Phase 6: User Interface Development – HealthCare360

---

1. Objective

To design and implement a healthcare-optimized user interface using Salesforce Lightning tools, enabling seamless management of patients, appointments, and insurance workflows.

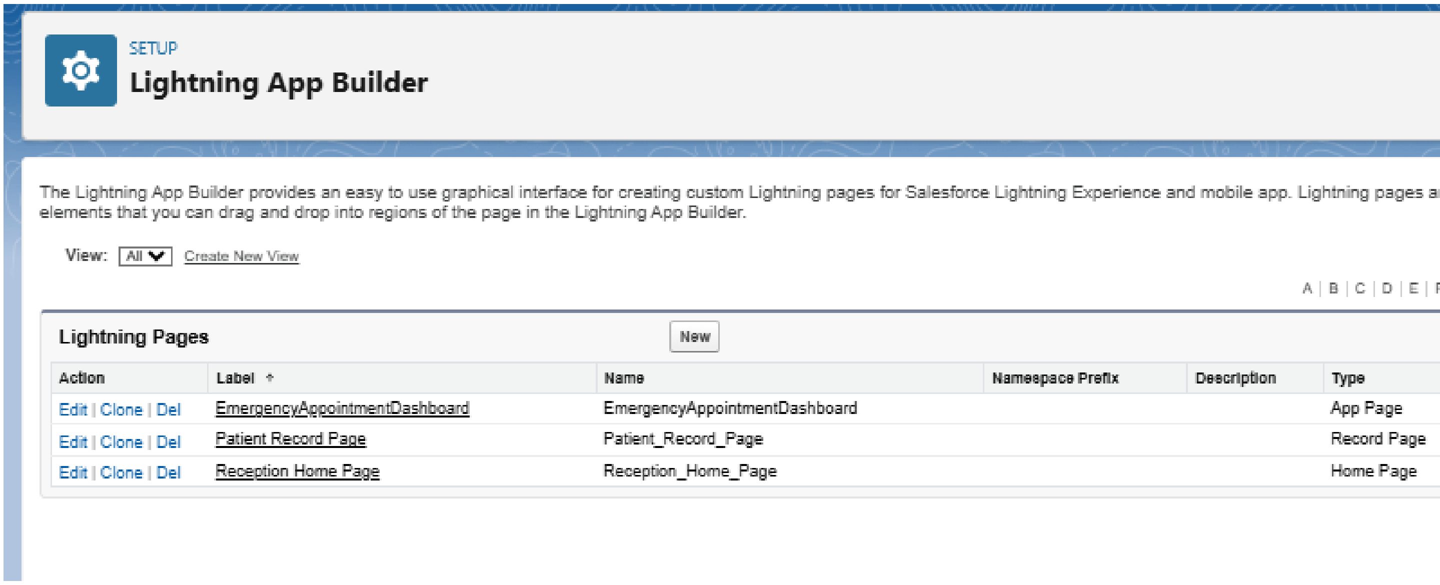
---

2. Lightning App Builder

Purpose: Create a custom Lightning app for healthcare staff.

Steps:

- Navigate to **Setup → App Manager → New Lightning App**
- App Name: *HealthCare360*
- Choose **Console Navigation** for multitasking
- Add **Utility Bar** items like “Quick Lookup” and “Insurance Status”
- Assign app to relevant profiles (Receptionist, Nurse, Admin)



SETUP

Lightning App Builder

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 a
elements that you can drag and drop into regions of the page in the Lightning App Builder.

