Phase 6: User Interface Development:

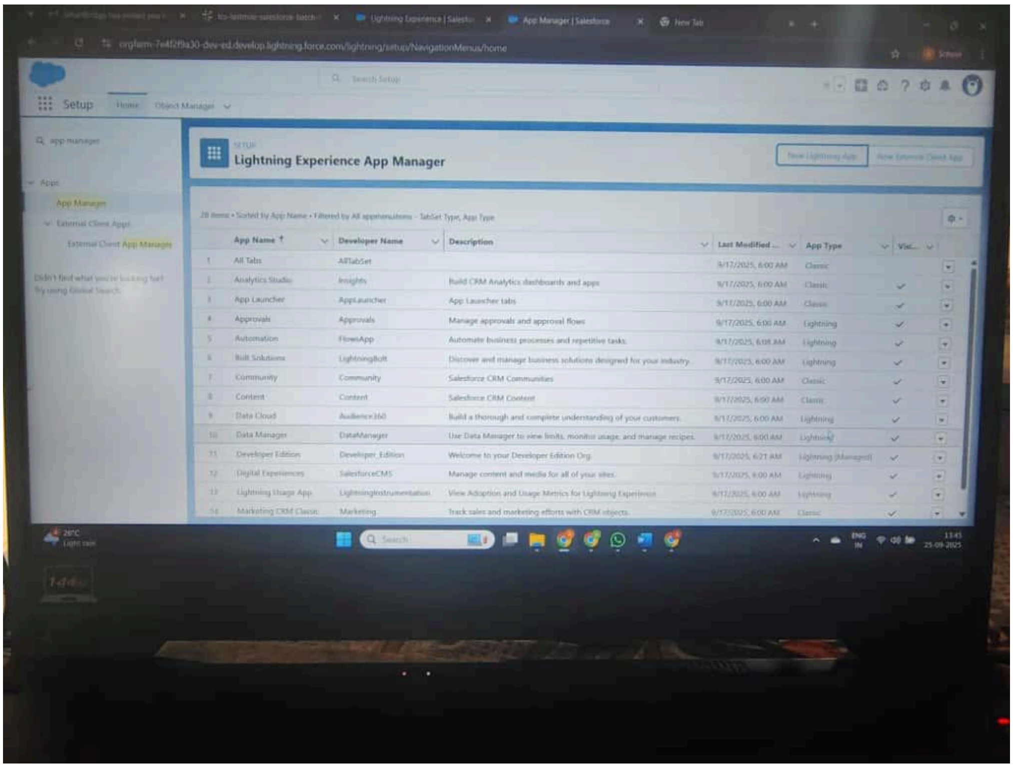
Lightning App Builder:

- Create custom applications using drag-and-drop.

- Steps: Setup → App Builder → New Lightning App → Add Tabs/Pages → Save & Activate
- LOGO:.

RESIDENT
CONNECT

Create a Lightning App (App Builder):



SETUP

Lightning Experience App Manager

28 items • Sorted by App Name • Filtered by All appmenuitems - TabSet Type, App Type

	App Name	Developer Name	Description	Last Modified ...	App Type	Visi...
1	All Tabs	AllTabSet		9/17/2025, 6:00 AM	Classic	
2	Analytics Studio	Insights	Build CRM Analytics dashboards and apps	9/17/2025, 6:00 AM	Classic	✓
3	App Launcher	AppLauncher	App Launcher tabs	9/17/2025, 6:00 AM	Classic	✓
4	Approvals	Approvals	Manage approvals and approval flows	9/17/2025, 6:00 AM	Lightning	✓
5	Automation	FlowsApp	Automate business processes and repetitive tasks.	9/17/2025, 6:08 AM	Lightning	✓
6	Bolt Solutions	LightningBolt	Discover and manage business solutions designed for your industry.	9/17/2025, 6:00 AM	Lightning	✓
7	Community	Community	Salesforce CRM Communities	9/17/2025, 6:00 AM	Classic	✓
8	Content	Content	Salesforce CRM Content	9/17/2025, 6:00 AM	Classic	✓
9	Data Cloud	Audience360	Build a thorough and complete understanding of your customers.	9/17/2025, 6:00 AM	Lightning	✓
10	Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	9/17/2025, 6:00 AM	Lightning	✓
11	Developer Edition	Developer_Edition	Welcome to your Developer Edition Org.	9/17/2025, 6:21 AM	Lightning (Managed)	✓
12	Digital Experiences	SalesforceCMS	Manage content and media for all of your sites.	9/17/2025, 6:00 AM	Lightning	✓
13	Lightning Usage App	LightningInstrumentation	View Adoption and Usage Metrics for Lightning Experience	9/17/2025, 6:00 AM	Lightning	✓
14	Marketing CRM Classic	Marketing	Track sales and marketing efforts with CRM objects.	9/17/2025, 6:00 AM	Classic	✓

Record Pages:

- Customize record detail and related lists.

- Example: Add a “Maintenance Requests” related list on the Resident record page.

Tabs & Home Page Layouts:

- Add Custom Tabs (e.g., *Resident Services*, *Community Events*).
- Modify Home Page: Setup → Lightning App Builder → Edit Home Page → Add components (e.g., “My Tasks”, “Community Announcements”).

Utility Bar:

- Quick access toolbar (e.g., “Raise Complaint”, “Event Calendar”).
- Steps: Setup → App Manager → Edit App → Add Utility Items.

