Phase 6: User Interface Development

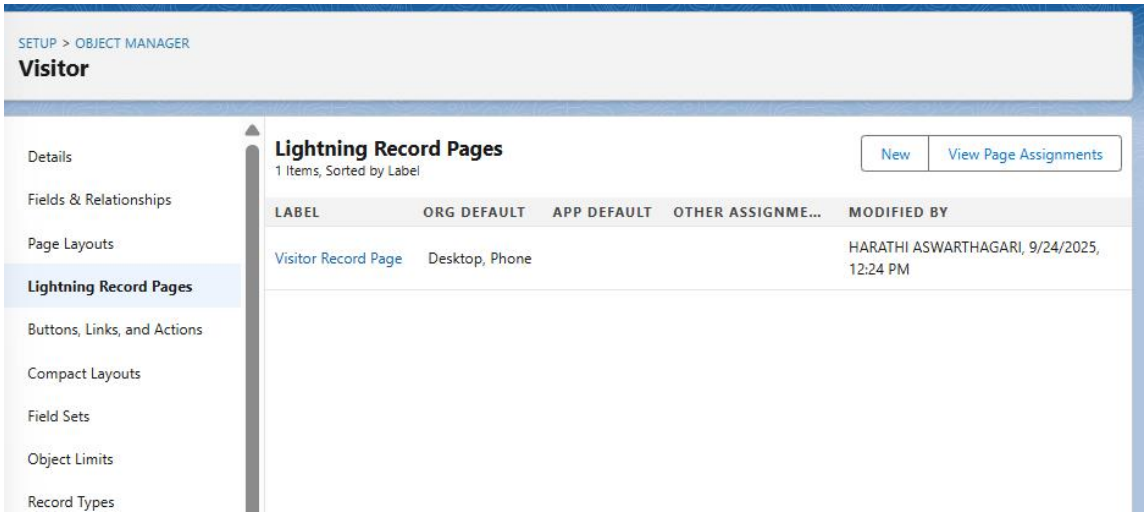
Project: Visitor checkIn system for offices

1. Lightning App Builder

Use Case:
The Lightning App Builder was used to design and customize the Visitor Management interface in Salesforce. It allowed us to create a record page layout for the Visitor object, incorporating the Visitor Check-In form and a dynamic list of checked-in visitors.

Implementation:

- Created a custom Lightning Record Page for the Visitor object.
- Added custom components (Aura/LWC) to display check-in forms and visitor lists.
- Configured tabs and layout sections for easy navigation.

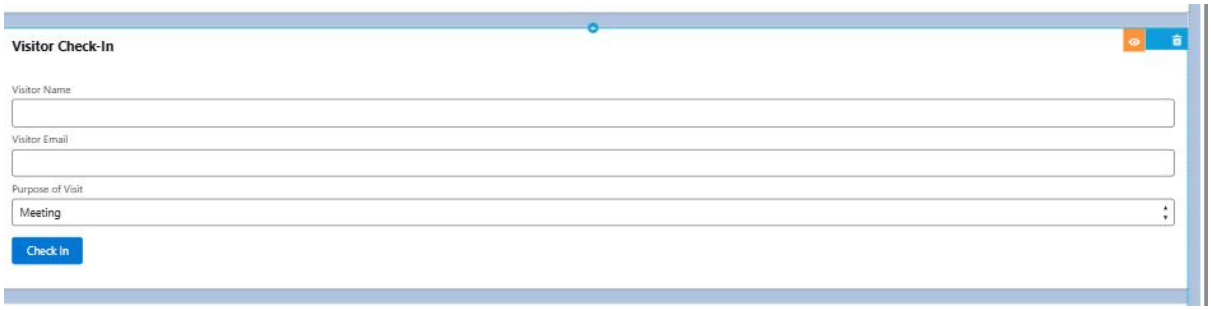


2. Record Pages

Use Case:
Customized the Visitor record page to show all essential fields (Name, Email, Status, Purpose of Visit, Check-In Time) and integrate the check-in workflow directly on the page.

Implementation:

- Added Highlights Panel to display key fields: Name, Status, Purpose.
- Configured Tabs Component:
    - Details Tab: All Visitor fields.
    - Related Tab: Related objects such as Host Employee.
    - Custom Tab: Added Visitor Check-In form component.



3. Tabs

Use Case:
Tabs were used to organize the Visitor record page for better usability. This allowed users to switch between Visitor details, related objects, and check-in actions without navigating away.

Implementation:

- Details Tab → Displayed all standard and custom fields.
- Related Tab → Showed related records such as Host Employee.
- Custom Tab → Displayed Visitor Check-In component (Aura).

4. Lightning Web Components (LWC) / Aura Components

Use Case:
LWC and Aura components were developed to capture visitor information dynamically and display checked-in visitors in a table.

Implementation:

- **Visitor Check-In Form (Aura):**
  - Input fields for Name, Email, and Purpose of Visit.
  - Picklist for predefined purposes: Meeting, Interview, Delivery, Other.
  - Check-In button triggers Apex method to save the record.
- **Visitor List (Aura):**
  - Displays all checked-in visitors dynamically using lightning:datatable.
  - Fetches data from Apex controller with @AuraEnabled method.