View: All

Lightning Pages

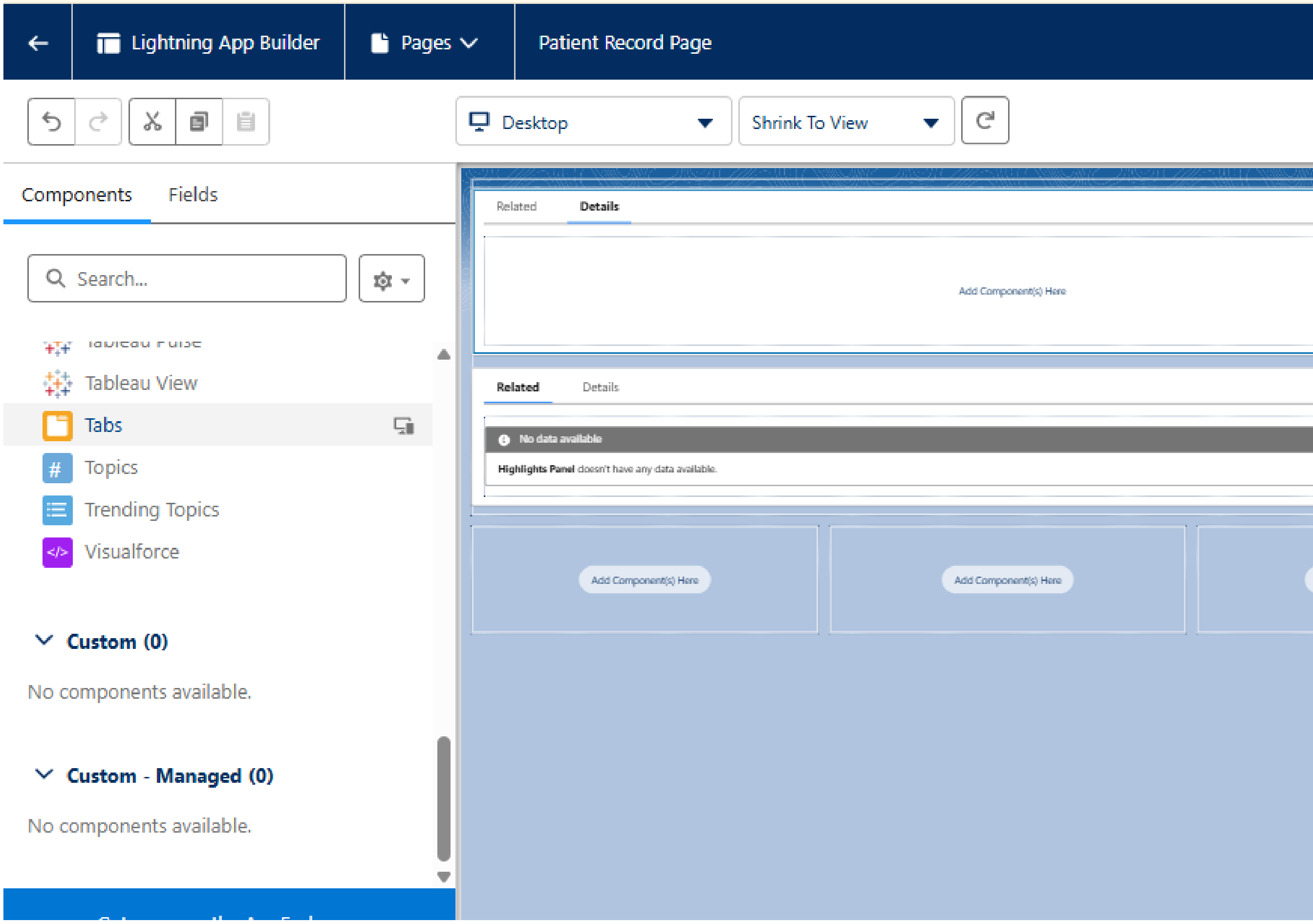
Action	Label	Name	Namespace Prefix	Description	Type
Edit | Clone | Del	EmergencyAppointmentDashboard	EmergencyAppointmentDashboard			App Page
Edit | Clone | Del	Patient Record Page	Patient_Record_Page			Record Page
Edit | Clone | Del	Reception Home Page	Reception_Home_Page			Home Page

3. Record Pages

Purpose: Customize record layouts for `Patient__c` and `Appointment__c` objects.

Steps:

- Go to **Setup → Lightning App Builder → New Record Page**
- Select object: *Patient* or *Appointment*
- Choose layout: Header + Two Columns
- Add components:
  - Highlights Panel (Name, Age, Status)
  - Tabs: Medical History, Insurance, Appointments
  - Flow Component: “Lock Appointment”
  - Custom Component: “Patient Timeline”

- 

No components available.

No components available.

No data available

Highlights Panel doesn't have any data available.

