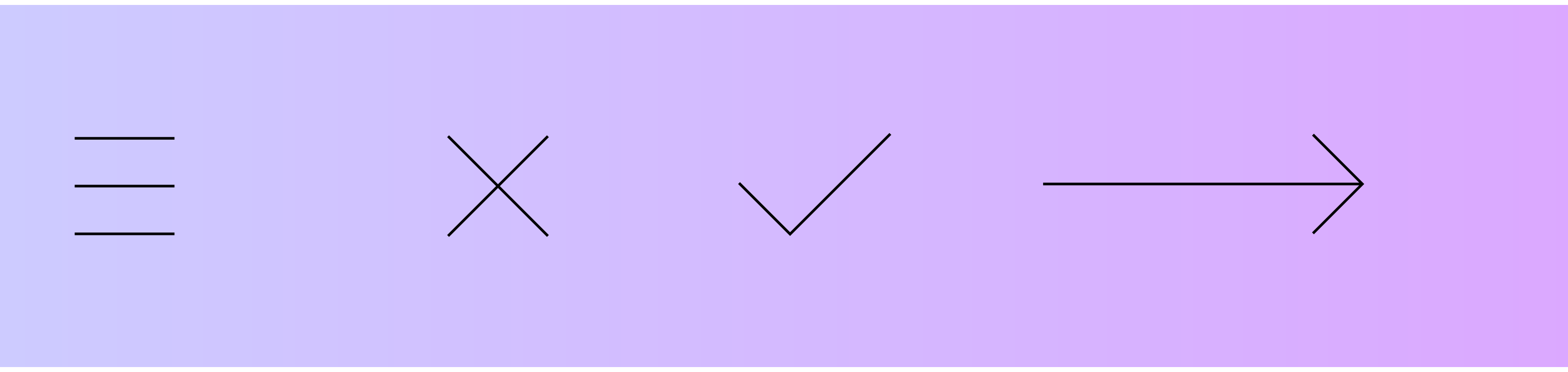
Navigation user experience can make or break your website. It influences everything from aesthetics to functionality and can be a determining factor in whether users understand your design and what you want them to accomplish.

When thinking about navigation, there are three golden design rules:

Use a mobile-first approach to streamline menus and reduce clutter.

Use trusted design patterns to ensure usability across devices.

Keep it simple with navigation prioritization and design.



PATTERN 1: ICONS

Icons are one of the most simple and important elements in a navigation pattern. It's because they are easily recognizable.

Think of how quickly you know what a cart icon does or what to do when you see an icon with a triangle inside a circle (play a video). These visual indicators enhance usability, improve cognitive recognition, and help users quickly identify their actions within a user interface.

Some icons have become so commonplace in navigation patterns that you probably don't think about them anymore. And this is a good thing. That means these icons have become a standardized element of navigation design patterns.

Common icons that are used in navigation include:

1. Hamburger menus
2. Buttons in navigation for a specific call to action
3. Kebab icons (three vertical dots that open into a larger list)
4. Doner icon (three horizontal lines stack in descending size to note filtering)
5. Cog icon for settings
6. Magnifying glass icon for search
7. Printer icon
8. Home icon
9. Shopping cart icon

What Icons Are Used For

1 Help locate information: Icons are visual elements used primarily to enhance visual recognition or bring attention to the most important element in a navigation menu. They improve scannability and comprehension so that users can quickly interact with an online product.

2 Convey meaning: Icons also have a few other purposes, including saving space in the design, often reducing the need for text-heavy navigation, allowing you to convey meaning in a compact form.

3 Create familiarity: Well-recognized icons also create familiar interaction models so that users innately know how a system works without thinking too hard about it. Icons guide users through actions and can help show the progression of where they have been and what's to come.

4 Grab user's attention: They also create a better visual experience that can make the interface more engaging and easier to use. For example, icons combined with color (red for alert icons or green for success icons) help create an intuitive experience.

Best Practices for Icons

TAKE CARE OF CONSISTENCY: Use a consistent design style, including size, color, line thickness, and alignment.

ADD CONTEXTUAL MEANING: Every icon should reflect the action or content it represents.

USE LABELS IF NEEDED: Combine icons with text labels for new or less obvious icons.

KEEP THE DESIGN SIMPLE: Icons should be simple and easily recognizable. Stick to simple shapes and clean lines.

ESTABLISH VISUAL HIERARCHY: Use size, color, and position to emphasize more important icons and de-emphasize secondary ones.

What Interactions to Use with Icons

Icons are often an interactive element in a design and paired with a click or tap action, much in the same way as a button. An icon may even be inside the frame of a button with a text element to show a click action.

ACTION: Users click or tap an icon to perform an action or navigate to a section.

ACTION: Hover states allow icons to change appearance or show additional information when users hover over them.

ACTION: Toggle icons can signify an on/off or singular change in a design, such as light to dark mode, an increasingly common navigation icon element.

FEEDBACK: A previously clicked icon changes state, such as color or having an underline, much like a link.

BUTTON: An icon in a button in the main navigation signifies an important menu element. Use no more than one per design.

How to Create an Icon Set

In UXPin, adding icons enhances both the visual appeal and functionality of your design. Here's how to do it:

Adding Icons

1. Open **Quick Tools** or press Cmd/Control + F and type "Icon" to search through UXPin's built-in icon library. You'll find a wide range of symbols and graphics.
2. Select the desired icon and place it on your canvas.

Customizing Icons

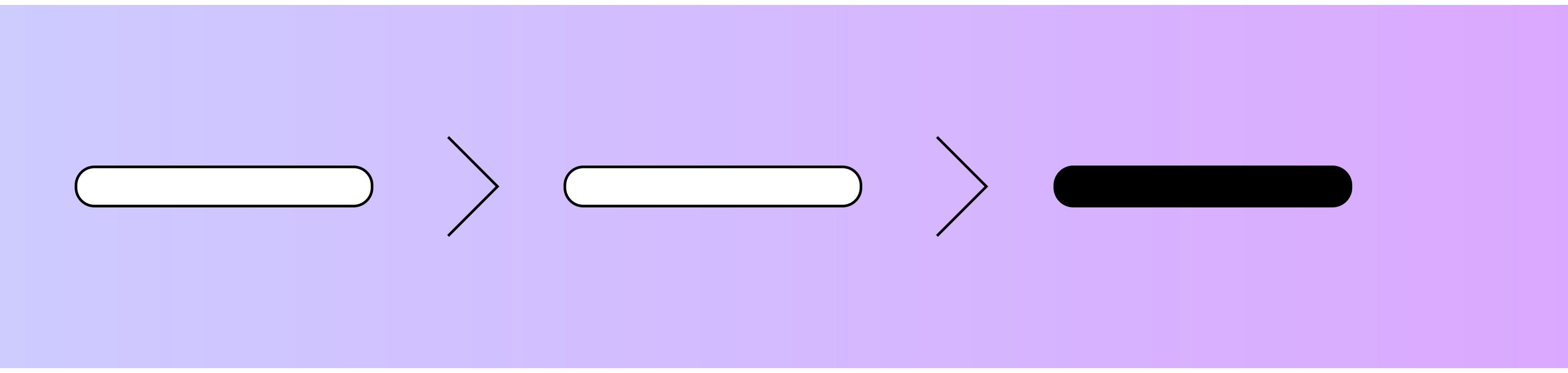
1. Once on the canvas, you can adjust the icon's size, color, and alignment in the **Properties Panel** on the right to match your design's style and color scheme.
2. This allows you to effortlessly fit the icon into your design's aesthetic and brand identity.

Using Icons with UXPin Merge

1. If you're using **UXPin Merge**, you can add icons from your component library to ensure design consistency across your project.
2. Open the **Merge Components** panel, search for the icon, and drag it onto your canvas. Changes made to the icon in the main codebase will automatically sync in UXPin, keeping your design updated and aligned with development.

Adding Interactions to Icons

1. In the **Interactions Panel**, you can assign actions to your icons, such as navigating between pages, displaying tooltips on hover, or activating popups.
2. This transforms icons from static visuals into dynamic, interactive elements that improve usability and engagement.