5. Apex with LWC / Aura

Use Case:
Apex classes were used to handle data insertion and retrieval for the Visitor Management system.

Implementation:

- **VisitorController Apex Class:**
  - insertVisitor → Inserts a new Visitor record.

- getCheckedInVisitors → Fetches all visitors with Status = Checked In.

- Integrated these methods with Aura components to make **imperative calls**.

Code:visitorCheckInForm.cmp

```
<aura:component controller="VisitorController"
implements="flexipage:availableForRecordHome,force:hasRecordId" access="global">
  <aura:attribute name="visitorName" type="String"/>
  <aura:attribute name="visitorEmail" type="String"/>
  <aura:attribute name="purpose" type="String"/>

  <aura:attribute name="purposeOptions" type="List" default="[
    {'label':'Meeting','value':'Meeting'},
    {'label':'Interview','value':'Interview'},
    {'label':'Delivery','value':'Delivery'},
    {'label':'Other','value':'Other'}
  ]"/>

  <lightning:card title="Visitor Check-In">
    <div class="slds-p-around_medium">
      <lightning:input label="Visitor Name" value="{!v.visitorName}"/>
      <lightning:input label="Visitor Email" value="{!v.visitorEmail}"/>
      <lightning:select label="Purpose of Visit" value="{!v.purpose}">
        <aura:iteration items="{!v.purposeOptions}" var="opt">
          <option value="{!opt.value}">{!opt.label}</option>
        </aura:iteration>
      </lightning:select>
      <lightning:button variant="brand" label="Check In" onclick="{!c.handleCheckIn}" class="slds-
m-top_small"/>
    </div>
  </lightning:card>
</aura:component>
```

visitorCheckInFormController.js

```
({
```

```
    handleCheckIn : function(component, event, helper) {
        var action = component.get("c.insertVisitor");
        action.setParams({
            name: component.get("v.visitorName"),
            email: component.get("v.visitorEmail"),
            purpose: component.get("v.purpose")
        });

        action.setCallback(this, function(response){
            var state = response.getState();
            if(state === "SUCCESS"){
                alert('Visitor checked in successfully!');
                component.set("v.visitorName", "");
                component.set("v.visitorEmail", "");
                component.set("v.purpose", "");
            } else {
                var errors = response.getError();
                alert('Error: ' + errors[0].message);
            }
        });
        $A.enqueueAction(action);
    }
})
```

Apex Controller (VisitorController)

```
public with sharing class VisitorController {
    @AuraEnabled
    public static void insertVisitor(String name, String email, String purpose) {
        Visitor__c v = new Visitor__c();
        v.Visitor_Name__c = name;
        v.Visitor_Email__c = email;
        v.Purpose_of_visit__c = purpose;
```

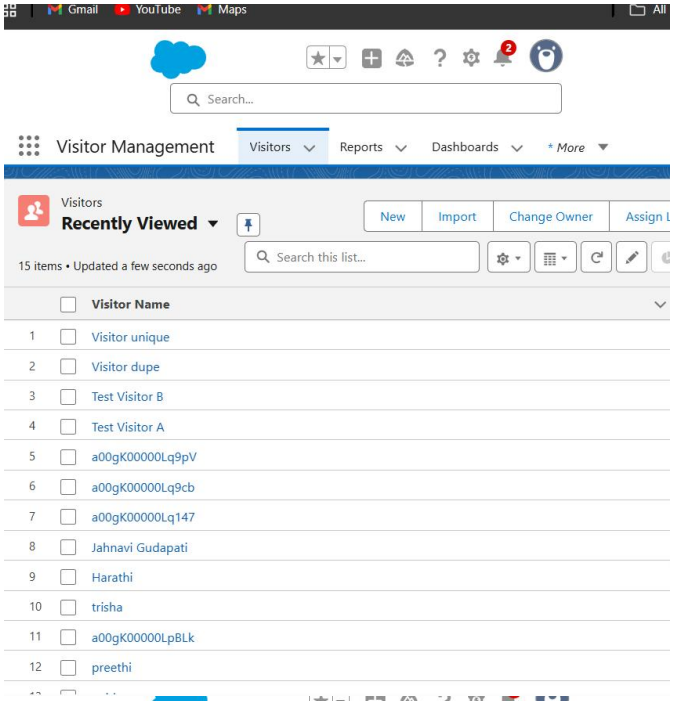
```
        v.Status__c = 'Checked In';
        v.Visitor_Check_In_Time__c = System.now();
        insert v;
    }
}
```

