

# Phase 6: User Interface Development

This phase covers Salesforce UI customization using Lightning App Builder, Lightning Web Components (LWC), and Apex integration. It enables building dynamic, user-friendly interfaces for Salesforce users.

## 1? Lightning App Builder

Purpose: Customize pages (Home, Record, App pages) using drag-and-drop components.

Steps:

- Go to Setup → Lightning App Builder.
- Click New → Choose Page Type (App Page, Home Page, Record Page).
- Enter Name & Description → Click Next.
- Choose Layout → Click Finish.
- Drag & drop standard or custom components.
- Click Save → Activate to make the page live.

Notes:

- Assign pages to Profiles or Apps.
- Use Preview to see the page before activation.

The screenshot displays the Salesforce Lightning App Builder interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar on the left contains 'Lightning App Builder'. The main content area features a 'Lightning App Builder' header with a gear icon. Below this, a descriptive paragraph states: 'The Lightning App Builder provides an easy to use graphical interface for creating custom Lightning pages for Salesforce Lightning Experience and mobile app. Lightning pages are built using Lightning components—compact, configurable, and reusable elements that you can drag and drop into regions of the page in the Lightning App Builder.' A 'View: All' dropdown and a 'Create New View' link are present. The 'Lightning Pages' table is shown with columns for Action, Label, Name, Namespace Prefix, Description, Type, Created By, and Last Modified By. The table lists two pages: 'Event\_Management\_App' and 'Event\_Record\_Page'.

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
<a href="#">Edit</a>   <a href="#">Clone</a>   <a href="#">Del</a>	Event_Management_App	Event_Management_App		App page for managing Events, Attendees, and Feedback	App Page	uid: 9/25/2025, 11:38 AM	uid: 9/25/2025, 11:47 AM
<a href="#">Edit</a>   <a href="#">Clone</a>   <a href="#">Del</a>	Event_Record_Page	Event_Record_Page		*Event Record Page – Manage Event Details and Related Info*	Record Page	uid: 9/25/2025, 12:03 PM	uid: 9/25/2025, 12:03 PM



## Lightning Page

[Help for this Page](#) 

Edit Clone Delete

Name	Event_Management_App	Label	Event Management App
Description	App page for managing Events, Attendees, and Feedback		
	<div> <div>Edit</div> <div>Clone</div> <div>Delete</div> </div>		

App
Event Management App
Sales

Purpose: Customize object record layouts.

Steps:

- Open Lightning App Builder → Record Page.
- Select Object → Existing Page / New Page.
- Configure Sections and add components: Related lists, Custom LWCs, Tabs.

- Click Save → Activate → Assign to App, Record Type, or Profile.

The screenshot displays the Salesforce Lightning App Builder interface. At the top, there are navigation buttons: a back arrow, a forward arrow, a trash icon, a copy icon, and a save icon. Below these are dropdown menus for 'Desktop' and 'Shrink To View', followed by 'Analyze', 'Activation...', and a blue 'Save' button.

The main interface is divided into three sections:

- Components:** A sidebar on the left with a search bar and a list of standard components (42 total). The list includes: Accordion, Action Launcher, Actions & Recommendations, Approval Trace, Assessment List, CRM Analytics Collection, CRM Analytics Dashboard, Dynamic Related List - Single, Einstein Next Best Action, Flow, Flow Orchestration Work Guide, Highlights Panel, Invoice Preview, Launchpad, and List View.
- Fields:** A central area showing a preview of the page layout with various components and fields.
- Page:** A right-hand panel for configuring the page details.
  - \* Label:** Event Record Page
  - \* API Name:** Event\_Record\_Page
  - \* Page Type:** Record Page
  - Object:** Event
  - Template:** Header and Left Sidebar (with a 'Change' button)
  - Description:** "Event Record Page – Manage Event Details and Related Info"
  - Enable page-level dynamic actions for the Salesforce mobile app:** A checkbox that is currently unchecked.

At the bottom, there is a 'Setup' bar with links to 'Setup', 'Home', and 'Object Manager'.