PATTERN 2: BREADCRUMBS

Breadcrumbs are a navigation pattern commonly used in complex systems to help users understand their location within a hierarchy of pages or data. They provide a visual trail of a user's path, making it easy to return to higher-level sections without starting over.

Often, breadcrumbs are designed as a series of links that show every page in the navigation tree. In addition to making a site easier to use, breadcrumbs can help search engines better understand and optimize your content.

Breadcrumbs can be location-based (showing where the user is in the site hierarchy), path-based (showing the path taken to reach the current page), or attribute-based (showing attributes or categories associated with the current content).

What Breadcrumbs Are Used For

1 Breadcrumbs show users their location: Breadcrumbs inform users of where they are within a website or system, helping them understand their current position in the content hierarchy.

2 Enable easy backtracking: Breadcrumbs provide a logical navigation structure that allows users to move back to previous sections without needing to rely on the browser's back button or guess their way through the site.

3 Simplify complex systems: Breadcrumbs are not only found on websites but are more commonly used in internal systems, management modules, and complex online tools for structured navigation.

4 Make navigation efficient: They minimize clicks and provide a more efficient way for users to explore content, leading to a better user experience.

5 Help visualize structure: Breadcrumbs allow users to visualize the overall

structure of a site or application, giving them a clearer understanding of its layout.

Best Practices for Creating Breadcrumbs

CREATE A CLEAR HIERARCHY: A breadcrumb trail should always start from the top-level page and show subsequent levels correctly.

USE SHORT, INTUITIVE LABELS: Ensure that each breadcrumb label is clear and descriptive, using familiar terminology. Try to keep labels to one or two words.

LINK ALL LEVELS, EXCEPT THE END: Each part of the breadcrumb (except the last/current page) should be clickable to allow easy navigation to higher levels.

PLACE BREADCRUMBS AT THE TOP OF PAGES: Breadcrumbs should appear near the top of pages, usually just below global navigation.

USE CLEAR SEPARATORS: Visual separators should appear between each individual breadcrumb and should not be confused with a letter or number. The most popular choice is an arrow (>), but pipes (|) and slashes (/) are also used.

What Interactions to Use with Breadcrumbs

While most breadcrumbs are simple links, some also include other user interface elements such as icons or tooltips. Use these additional interactions only when they enhance the usability of your product, not for purely visual purposes.

FEEDBACK: Use interactive icons to represent different levels or stages in the breadcrumb trail.

ACTION: Hover effects signal to users that text in a breadcrumb is a link.

ACTION: Allow users to click breadcrumb icons (in addition to text) for faster interaction.

ACTION: Display additional information or labels when users hover over breadcrumb icons or text.

How to Create Breadcrumbs

UXPin provides a flexible way to create and customize them, whether you're using the built-in tools or leveraging UXPin Merge for code-based components.

Adding Breadcrumb Components

1 Open **All Assets** in the Quick Tools menu or press Cmd/Control + F to search for "Breadcrumb."

2 Select components like **Breadcrumb Nav** or **Breadcrumb Item** and drag them onto your canvas.

3 If you're using **UXPin Merge**, access the **Design System Libraries** in the left panel and choose a library like **Material Design**, **Ant Design**, or **Bootstrap**. In these libraries, you can find **Breadcrumb** components under the **Navigation** section. Drag the breadcrumb component onto the canvas.

Customizing Breadcrumbs

1 Once on the canvas, customize the breadcrumb's color, font, spacing, and other design properties in the **Properties Panel** on the right.

2 This ensures your breadcrumbs fit seamlessly into your design, whether using native UXPin components or code-based elements from Merge libraries.

Adding Interactions

3 To make your breadcrumbs interactive, select each breadcrumb link and open the **Interactions Panel**.

1 Choose **Click** as the trigger and **Navigate** as the action, specifying the target page or section for each breadcrumb item. This creates a smooth navigation experience for users.

TIP: You can preview your design in UXPin by clicking on the "Get code" button, located on the top right corner of the editor.

PATTERN 3: APP BAR

An app bar is the essential navigation element on small screens. It can include everything from branding and navigation to screen titles and actionable elements. App bars are often contextual and can change throughout the user journey to work best with the rest of the content in the viewable area to enhance the overall experience.

It provides users with quick access to primary functions, navigation, and key actions. Positioned typically at the top of the screen, the app bar serves as an anchor point for navigation and offers visual structure and functional access.

App bars may remain sticky at the top during scroll or move with the content, depending on the design and use case.

What an App Bar Is Used For

1 Make smaller screens readable: They act as an interactive hub, offering core functions and primary navigation within the app, making them ideal for mobile and tablet interfaces.

2. Aid navigation: App bars typically include branding, primary navigation (often inside a hamburger icon), and tap-friendly icons for quick access to key features.
3. Contrasting design for visibility: App bars are usually styled in a color that contrasts with the content below, ensuring they stand out and are easily accessible.
4. Context-specific functionality: The app bar's purpose and displayed options can change depending on the page or context, providing action buttons like search, add-to-cart, or notifications specific to the current screen.
5. Provide account access: They often include icons for account management, allowing users to access settings, preferences, or sign out.

Best Practices for App Bar Design

ESTABLISH VISUAL HIERARCHY: Establish a clear visual hierarchy by grouping related items, making important actions prominent, and using consistent spacing.

PRIORITIZE KEY ACTIONS: Focus on the most important actions and relegate other links to the footer.

STICKY BARS CREATE CONSISTENCY: A sticky or fixed app bar remains visible as users scroll so they never lose track of where they are.

CUSTOMIZE CONTEXTUALLY: Modify the app bar based on the user's current page or actions, but maintain global elements such as branding, navigation, or login information.

USE A SIMPLE DESIGN: Keep the design clean and minimal to avoid distractions. When necessary, use icons, text labels, or both.

What Interactions to Use with App Bars

The biggest design challenge with app bars is that users work in a small space. Interactions have to be direct and clear to avoid getting lost on the small screen or hidden by other elements.

ACTION: Icons in the app bar are often tappable and can open menus, search, or start a new task.

FEEDBACK: Effects like color changes or tooltips provide extra context that a user has already interacted with an element.

ACTION: Icons can trigger dropdown menus or modal popups, offering additional actions or content. Just remember to include an easy way for users to close them!

FEEDBACK: Icons in the app bar can show badges or indicators to alert users of new messages, notifications, or updates.

ACTION: Expand the search icon into a full search field, allowing users to input queries directly in the app bar.

BUTTON: Interacting with a profile icon shows account-related actions, such as user settings, notifications, or sign-out options.

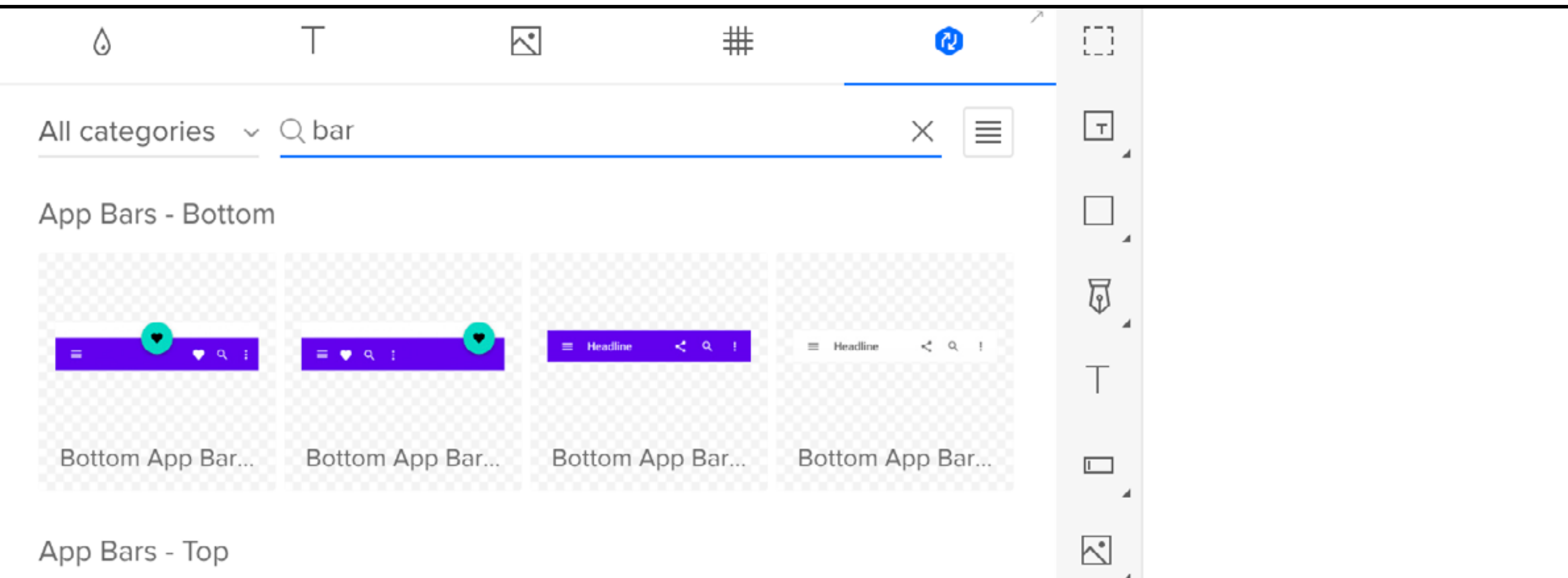
ACTION: To enhance user experience, use smooth animations, such as expanding menus or showing content with a fade-in effect.

