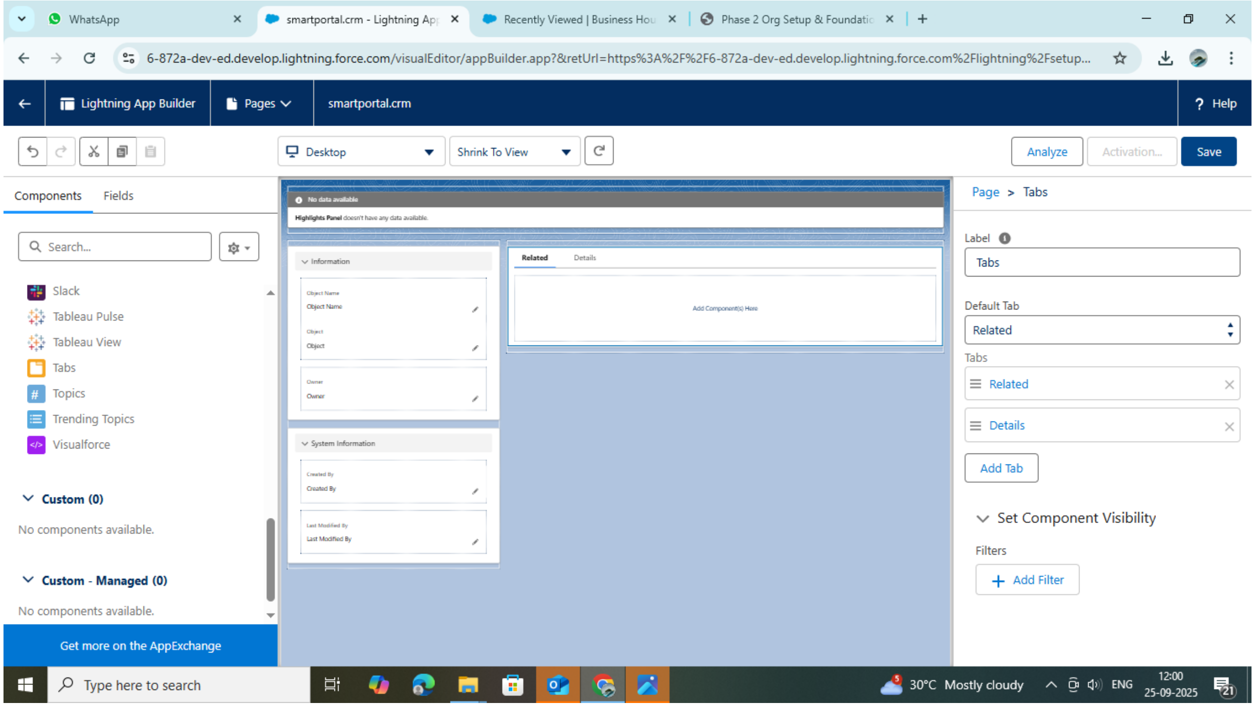
Phase 6: User Interface Development

This phase focuses on building a user-friendly interface in Salesforce for the Smart Property Portal using Lightning components and other UI tools.

Step 1: Lightning App Builder

Go to Setup → Lightning App Builder.

Create or edit apps to include tabs, pages, and components for Sales Agents and Managers.



6-872a-dev-ed.develop.lightning.force.com/visualEditor/appBuilder.app?&retUrl=https%3A%2F%2F6-872a-dev-ed.develop.lightning.force.com%2Flightning%2Fsetup...