Below the main interface, there is a 'Lightning App Builder' section. It includes a search bar for 'Lightning App Builder' and a list of user interface components. A table titled 'Lightning Pages' shows the following data:

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
<a href="#">Edit</a>   <a href="#">Clone</a>   <a href="#">Del</a>	Event Management App	Event_Management_App		App page for managing Events, Attendees, and Feedback	App Page	ytd: 9/25/2025, 11:38 AM	ytd: 9/25/2025, 11:47 AM
<a href="#">Edit</a>   <a href="#">Clone</a>   <a href="#">Del</a>	Event Record Page	Event_Record_Page		"Event Record Page – Manage Event Details and Related Info"	Record Page	ytd: 9/25/2025, 12:03 PM	ytd: 9/25/2025, 12:03 PM

### 3 Tabs

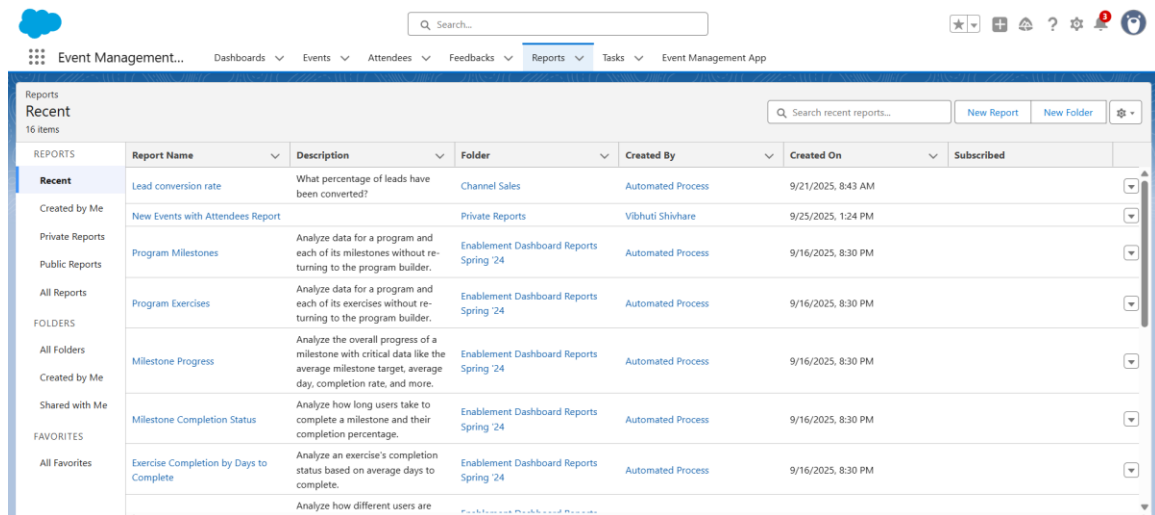
Purpose: Organize objects, components, or pages in a single app.

Steps:

- Go to App Manager → Edit App.
- Under Navigation Items, click Add Tab.
- Select Standard Object / Custom Object / Lightning Page.
- Save changes → Open app to verify tabs.

Notes:

- Control tab access using App Personalization Settings.
- Tabs can include Lightning Components, Visualforce Pages, or Reports.



## 4? Home Page Layouts

Purpose: Create dashboards or personalized home pages.

Steps:

- Go to Lightning App Builder → Home Page.
- Click New → Standard Home Page.
- Drag & drop components: Reports, Dashboards, Custom LWCs.
- Save → Activate → Assign to Profiles.