How to Create an App Bar

Creating an App Bar in UXPin is easy and flexible, whether you're using pre-built libraries or generating components with the AI Component Creator. Here's how to do it:

Using Pre-built Libraries

1. Navigate to the **Classic Libraries** in UXPin, such as **iOS** or **Material Design**.
2. In the search bar, type "App Bar" or "Bar" to locate the relevant component.
3. Drag and drop the App Bar component onto your canvas.



Using the AI Component Creator

1. Open the **AI Component Creator** from the Quick Tools panel.
2. Choose the library you want to use (e.g., Material Design) and type a detailed prompt like "Generate an app bar with navigation icons and a title."
3. The AI will generate an App Bar component that you can place onto your canvas.

Customizing the App Bar

1. Once the App Bar is on your canvas, use the **Properties Panel** on the right to customize it.
2. Adjust colors, spacing, icons, and text to align with your design vision and brand.

Adding Interactions

1. To enhance functionality, select icons or buttons in the App Bar and open the **Interactions Panel**.
2. Set interactions like **Click** to trigger navigation or display additional information.

PATTERN 4: FOOTER

After the primary navigation, the footer is the most important navigation element even though it is located at the bottom of a website page or app. In enterprise applications and websites, the footer often contains secondary navigation links, utility functions, legal information, contact details, and other resources that do not require top-level prominence but are important to users.

What's great about a footer is that it can be large or small or include multiple levels of information. Some designs include a primary footer with relevant links for users and a much smaller, secondary footer for legal information and policies.

A user-friendly footer is simple, responsive, well-organized, and accessible. It may include icons, a basic form, and links for content or social media. Some footers also include navigation aids, such as a "Back to Top" button for longer-scrolling pages.

For the most part, a footer is the same throughout the design and provides a consistent, end-of-page navigation option to help users who have finished engaging with a specific page or piece of content but want more from your website or app.

What a Footer is Used For

1. Aid navigation: They often include primary or secondary navigation links, helping users find less prominent sections of a site or app. Footers are also a great place for additional features such as email newsletter signups, social media links, and external channels.
2. Accessibility: Some websites use the footer to display a full sitemap, ensuring all pages are accessible and easy to locate.
3. Provide business details: Footers are typically used for contact details, legal policies, and support information, which are beneficial for SEO and user convenience.
4. Displaying key information: Because the footer appears on every page, users often scroll to it first when searching for an address or phone number.

Best Practices for Footer Design

GROUP INFORMATION INTO CLEAR SECTIONS: Group similar items and organize them to reflect importance and function. Groupings might include legal information, contact details, and navigation links.

INCLUDE BRANDING AND CONTACT INFORMATION: The footer should include branding without overpowering the functional elements. Make sure to include NAP (name, address, phone) information for search engines and directories.

AIM FOR READABILITY: Keep the design simple, with easy-to-read fonts, sufficient spacing, and minimal clutter.

Create plenty of contrast between the background and links, and avoid making text too small.

LINK EVERYTHING: Place high-importance links prominently in the footer. Everything that can be linked should be, even if those links use a different style from the rest of the design.

MAKE IT ACCESSIBLE: Design the footer with proper contrast, keyboard navigation, and screen reader compatibility.

What Interactions to Use with Footers

Footer interactions should focus on navigability and quick access. If a user is looking for something in the footer, it likely means they were unable to find it elsewhere. Maintain trust and enhance the user experience by making every footer interaction simple and fast.

FEEDBACK: Use color change or underline to show that a link has been previously clicked.

ACTION: Hover effects can signify clickable items in the footer, such as a color change, underline, or enlargement of an icon or text element.

BUTTONS: Use an icon button for interactive elements such as online support or chat.

BUTTON: Include a "Back to Top" icon or button for long scrolling pages.

BUTTON: Use social media icons as buttons to link to external pages. Make sure they open in a new tab!

ACTION: Allow for collapsible or expandable footer menus on smaller screens. Use clear indicators to show sections that can be expanded or collapsed.

How to Create a Footer

Creating a footer in UXPin is simple and flexible, whether you use pre-built components or generate one with the AI Component Creator.

Using Pre-built Footer Components

1. Open **All Assets** from the Quick Tools panel or press Cmd/Control + F and search for "Footer."
2. Browse the available footer components and drag your chosen footer onto the canvas.
3. Customize the footer in the **Properties Panel** by adjusting the layout, adding/removing sections (such as social media icons or navigation links), and modifying colors, fonts, and spacing to match your design.

Using the AI Component Creator

1. Open the **AI Component Creator** from the Quick Tools panel.
2. Enter a prompt like "Create a footer with social media icons, contact information, and navigation links."
3. The AI will generate a footer component based on your description. Once on the canvas, you can customize it in the right-hand **Properties Panel**, adjusting colors, fonts, spacing, and layout to fit your design's style.

Adding Interactions

1. Make the footer functional by adding interactions. Select any link or icon in the footer, open the **Interactions Panel**, and set actions like **Click** to trigger **Navigate**, specifying the target page or section.
2. Repeat this process for all clickable elements to ensure smooth navigation.

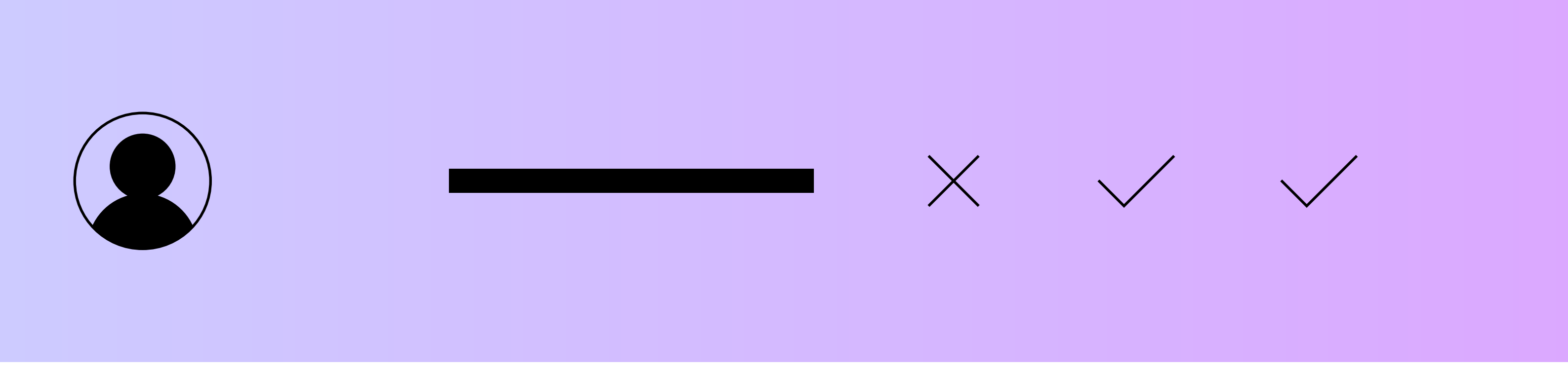
ACCOUNT MANAGEMENT PATTERNS

05

We're going to switch gears to one of the most frustrating design patterns that most of us encounter daily—account management. When done well, it's one of those things you don't even notice: login, perform an action, success! But poorly executed account management patterns make a mess, cause user frustration, and make you want to abandon the app or interface for something else entirely.