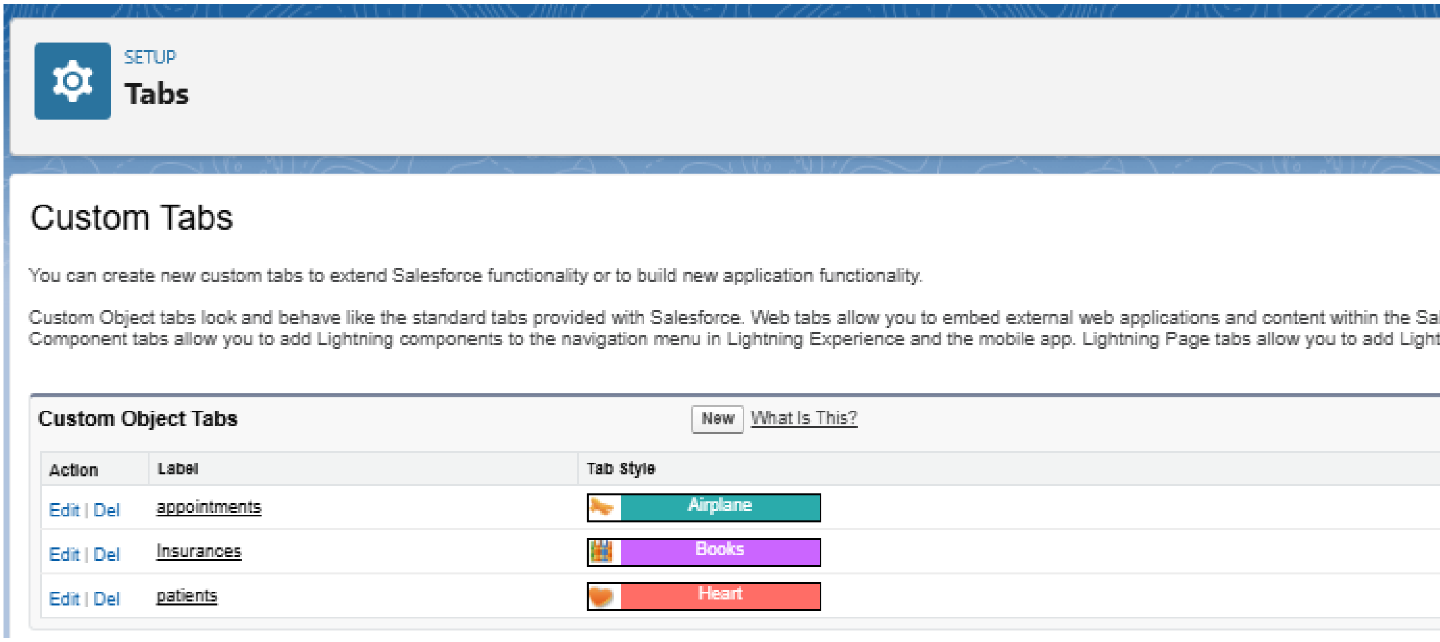
---

4. Tabs

Purpose: Organize navigation for healthcare roles.

Steps:

- Go to **App Manager → Edit App → Navigation Items**
- Add: *Patients*, *Appointments*, *Insurance*, *Reports*, *Dashboards*, *Flows*
- Reorder items based on role priority (e.g., Receptionist sees Appointments first)



Tabs

Custom Tabs

You can create new custom tabs to extend Salesforce functionality or to build new application functionality.

Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Sa
Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app. Lightning Page tabs allow you to add Light

Custom Object Tabs

Action	Label	Tab Style
Edit | Del	appointments	Airplane
Edit | Del	Insurances	Books
Edit | Del	patients	Heart

---

5. Home Page Layouts

Purpose: Provide a dashboard-style homepage for quick access.

Steps:

- Go to **Lightning App Builder → New Home Page**
- Add:
  - Dashboard: Daily Appointments
  - List View: Upcoming Checkups
  - Flow: “New Patient Intake”
- Assign homepage to relevant profiles