Page > Tabs

Label: Tabs

Default Tab: Related

Tabs: Related, Details

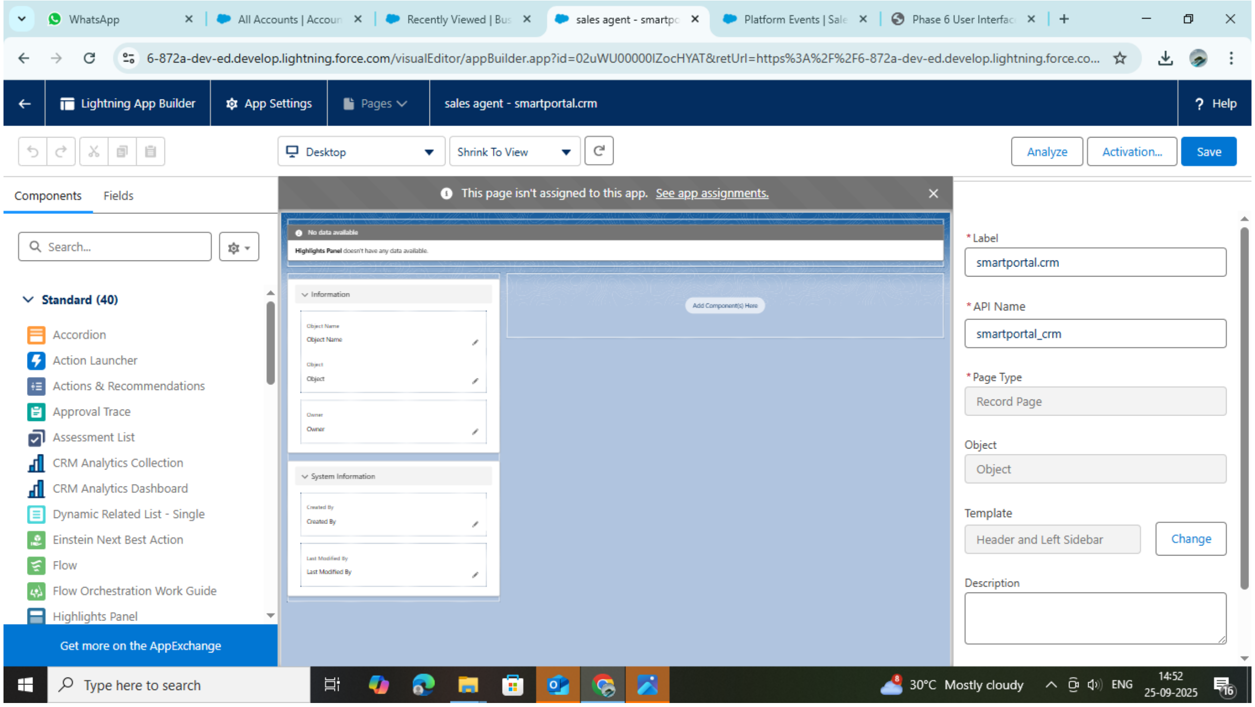
Step 2: Record Pages

Customize Lightning Record Pages for objects like Account, Property, and Visit.

Arrange components to show related lists, highlights, and key fields.

Step 3: Tabs

Setup → App Manager → Edit App → Add tabs for Accounts, Properties, Visits, and Reports.



6-872a-dev-ed.develop.lightning.force.com/visualEditor/appBuilder.app?id=02uWU00000lZocHYAT&retUrl=https%3A%2F%2F6-872a-dev-ed.develop.lightning.force.co...

This page isn't assigned to this app. See app assignments.

No data available

Highlights Panel doesn't have any data available.

* Label: smartportal.crm

* API Name: smartportal_crm

* Page Type: Record Page

Object: Object

Template: Header and Left Sidebar

Description:

Step 4: Home Page Layouts

Design Home Pages with key metrics, charts, and shortcuts.

Include components such as Recent Records, Custom Links, and Dashboards.

Highlight KPIs relevant to Sales Agents and Managers.

Add personalized components using dynamic visibility rules.

Test home page layouts for different profiles to ensure user-specific customization.

Step 5: Utility Bar

Add utility bar items for quick actions like creating a new Property, logging a Visit, or checking Notifications.

Include tools like calculators, quick links, or recent items for convenience.

Customize icons, labels, and order to optimize usability.

Test utility bar actions to ensure they perform correctly across different pages and profiles.

Step 6: Lightning Web Components (LWC)

Develop custom LWC components for specific UI needs.

Example: Property Card, Lead Details Panel, Visit Scheduler.

Ensure responsive design for mobile and desktop.

Test components thoroughly with sample data.

Step 7: Apex with LWC

Use Apex methods to fetch or update data from LWC.

Example: Create an Apex class to get property list and call it from LWC.

Handle exceptions gracefully to avoid errors in the UI.

Ensure proper security by enforcing CRUD and FLS in Apex methods.

Step 8: Events in LWC

Use custom events to communicate between parent and child components.

Example: Trigger a parent refresh when a new Property is added.

Test event propagation and handlers for correct behavior.

Step 9: Wire Adapters

Use @wire to retrieve data from Salesforce declaratively in LWC.

Example: @wire(getRecords) to display a list of Properties

Handle loading states and errors to improve user experience.

Step 10: Imperative Apex Calls

Call Apex methods imperatively when you need control over execution.

Example: Save a Property after user clicks a button.

Manage promise resolution and errors properly.

Test edge cases to ensure reliable performance.

Step 11: Navigation Service

Use Lightning Navigation service to programmatically navigate users to pages, records, or components.

Example: Navigate to a Property record after creation.

Handle URL parameters and dynamic navigation for flexibility.

Test navigation actions across different profiles and devices.