Account management patterns in enterprise UX can also include some pretty intense internal systems and forward-facing interactions with users or customers. The commonality is that managing user accounts in large-scale or complex systems is critical for ensuring that users can efficiently manage their accounts and tasks.

Many account management patterns you may use require third-party support or interactions, such as single sign-on (SSO) or multi-factor authentication (MFA, 2FA). In contrast, others can be done completely within your internal systems, such as role-based access or user profiles and preferences.



PATTERN 1: PERMISSION

Permissions in the context of account management user patterns are rules or settings that control what actions a user or group of users can perform within a system. They are essential for regulating access to data, resources, and functionality based on role, ensuring that users can only access information or perform authorized tasks.

Any system for which you have an account has user patterns for permissions, including internal systems, such as sales or inventory management or managing your company's website, and external systems, such as making a purchase online or playing fantasy football or another game online.

What Permissions Are Used For

1. Provide security: Permissions ensure only the right people access parts of a resource or tool that they are supposed to. Systems can have simple permissions—everyone is either logged out or logged in with the same set of rules—or complex permissions, with rules that vary by user and user group.
2. Manage users: Some examples of permissions include only allowing certain users access to records or files, such as a human resources manager having the ability to see employee files while other company members cannot.

Functional permissions can restrict the actions users can take within the system, such as being able to read and write files or only view them.

Best Practices for Permissions

USE ROLE-BASED ACCESS CONTROL: Assign permissions based on predefined roles, such as Admin, Editor, and Viewer, and grant only the minimum permissions required to perform tasks.

BREAK PERMISSION LEVELS INTO GRANULAR PIECES: Break down permissions into small, specific actions to offer more control and better align with roles.

ALLOW PERMISSIONS TO INHERIT TO LOWER-LEVEL OR CHILD OBJECTS: Allow child objects, such as files or subfolders, to inherit permissions from parent objects.

DESIGN TIME-BASED PERMISSION RULES: Grant permissions for a limited time and then allow them to expire automatically.

MAKE THE INTERFACE EASY TO MANAGE: If administrators can assign, modify, or revoke permissions easily, they are more likely to use rules and roles properly. This supports compliance by providing a clear record of who has access to what permissions were modified and when.

CREATE AN AUDIT TRAIL: Changes to permissions and users should be logged and easily reviewable.

What Interactions to Use with Permissions

Interaction design for permissions in account management focuses on creating clear, intuitive interfaces that allow users and administrators to manage permissions efficiently. The design must provide visual cues and affordances to help users understand what actions they can take, their current permissions, and how to make modifications. Interactions may include icons, labeling, color, layouts, and common controls.

ACTION: Use toggle switches or sliders to enable or disable specific permissions.

ACTION: Use dropdowns or radio buttons to allow administrators to select from predefined permission levels for new users.

ACTION: Use a form for a new user to request access or permission to a digital product.

FEEDBACK: Checkboxes can allow users to select multiple permissions at once and while acknowledging selections.

CONTENT MANAGEMENT: Display a grid or table format where permissions are presented in rows (or columns), and users can check boxes or use toggles to adjust permissions across multiple categories or users at once.

FEEDBACK: Use a confirmation dialog when users attempt to modify or revoke critical permissions.

FEEDBACK: Use color or icons to indicate various permission states, such as active, inactive, or restricted.

ACTION: Use commonly accepted icons for permissions interactions: lock for restricted access, eye for view only, pencil for editing, trash for delete, or key for admin access.

ACTION: Allow bulk interactions for administrators. Bulk actions, such as batch upload or selection of users, to apply the same permissions across multiple accounts in one action can be huge time savers at the enterprise level.

How to Create Permissions

Here's how the AI Component Creator can help you build a permission-based component:

1. Open the AI Component Creator:

    - Go to the Quick Tools panel and select AI Component Creator.

2. Describe the Component:

   - In the input field, write a detailed prompt such as: "Create a permission-based component that shows or hides information depending on the user's role. Include buttons to set roles and an area where content is hidden or displayed based on the selected role."

3. Generate the Component:

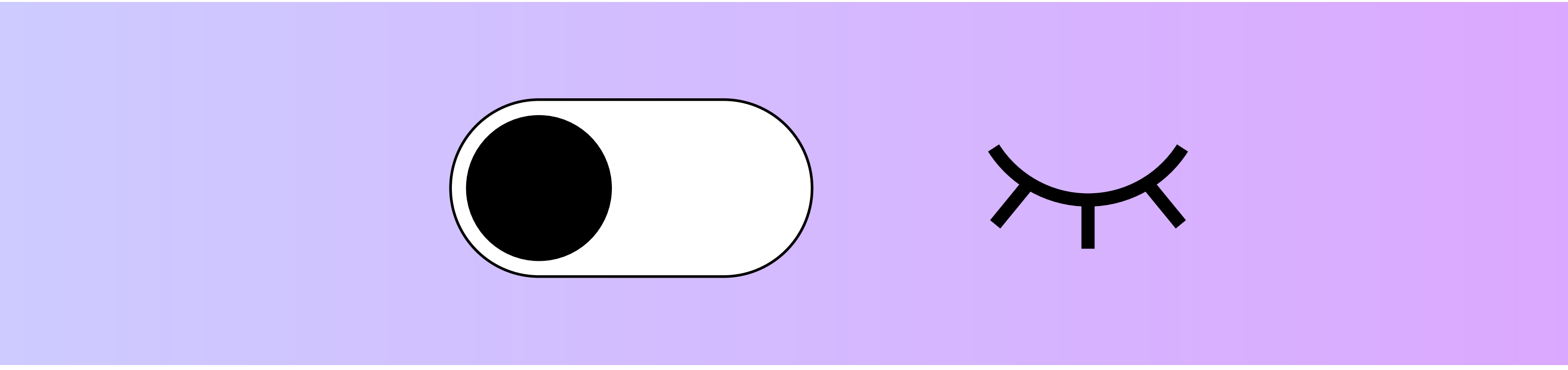
   - The AI will generate a component with the ability to assign roles and control content visibility. This component will likely include UI elements like buttons or toggles to switch between roles and a section that adjusts visibility based on those permissions.

4. Customizing the Component:

   - Once the component is placed on the canvas, use the Properties Panel to further customize it. Adjust styles, roles, and the logic for showing/hiding content.

5. Adding Interactions:

   - Open the Interactions Panel to set the conditions for hiding or showing content. You can set triggers like "Click" or "User Role Change" to toggle visibility.
   - Define the actions that correspond to specific roles, ensuring that certain users can only see what's intended for their access level.



PATTERN 2: HIDING INFORMATION

Hiding information in account management interfaces can be very valuable, especially for security, simplifying complex systems, and managing users more effectively.

While hiding information may seem like a secretive function, it is not. This can actually enhance the user experience by cutting features and options that will only clutter an interface in the path to streamlining paths to success and task completion.

Best practices for hiding information include role-based visibility, conditional displays, on-demand access, and intuitive interaction designs like collapsible sections, progressive forms, and hover-to-reveal actions.

Thoughtful interaction design ensures that users get the access they need without being overwhelmed or having access to sensitive data unless absolutely necessary.

What Hiding Information Is Used For

1. Prevent unauthorized access: Hiding information primarily serves to restrict access to sensitive data such as financial details, personal information, or proprietary content within an account management system.
2. Role-based data visibility: By hiding data based on user roles, only authorized users can view or interact with specific information, ensuring that content is accessible only to those with the appropriate permissions.
3. Enhanced security: Concealing non-essential or sensitive information minimizes security risks, preventing unauthorized users from accidentally discovering vulnerabilities.
4. Simplified interfaces: Hiding irrelevant options or features for lower-level users reduces complexity and confusion, resulting in a cleaner interface that improves focus and productivity for different user roles.