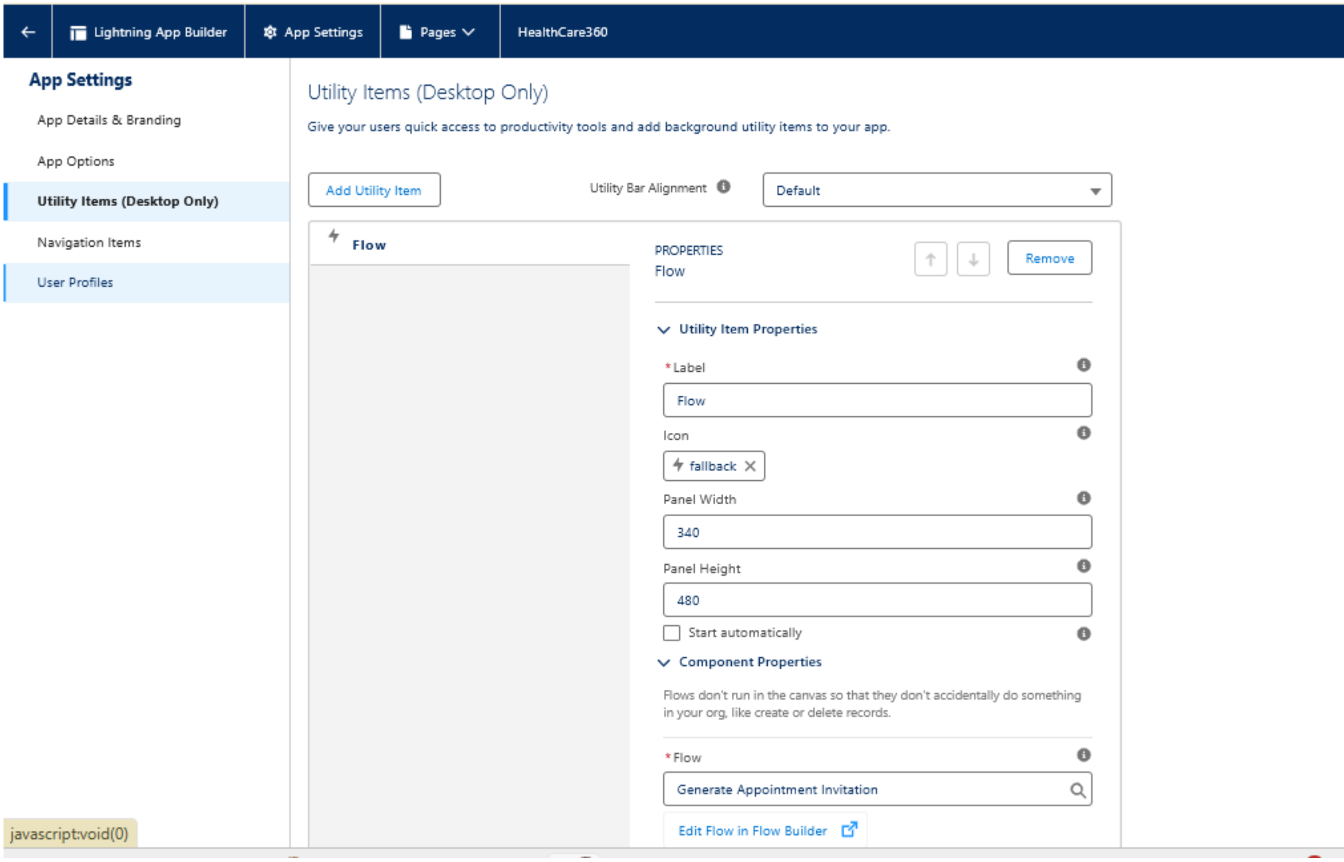
---

6. Utility Bar

Purpose: Offer quick access to tools from any screen.

Steps:

- Go to **App Manager → Edit App → Utility Items (Desktop Only)**
- Add Utility Item: Flow
  - Label: *Generate Appointment Invitation*
  - Panel Width: 340, Height: 480
  - Start Automatically: Unchecked
  - Select Flow: *Generate Appointment Invitation*
- Add additional items as needed (e.g., Insurance Verification)



Utility Items (Desktop Only)

Give your users quick access to productivity tools and add background utility items to your app.

Utility Bar Alignment: Default

PROPERTIES
Flow

Utility Item Properties

*Label: Flow

Icon: fallback

Panel Width: 340

Panel Height: 480

□ Start automatically

Component Properties

Flows don't run in the canvas so that they don't accidentally do something in your org, like create or delete records.