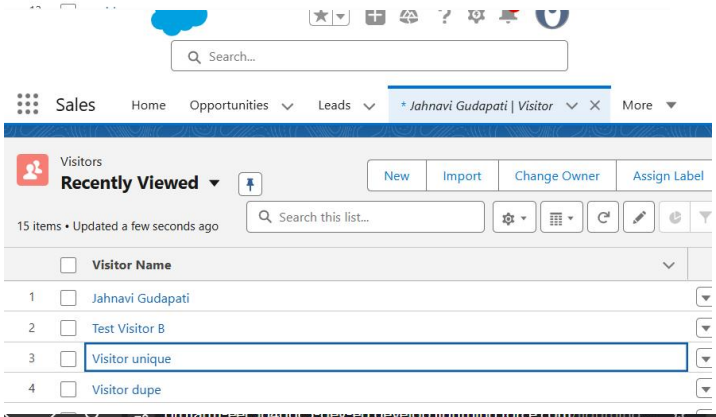
6. Events in LWC / Aura

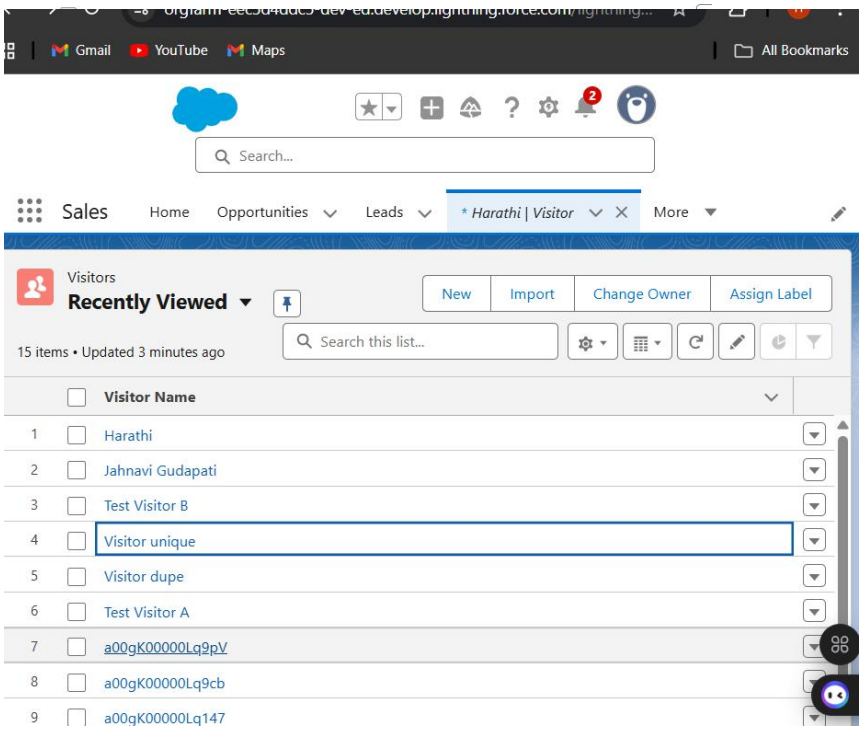
Use Case:
Aura event handling was implemented to update the visitor list dynamically after a new check-in.

Implementation:

- doInit method in Aura controller fetches visitor list on component load.
- After a visitor checks in, the list refreshes automatically without page reload.


Gmail
YouTube
Maps
Search...
Visitor Management
Visitors
Reports
Dashboards
* More
Visitors
Recently Viewed
New
Import
Change Owner
Assign L
15 items • Updated a few seconds ago
Search this list...
Visitor Name
1 Visitor unique
2 Visitor dupe
3 Test Visitor B
4 Test Visitor A
5 a00gK00000Lq9pV
6 a00gK00000Lq9cb
7 a00gK00000Lq147
8 Jahnavi Gudapati
9 Harathi
10 trisha
11 a00gK00000LpBLk
12 preethi


Search...
Sales
Home
Opportunities
Leads
* Jahnavi Gudapati | Visitor
More
Visitors
Recently Viewed
New
Import
Change Owner
Assign Label
15 items • Updated a few seconds ago
Search this list...
Visitor Name
1 Jahnavi Gudapati
2 Test Visitor B
3 Visitor unique
4 Visitor dupe


Gmail
YouTube
Maps
All Bookmarks
Search...
Sales
Home
Opportunities
Leads
* Harathi | Visitor
More
Visitors
Recently Viewed
New
Import
Change Owner
Assign Label
15 items • Updated 3 minutes ago
Search this list...
Visitor Name
1 Harathi
2 Jahnavi Gudapati
3 Test Visitor B
4 Visitor unique
5 Visitor dupe
6 Test Visitor A
7 a00gK00000Lq9pV
8 a00gK00000Lq9cb
9 a00gK00000Lq147

Based on checkIn visitors the list keep changing

7. Wire Adapters & Imperative Apex Calls

Use Case:

- **Wire Adapters:** Can be used in LWC to fetch Salesforce data reactively.
- **Imperative Apex Calls:** Used in Aura components to fetch visitor data after check-in.

Implementation:

- Aura component uses **imperative call** to getCheckedInVisitors Apex method.
- Updates visitorList attribute, which is rendered in lightning:datatable.

8. Navigation Service

Use Case:
Navigation Service can be used to **redirect users to the Visitor record page or other related pages** after check-in.

Implementation:

- After check-in, user is optionally navigated to the Visitor record detail page.