Best Practices for Hiding Information

CREATE CONDITIONAL VISIBILITY BASED ON ROLE-BASED ACCESS CONTROL: Display or hide elements based on conditions, such as user role, status, or permissions.

INCLUDE PROGRESSIVE DISCLOSURE: Hide advanced features and settings and reveal them as needed, such as when a user reaches a specific task or process.

IMPLEMENT GRANULAR PERMISSIONS FOR DATA VISIBILITY: Add finer control to data for users beyond specific roles so specific fields or sections can be hidden when unneeded. This might include conditional fields in forms or for other information.

USE SUBTLE VISIBILITY INDICATORS: Use subtle indicators, such as icons, to inform users that additional information or features are available. Consider different indicators for information hidden to make the interface easier versus information hidden due to access or permission level.

CONSIDER ON-DEMAND ACCESS TO HIDDEN INFORMATION: Allow information to be hidden by default but accessible upon request or through user authentication.

What Interactions to Use with Hidden Information

The most important interactions when it comes to hidden information are the ones that note more data is available or those that help users get access to the tools they need.

FEEDBACK: Use icons or symbols to indicate that more information is available. This might include a "+" or "More" notation when hiding for load purposes. This might include an icon or cue to log in when hiding for access purposes.

ACTION: When hiding sensitive information by default, such as password entry fields, allow the user to view the data to ensure success.

ACTION: Create progressive or conditional form inputs that show or hide information based on already entered data. For example, you probably don't need a shipping address for an order that will be picked up in-store, so you can hide that field.

CONTENT MANAGEMENT: Control search results only to include information for which the user has access.

CONTENT MANAGEMENT: Show or hide information based on the user's actions to ensure they see data relevant to their current task.

CONTENT MANAGEMENT: Use asynchronous loading to fetch hidden data only when needed, keeping the user interface light.

How to Create a Hidden Information Design Patterns

This feature works great when testing prototypes with users. If you want to temporarily hide a component, element, or group from the canvas, UXPin makes it simple:

Using the Layers Panel

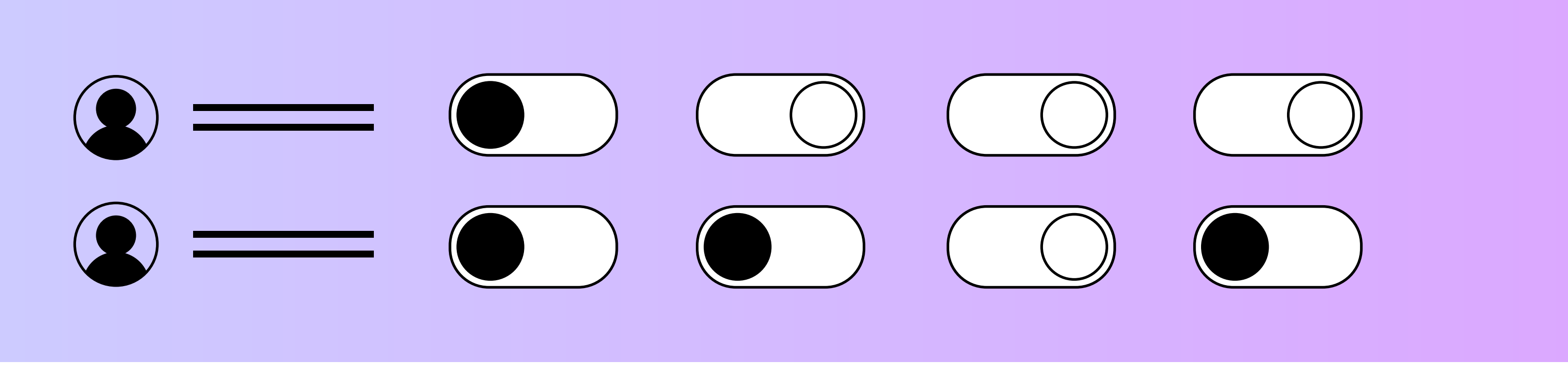
1 Select the component or group you want to hide.

2. In the **Layers Panel**, locate the eye icon next to the selected layer or group.
3. Click the eye icon to hide the component, making it invisible on the canvas.
4. This feature is especially useful when working with different states in an interactive design. For example, if you're creating a button with various states (default, hover, active), you can hide the default state while working on or previewing other states like hover or click.

Hiding Coded Components

1. For coded components, the process is the same. In the **Layers Panel**, click the eye icon next to the coded component's layer or group to hide it from the canvas.
2. This allows you to focus on specific elements or states without cluttering your design, making the workflow smoother and more efficient when creating interactive components.

TIP: You can test UXPin's prototypes with real users. Go to share mode, copy the prototype link, and send it to a user who can interact with the prototype without realizing it's a prototype, not a working product.



PATTERN 3: GRANTING ACCESS

While we've examined permissions and information management, granting access has its own set of user patterns that contribute to function and usability.

Granting access involves assigning permissions or roles to users that determine what resources, data, or functionalities they can interact with. This process is essential in enterprise systems to ensure the right people have access to the right information at the right level.

For internal systems, it allows for controlled sharing of sensitive data and enables collaboration while maintaining security and compliance.

For external or client-facing systems, it ensures that third parties can create accounts and save information but don't have access to internal systems, even if the system they engage with, such as an eCommerce platform, connects to an internal system like inventory.

What Granting Access Is Used For

1. Controls user access to resources: Granting access ensures that users only interact with tools and data relevant to their role, protecting sensitive information and preventing unauthorized actions.
2. Facilitates workflow management: Specific access levels can be assigned to delegate permissions for different features or content, streamlining collaboration and workflow management.
3. Supports compliance with data protection regulations: Defining permissions helps organizations comply with regulations like GDPR and HIPAA, ensuring sensitive data is only accessible to authorized users.
4. Enables personalization: Granting access for clients or customers creates personalized experiences based on purchases or account settings, providing a controlled and tailored user experience.

Best Practices for Granting Access

USE ROLE-BASED ACCESS CONTROL: Assign access based on roles within or outside your organization. Each role has a specific set of permissions and access rules.

USE TIME-BASED ACCESS: Grant access for a specific amount of time or during inactivity. Then, based on your present conditions, require users to log in again or reset passwords.

PROVIDE CLEAR COMMUNICATION: Be clear and direct about what users can and can't do within their access level.

From a client perspective, this might include limited time access for lower-level roles to content or software, while higher roles get unlimited access.

MONITOR ACCESS LOGS: Track and log when access is granted, by whom, and for what purpose. Regularly monitor access rights to ensure compliance with your policies.

What Interactions to Use with Granting Access

There aren't many bells and whistles here. Your interface and interactions for granting access should be simple and intuitive. Don't stray from common user patterns or labels here.

BUTTONS: Use common terms for access such as "Login," "Sign Up," or "Request Access."

ACTION: Use sliders or toggles to adjust a user's access level to a resource so administrators can make adjustments quickly.

ACTION: When granting access, implement additional security checks, such as two-factor authentication (2FA).

FEEDBACK: Provide a clear signal to let users know they are signed in, such as using their name and icon or a color "light" or toggle to show their active status.

FEEDBACK: Confirm new signups or logins so users know access has been granted.

FEEDBACK: Provide immediate notification when access is revoked. Bonus if you provide the why as well.

CONTENT MANAGEMENT: Create templates for common access scenarios to streamline the process.

CONTENT MANAGEMENT: Review and recertify granted access to ensure that users are active and should still be part of the system.

How to Grant Access in a Design

To design a component in UXPin that allows users to grant access based on interactions within the prototype, you can use the AI Component Creator. Here's how:

1. Open the AI Component Creator:
   - Navigate to the **Quick Tools** panel and select **AI Component Creator**.

2. Describe the Component:

    - Input a detailed prompt like: "Create a component that allows users to grant or restrict access to certain areas based on interactions. Include a button for granting access and a field that becomes visible only when access is granted."

3. Generate the Component:

    - Click **Generate**, and the AI will create a basic component based on your description. It might include buttons for granting or restricting access, along with fields or sections that respond to user interactions.

4. Customizing the Component:

    - Once generated, customize the component in the **Properties Panel**. Adjust the layout, labels, or fields to fit your design style and interaction needs.