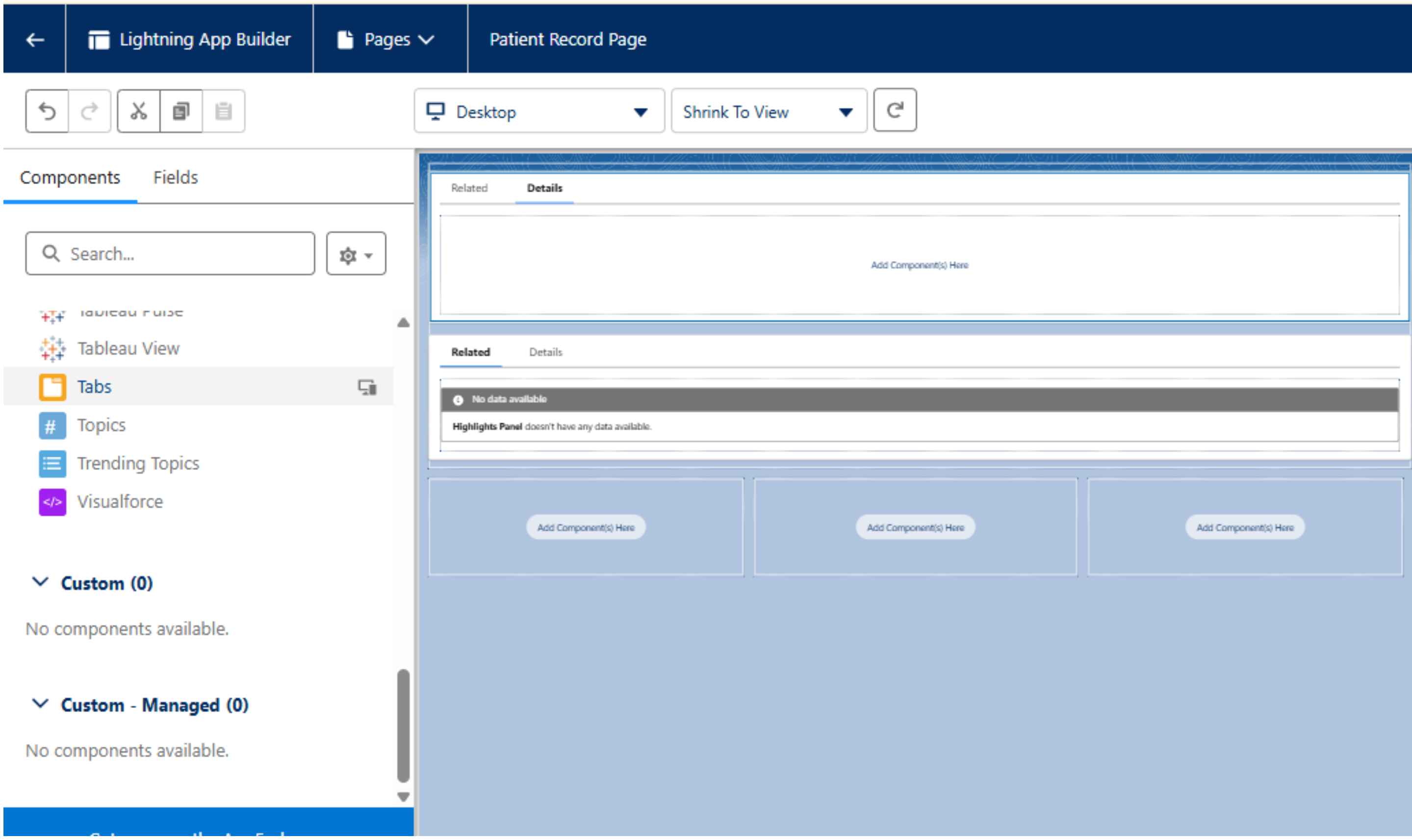
*Flow: Generate Appointment Invitation

7. Lightning Web Components (LWC)

Purpose: Use custom components for patient search, appointment locking, and insurance verification.

Steps:

- Create and deploy components using VS Code (optional)
- Ensure components are exposed to Lightning App Builder
- Add components to Record Pages or Utility Bar



No components available.

No components available.

No data available

Highlights Panel doesn't have any data available.

---

8. Apex with LWC

Purpose: Connect LWC components to backend logic for data operations.

Steps:

- Use Apex classes to fetch or update patient and appointment data
- Integrate Apex methods with LWC using wire adapters or imperative calls
- Ensure Apex classes are marked as `@AuraEnabled` and deployed

---

9. Events in LWC

Purpose: Enable communication between components.

Steps:

- Child component sends event (e.g., “Appointment Locked”)
- Parent component listens and responds (e.g., refreshes UI or shows message)
- Useful for modular workflows like appointment confirmation

---

10. Wire Adapters

Purpose: Automatically fetch data when component loads.

Steps:

- Connect LWC to Apex using reactive wire adapters
- Display data such as today’s appointments or patient list
- Use for dashboards or summary views

---

11. Imperative Apex Calls

Purpose: Trigger backend logic manually via button click.

Steps:

- User clicks “Verify Insurance”
- System sends request to Apex method
- Result (e.g., “Verified”) is shown in UI
- Ideal for real-time actions like insurance checks

---

12. Navigation Service

Purpose: Redirect users to specific records or pages.

Steps:

- Use navigation logic to open Patient or Appointment records
- Helpful after form submission or Flow completion
- Supports seamless transitions across components