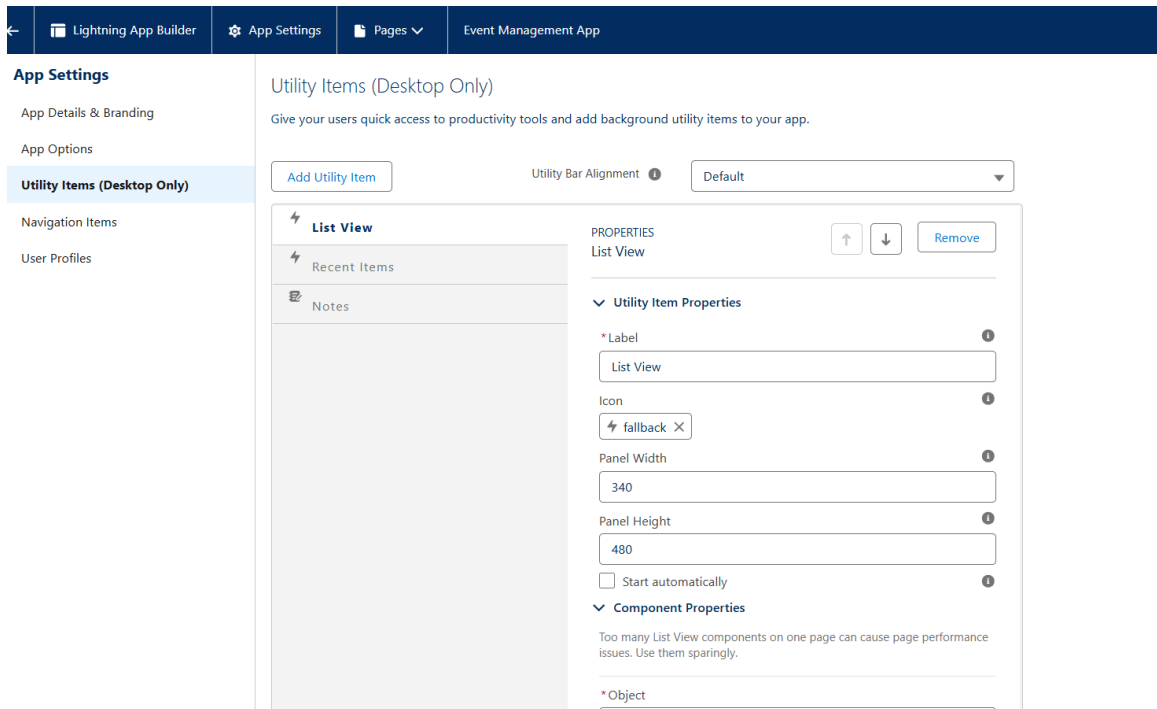
## 5? Utility Bar

Purpose: Quick access to tools at the bottom of the app.

Steps:

- Open App Manager → Edit App → Utility Bar.
- Click Add Utility Item.
- Choose component (e.g., Notes, Custom LWC, Flows).
- Configure Label, Icon, Size.

- Save & verify in app.



## 6 Lightning Web Components (LWC)

Purpose: Build reusable, responsive UI components.

Structure:

- .js → Component logic
- .html → Template
- .js-meta.xml → Metadata

Steps:

- Open VS Code → SFDX project → Create LWC.
- Implement logic & styling.
- Deploy using SFDX: Deploy.

Notes:

- LWCs can be used in App Builder, Record Pages, Tabs, or Utility Bar.

## 7 Apex with LWC

Purpose: Fetch or manipulate Salesforce data using Apex.

Steps:

- Write @AuraEnabled Apex methods.
- Call methods in LWC using Wire Adapter (reactive) or Imperative Call (on user action).
- Handle responses & errors in LWC JS.

## 8? Events in LWC

Purpose: Handle component communication.

Types:

- Custom Events → Child → Parent
- Lightning Message Service → Cross-component communication
- Standard DOM events → Click, Change, Input

Steps:

- Create & dispatch CustomEvent in child component.
- Listen in parent with on<eventname> attribute.

## 9? Wire Adapters

Purpose: Retrieve Salesforce data reactively.

Examples:

- @wire(getRecord, { recordId, fields })
- @wire(getObjectInfo, { objectApiName })

Notes:

- Wire automatically updates UI when data changes.
- Mostly for read-only operations.

## ? Imperative Apex Calls

Purpose: Call Apex methods on-demand.

Steps:

- Import Apex method in LWC JS.
- Call method inside a function (e.g., button click).
- Handle Promise: myApexMethod({ param1: value }).then(result => { /\* handle result \*/ }).catch(error => { /\* handle error \*/ });

Notes:

- Good for user-triggered actions.

- Gives explicit control over when the call happens.

### ✓ **Best Practices**

Keep LWCs modular and reusable.

Always handle errors in Apex calls.

Use Profiles / Permission Sets to control access.

Test Home Page and Record Page layouts for multiple profiles.

Deploy using Change Sets or SFDX for version control.

### **I mainly Concentrated on**

- **Lightning App Builder**
- **Record Pages**
- **Tabs**
- **Home Page Layouts**
- **Utility Bar**