5. Adding Interactions:

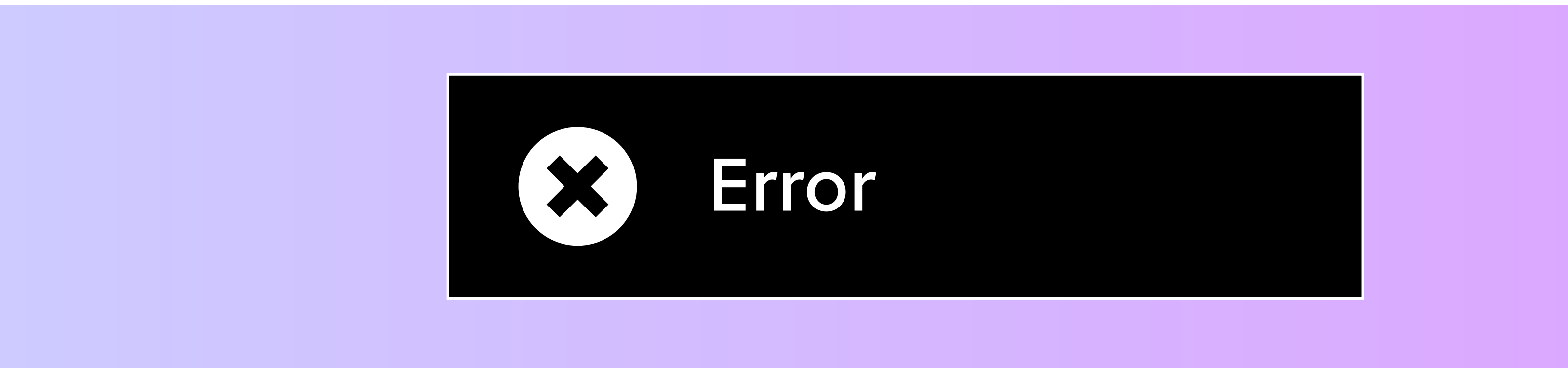
    - Select the button or trigger component (like a checkbox or toggle) and open the **Interactions Panel**.
    - Set interactions to allow users to grant access. For example:
        - Set the trigger to **Click**.
        - Choose **Set State** as the action and configure it to show a hidden section or unlock certain elements when the access is granted.

ALERT PATTERNS

06

Alert interaction design patterns focus on effectively notifying users of critical information while minimizing disruption to their workflows. Alerts must be clear, action-able, and contextual, particularly in complex enterprise environments.

You probably use patterns in your designs, including error messages, popups, modals, and snack bars. While each of these tools has a distinct and different purpose, it must be quick and easy to understand and have a strong informational component with next steps for users.



PATTERN 1: ERROR MESSAGES

Error messages are among the most common alerts that users encounter. A well-designed alter, using a common user pattern, helps the user correct their mistakes and keep them on a path moving forward. It ensures users are aware of what went wrong, why it happened, and how to fix it without disrupting their workflow unnecessarily.

When error messages are unclear or don't offer a solution, users will often abandon their work, making this a vital interaction for all online products.

Considering all of this, every error message or alert should include three things in the most succinct and efficient manner possible.

Description of the problem and what went wrong: This could include forgetting the @ in an email address or entering an incorrect password during login.

Solution to resolve the issue: A good error message tells users how to correct their missteps, usually in real-time, on the screen next to the problem.

Nondisruptive workflow progress: Users should be able to make the necessary corrections inline, without having to navigate away, and continue their progress.

What Error Messages Are Used For

1. Provide guidance for user errors: Error messages inform users when there's an issue with their input, such as form validation errors due to typos or incorrect formatting, and offer suggestions to help correct the problem.
2. Highlight permission or access issues: If a user encounters an error due to a lack of permissions, the message should indicate the restriction and provide next steps, like contacting an administrator for access.

3 Indicate system or backend failures: System errors occur when there's a backend issue or a request that can't be processed, often suggesting actions like retrying later.

4 Address dependency issues: Dependency errors inform users when actions are blocked due to missing required steps or incomplete processes, helping them understand how to proceed.

Best Practices for Creating Error Messages

WRITE CLEAR AND ACTIONABLE MESSAGES: Error alerts should explain what happened and how to fix it. Use non-technical language to make instructions easy to understand.

PRIORITIZE ERRORS: If there are multiple errors, rank them from most critical to least when providing feedback. You can also group errors by category if that enhances comprehension.

PLACE ALERTS NEXT TO THE ERROR: Place error messages where the issue occurred. A form error, for example, should show as an inline error next to the problematic field, not at the top of the form. You may also consider both placements.

USE A CONSISTENT STYLE: Error messages should not take over the screen unless they represent a critical, system-level issue and should use consistent visual cues that make sense to a user. Commonly, an error alert will show in red with an "x," for example, and may turn green with a "check" when corrected.

TAKE CARE WITH TONE: While detailing a mistake, ensure the error message copy does not blame the user.

ACCESSIBILITY MATTERS: Make sure error messages are accessible to screen readers and adhere to Web Content Accessibility Guidelines. Use ARIA roles for assistive technologies.

What Interactions to Use with Error Messages

Error messages should prioritize user-friendly interactions that help users proceed with their activity. This might include anything from validation to help prevent user frustration to a summary for an element with multiple errors.

FEEDBACK: Use inline error messages so that as soon as the user finishes interacting with a field, an error message displays next to it if incorrect information is present. Additionally, when the field is validated, the error should switch to a success message.

ACTION: Use toast/banners for minor errors that do not disrupt the workflow. A toast or banner is commonly applied to the top or bottom of the screen, and it either auto-disappears or can be dismissed by the user.

ACTION: Use modal error alerts (see Pattern 3) for critical errors that block the user from continuing or require immediate attention.

ACTION: Use an error summary to display multiple errors in a single element, such as a form. List all the errors and a method of resolution for each.

CONTENT MANAGEMENT: Use a persistent error banner at the system level for system-wide errors that impact functionality despite user input. The most common pattern is to place a banner across the tops of impact pages.

How to Create Error Messages

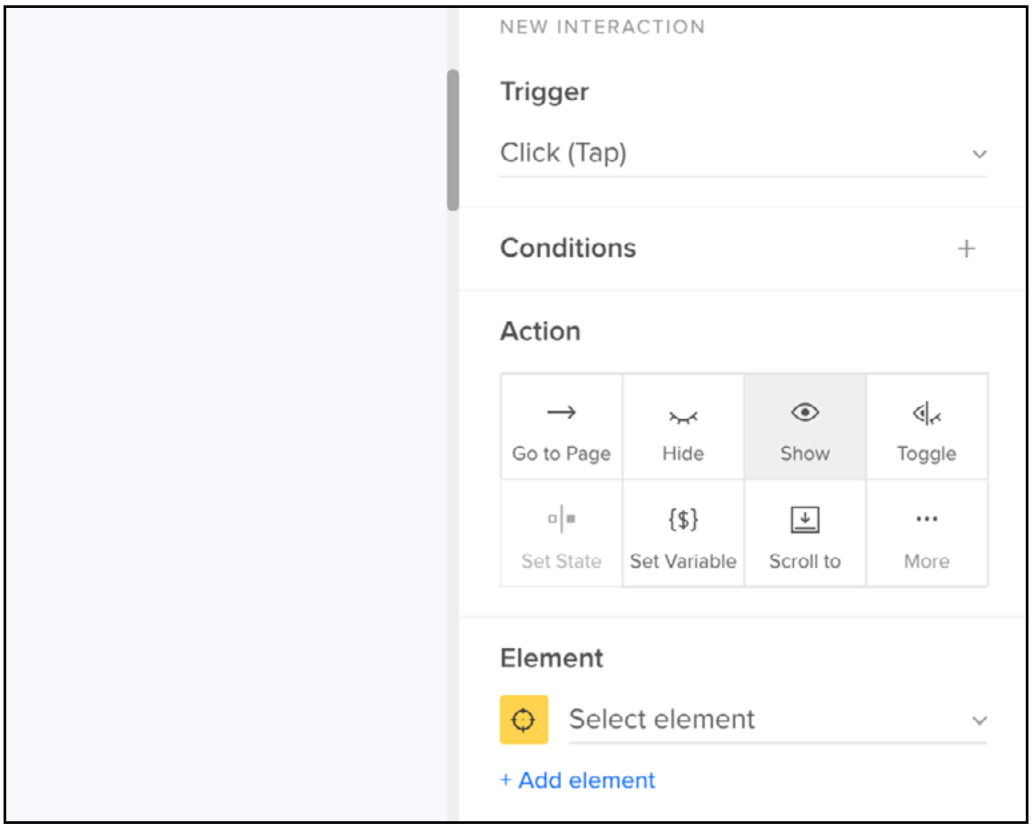
Creating and displaying error messages in UXPin ensures users receive clear feedback when they make mistakes. Follow these steps to design and activate error messages in your prototype:

1 Add the Error Message:

- Go to **Quick Tools** or use the shortcut Cmd/Control + F to search for "Text."
- Add a text box to your canvas and type in your error message (e.g., "Please fill out this field.").
- Customize the message using the **Properties Panel**. Adjust the text color to red (commonly used for errors), increase the font size, and add warning icons if needed to make the message more noticeable.

2 Set Conditions for Displaying the Error Message:

- Select the error message component on the canvas.
- Open the **Interactions Panel** and set a trigger. For example, use a **Click** trigger on a button or a Form Submission trigger.
- Set the action to **Show** so the error message will appear when an incorrect input or a missing field is detected.


NEW INTERACTION

Trigger: Click (Tap)

Conditions

Action: Show

Element: Select element

3 Add a Hide Interaction:

- For usability, create a second interaction to **Hide** the error message when the user corrects the input. You can set this trigger to “Change” or “Focus” on the input field.
- You can also set a timeout for the message to disappear after a few seconds if that fits your design.

PATTERN 2: POPUPS

By design, a popup is supposed to interrupt user engagement and their journey to show something important. Popups are most commonly used to show additional information pertaining to what a user is doing or to request additional action, but a user does not have to "do something" for a popup to appear.

They are both attention-getting and will interrupt users. This type of later user pattern should be designed to draw immediate attention with a single goal; provide critical information, such as an update, discount, or warning; and often require users to acknowledge the message before they can proceed. In the latter instance, closing the popup counts as acknowledgment in most instances.

In enterprise UX, use popups judiciously to avoid frustrating users and disrupting productivity while ensuring they effectively communicate high-priority information. The biggest concern with popups is that if they are used too frequently, they will become comparable to the story of the girl who cried wolf, and users will begin to ignore your popup regardless of its content.

What Popups Are Used For

1. Transactional popups: These popups are often used for promotional purposes, such as offering discounts in exchange for user information like an email or phone number.
2. Error and feedback popups: Popups are used to inform users of critical system errors or issues that require immediate attention, like network failures or data corruption.
3. Confirmation and warning popups: These alerts confirm actions or warn users before performing irreversible tasks, such as deleting accounts, often requiring a secondary confirmation to proceed.
4. Unsaved changes warnings: Popups can also warn users that they are about to leave or perform an action without saving, preventing potential data loss and ensuring a smoother user experience.

Best Practices for Creating Popups

LIMIT USE: Use popups only when necessary. Overusing this alert style can frustrate users and lead to alert fatigue, which can cause them to ignore them.

CRAFT COPY THAT'S CLEAR AND CONCISE: Users should immediately understand the alert's purpose and what to do next.

INCLUDE A CALL TO ACTION BUTTON: Use distinct and actionable button labels such as "Confirm," "Cancel," or "Save" to help users understand what to do. Use a button, rather than a text link, to make it overly clear how to interact with the popup.

OFFER A WAY TO EXIT: Give users a way out, such as an "x" to close the popup.

LIMIT TO ONE POPUP PER PAGE: Never layer multiple popups. If multiple issues need attention, queue alerts, resolve them in a single popup, or consider another alert pattern.

What Interactions to Use with Popups

Most popups are either transactional, with some type of input from a user; or informational, helping the user understand why something is or is not happeningt. The most important interaction for every popup alert design is the ability for the user to close or dismiss the popup module itself, unless action is absolutely mandatory to proceed.

ACTION: Use a popup that takes over the screen, dimming or disabling the background to prevent other actions. The user must explicitly confirm or cancel to proceed.

ACTION: Show information without blocking content, with a popup that can be dismissed easily, either with an "X" or "Dismiss" button or clicking outside the modal.

FEEDBACK: Use a popup to show that additional user input is needed, such as asking for missing information in a form.

FEEDBACK: Provide an error notification that requires attention from the user with a short description of the issue and an action to resolve it.

How to Create Popups

Creating popups in UXPin is a great way to provide quick information or guidance without disrupting the user's experience. Here's how to design and set up a popup:

1. Design the Popup Layout:

    - Go to **Quick Tools → Shapes** and add a **Rectangle** to serve as the background of the popup.
    - Position the rectangle in the center or any location that grabs the user's attention. Customize its size, color, and add shadows to make it stand out from the main interface.
    - Inside the rectangle, add elements like a message, call-to-action button, or icons.
    - Optionally, add a semi-transparent layer behind the popup to dim the background, drawing focus to the popup's content.

2. Group the Elements:

    - Select all the elements of the popup (e.g., background, text, buttons) and group them together for easier management.

3 Set Up Interactions:

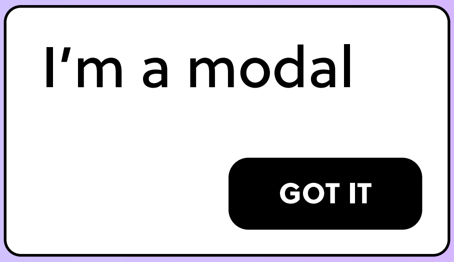
- Open the **Interactions Panel** and add a trigger to show the popup. This could be a button click, or you can set the popup to appear after a delay when the page loads.
- For closing actions, add interactions such as:
- Clicking a close button.
- Clicking outside the popup.
- Pressing the **Escape** key for a more user-friendly experience.

4 Test the Popup:

- Click **Preview** to ensure the popup appears and closes as intended.

5 Add Animations (Optional):

- For smoother transitions, you can add fade-in or slide animations to the popup, enhancing the user experience.



PATTERN 3: MODALS

A modal is a specific type of popup alert triggered by a specific user action. Like popups, modals appear on top of the main content on the screen and are used for important information or to enter user inputs.

A modal can appear at any point in a user journey and has a wide range of applications, from showing an announcement or notification to asking for user input and even application onboarding. It can look like a confirmation box, form, or simple notification.

A common element in this user pattern is that a modal will certainly ask a user to do something, such as click a button or link, and will not allow the user to proceed until the action is complete. Modals may be used for confirmations, warnings, or to gather inputs or data.

Modals can be frustrating for users, but when designed well, using expected user patterns, this element can be beneficial, ensuring that users engage with critical information or decisions.

If you are unsure whether a modal is the right choice for your digital product, ask yourself this question: Is user interaction required to proceed? If so, a modal may be the right solution.

What Modals Are Used For

1. Block interaction until closed: Modals display content on top of an overlay, preventing interaction with the main page until the user closes the modal or completes a required action.
2. Triggered by specific workflows: Modals are commonly used for tasks like adding or deleting users, sharing content, or entering data, providing users with focused interaction.
3. Binary choices and confirmations: Most modals present a binary choice, asking users to either confirm or cancel an action, such as deleting an item or confirming a submission.
4. Complex data entry tasks: Some modals support more detailed inputs, such as creating content or filling out forms, allowing users to complete actions without navigating away from the current page.

Best Practices for Modal Design

FOCUS ON CONTENT AND COPY: Modal content should be concise, clear, and actionable. If a user doesn't know what to do at a glance, you should rethink microcopy.

USE THEM SPARINGLY: Save usage for critical interactions that cannot be completed without user action.

PROVIDE A CLEAR BUTTON: Buttons should be easily identifiable and include a descriptive label.

ADD A HELP OPTION: If a modal involves a complex decision, offer help with tooltips, links, or collapsible sections.

TAKE CARE WITH TIMING: Present modals at the right time within the workflow and only when the user action determines it is necessary.

SIMPLIFY MANAGEMENT: Users should be able to dismiss modals by clicking outside the modal or with a close button unless the action is critical.

BLOCK BACKGROUND CONTENT: Prevent users from interacting with anything else until the modal action is confirmed or closed. This ensures that the user understands content in the modal is a priority.

What Interactions to Use with Modals

The same best practices you use for popups apply to modals. These commonly accepted user patterns are understandable for most users, helping to ensure a successful engagement with your digital product or design.

ACTION: Use a modal that disables the background to prevent other actions. The user acts to proceed.

FEEDBACK: Use a modal to request user input, such as asking for missing information or user authentication.

FEEDBACK: Use a modal to confirm required user attention with a short description of the issue and an action to resolve it.

CONTENT MANAGEMENT: Use a modal to confirm acknowledgment of system updates, disclosures, or information such as cookie or privacy policies.

How to Create a Modal

A modal is a great way to gather input or display critical information without navigating away from the current page. Follow these steps to design a modal in UXPin:

1. Design the Modal Structure:

   - Start by adding a **Rectangle** from **Quick Tools → Shapes** to serve as the modal's background. Ensure it's appropriately sized and centered on the screen.
   - Inside this rectangle, add elements such as text fields, buttons, or images for the modal content.
   - Customize properties like colors, borders, and shadows in the **Properties Panel** to make the modal visually distinct.

2 Create a Dimmed Background:

- Add another **Rectangle** behind the modal, covering the entire screen, and set its color to semi-transparent. This creates a dimmed effect that draws focus to the modal.

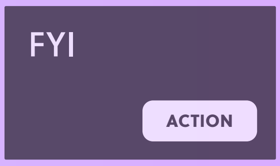
3 Group the Elements:

- Select all elements that make up the modal (background, content, etc.) and group them together for easier management.

4 Set Up Interactions:

- With the group selected, go to the **Interactions Panel** to define how the modal will appear and disappear.
- Choose a **trigger action**, like clicking a button or submitting a form, to show the modal.
- For closing actions, add interactions to hide the modal. This could be triggered by clicking a close button within the modal, clicking outside the modal area, or pressing the **Escape** key.

TIP: To enhance the experience, you can add fade-in or slide effects using the Interactions Panel to make the modal transitions smoother and more engaging.



PATTERN 4: SNACKBARS

A snackbar is a small, often temporary notification at the bottom or top of the screen that contains important information. Most commonly found in mobile apps, they include non-intrusive messages or updates. You will likely see and use snackbar alerts in desktop products as well.

Unlike some of the other alert patterns in this chapter, snackbars break out of the user flow to provide information, but don't generally require immediate attention.

The most common example of a snackbar is the **"Undo" action** after deleting an email. Often available as an action on email or desktop, the small alert user pattern appears and lasts for a few seconds, and if the user does not engage, it goes away on its own. Unlike a modal alert, snackbars do not require user engagement.

The primary purpose of a snackbar alert is to provide brief feedback on user interaction. The design is often nonintrusive and small. It doesn't include a lot of color or buttons, and often, the most desired action happens without user engagement.

Finally, a snackbar can be used to notify users of success, error, or progress updates without requiring immediate engagement.

What a Snackbar is Used For

1. Undo actions: The most common use for a snackbar is to provide feedback for an action that allows for quick, undoable interactions, such as undoing a delete or resetting a change.
2. Task completion notifications: Snackbars are effective for notifying users of successful actions, such as saving a document, submitting a form, or uploading a file, helping prevent duplicate entries and providing instant feedback.
3. Minor error and warning alerts: Snackbars can indicate minor errors or issues that don't require immediate attention, such as failed 2FA attempts or temporary system warnings, offering a low-interruption way to keep users informed without using modals or popups.

Best Practices for Snackbar Design

KEEP IT BRIEF: These alerts are small, and messaging should be as succinct as possible and direct.

USE AS AN INFORMATIONAL TOOL: Snackbars should not require user attention to proceed.

AUTO-CLOSE: Snackbars should automatically disappear after 3 to 5 seconds; maybe a little longer for an undoable action.

USE A STANDARD, NON-BLOCKING LOCATION: Place the snackbar in a consistent location throughout the design, typically at the bottom (material design) or top. Make sure the snack bar doesn't obscure critical UI elements.

USE COLOR FOR ADDITIONAL INFORMATION: Colors can enhance comprehension: Green or blue for success, red for error or failure, yellow or orange for warnings.

INCLUDE A LINK, IF NEEDED: If the snackbar involves an action, such as undo, make the action link clear and accessible, such as using a button or a text link that includes bold and underlined text.

What Interactions to Use with Snackbars

Snackbar interactions are pretty simple, with a choice to do something or ignore the user interface element altogether.

ACTION: A snackbar appears on the screen with a short message based on the user's actions and disappears after a few seconds.

ACTION: The snackbar appears with an ask for optional user engagement and disappears automatically with no engagement.

FEEDBACK: The user can click for a single action, such as "Undo."

CONTENT MANAGEMENT: Show a notification, confirmation, or update that does not require engagement.

How to Create a Snackbar

Snackbars are great for providing brief, unobtrusive feedback to users. Here's how to create one in UXPin:

1 Design the Snackbar:

- Add a **Rectangle** from **Quick Tools → Shapes** to serve as the background for the snackbar.
- Insert **Text** for the message and adjust the font properties (size, color, etc.) using the **Properties Panel** on the right.
- If you want to include an icon, use UXPin's Icon library from Quick Tools or upload your own. Position the icon next to the text, adjusting alignment and spacing for balance.
- Group the **Rectangle**, **Text**, and **Icon** for easier handling as a single unit.

2 Set Up Interactions:

- Select the grouped snackbar and open the Interactions Panel.
- Set a Trigger (e.g., button click or form submission) to show the snackbar.
- Add a Show action to make the snackbar appear, and then set a second

interaction for the snackbar to disappear after a delay (e.g., 3 seconds) by adding a Hide action.

3 Speed things up with AI:

- Open the AI Component Creator from the Quick Tools panel.
- Enter a prompt like: "Create a Material-UI snackbar that displays a message at the bottom of the screen when triggered, with customizable colors, text, and an automatic hide after 3 seconds."
- The AI will generate a basic snackbar component. Once on the canvas, customize it using the Properties Panel. Adjust the message text, colors, and position to suit your design.
- Set up interactions: choose a trigger (like a button click) to display the snackbar, and add a Hide action with a delay to automatically dismiss the snackbar after a few seconds.
- Test the snackbar in Preview Mode to ensure it functions as expected.

OUTRO

07

As we conclude our exploration of interaction design patterns, it's clear that these frameworks serve as essential tools for crafting functional and engaging user experiences. Moreover, we hope you've seen beyond what you can create and how creating in this manner could create efficiencies and better workflows for your team.

Throughout this eBook, we've delved into various categories of interaction design, each with a set of best practices. Understanding these patterns empowers you to create intuitive interfaces that enhance usability and drive user satisfaction.

1. **USER INPUT PATTERNS:** How to streamline form fields, rating systems, and buttons to optimize user engagement.
2. **DATA DISPLAY PATTERNS:** Techniques for effectively visualizing information through tables, charts, and lists, ensuring users can easily interpret complex data.
3. **PROGRESS TRACKING PATTERNS:** Methods like wizards and progress indicators that guide users through processes, reducing friction and improving completion rates.
4. **NAVIGATION PATTERNS:** Strategies to help users effortlessly navigate your applications using icons, breadcrumbs, and well-structured app bars.
5. **ACCOUNT MANAGEMENT PATTERNS:** Approaches for managing permissions and hiding sensitive information, ensuring users have the right access without feeling overwhelmed.
6. **ALERT PATTERNS:** Best practices for error messages, modals, and snackbars that keep users informed without disrupting their workflows.

By implementing these interaction design patterns, you can cultivate a more user-centric approach to product design. Consistency and familiarity enhance usability and reduce the learning curve for users, leading to increased productivity and satisfaction.

UXPin {M} merge

Scale your Design Operations

Bring designers and developers together. Use a single source of truth to remove product drift and increase design velocity. Try **UXPin** with **Merge technology** to have designers prototype with the same components devs build products with.

Discover Merge