Lightning Web Components (LWC):

- Reusable UI components written in JavaScript, HTML, and CSS.
- Example: `residentFeedback.js` + `residentFeedback.html` to capture resident satisfaction.
- import { LightningElement, track } from 'lwc';

```
import { LightningElement, track } from 'lwc';
export default class ResidentFeedback extends LightningElement {
  @track feedback = '';
  handleChange(event) {
    this.feedback = event.target.value;
  }
  handleSubmit() {
    // Save feedback in Salesforce
    }
}
```

Apex with LWC:

- Fetch data from Salesforce backend.

```
import { LightningElement, wire } from 'lwc';
import getResidents from
'@salesforce/apex/ResidentController.getResidents';

export default class ResidentList extends LightningElement {
    @wire(getResidents) residents;
}
 public with sharing class ResidentController {
    @AuraEnabled(cacheable=true)
    public static List<Contact> getResidents() {
        return [SELECT Id, Name, Email FROM Contact LIMIT 20];
    }
}
```

Events in LWC

- Parent-child communication using events (`CustomEvent`).
- Example: Child “FeedbackForm” fires an event → Parent “ResidentPage” updates UI.

Wire Adapters & Imperative Apex Calls

- **Wire:** Automatically fetch data with reactive updates.
- **Imperative:** Call Apex explicitly on button click.

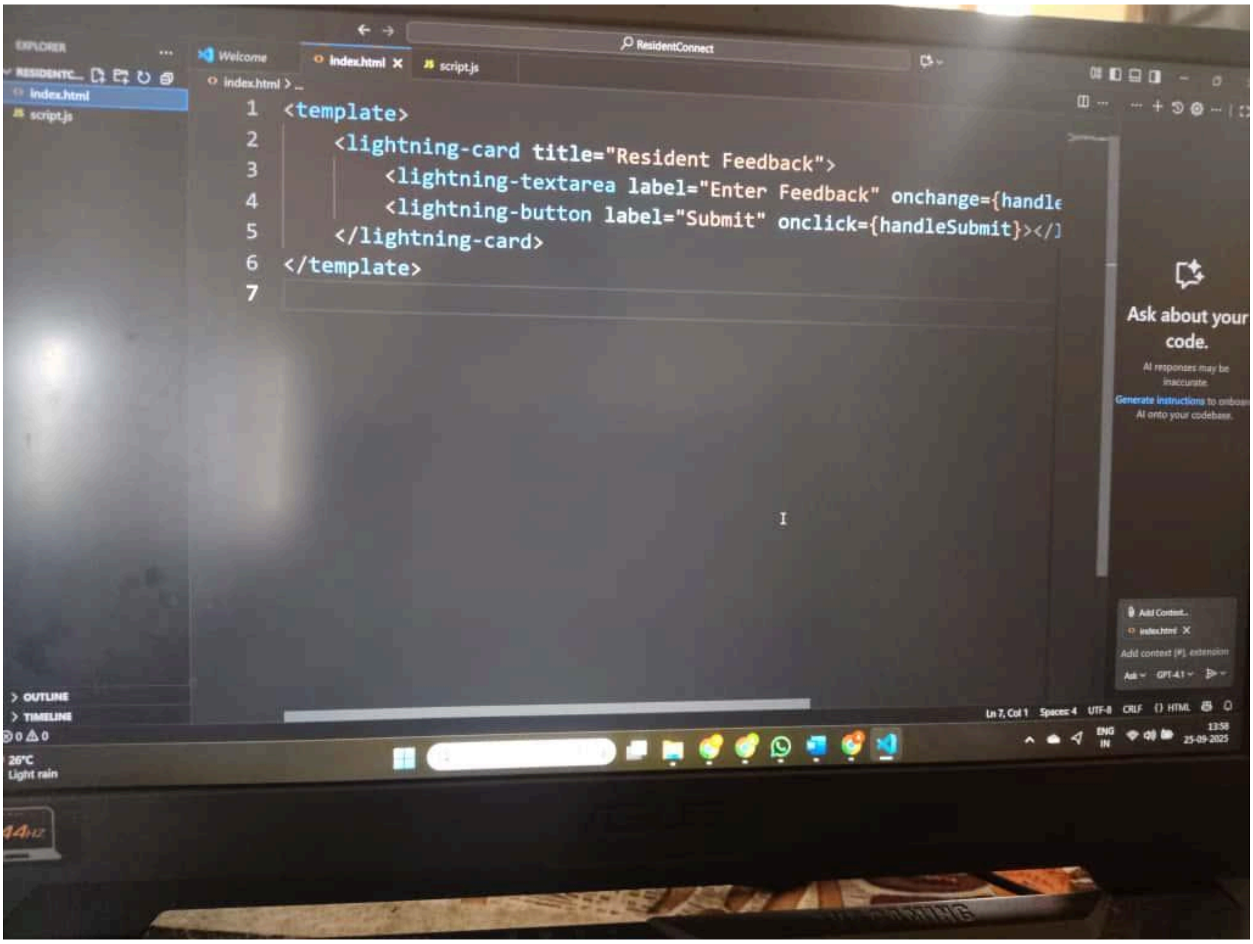
```
import { LightningElement } from 'lwc';
import sendNotification from
'@salesforce/apex/NotificationController.sendNotification';

export default class NotifyResidents extends LightningElement {
    handleNotify() {
        sendNotification({ message: 'Community Event Tomorrow!' });
    }
}
```

Navigation Service

- Navigate between records, pages, or external links.

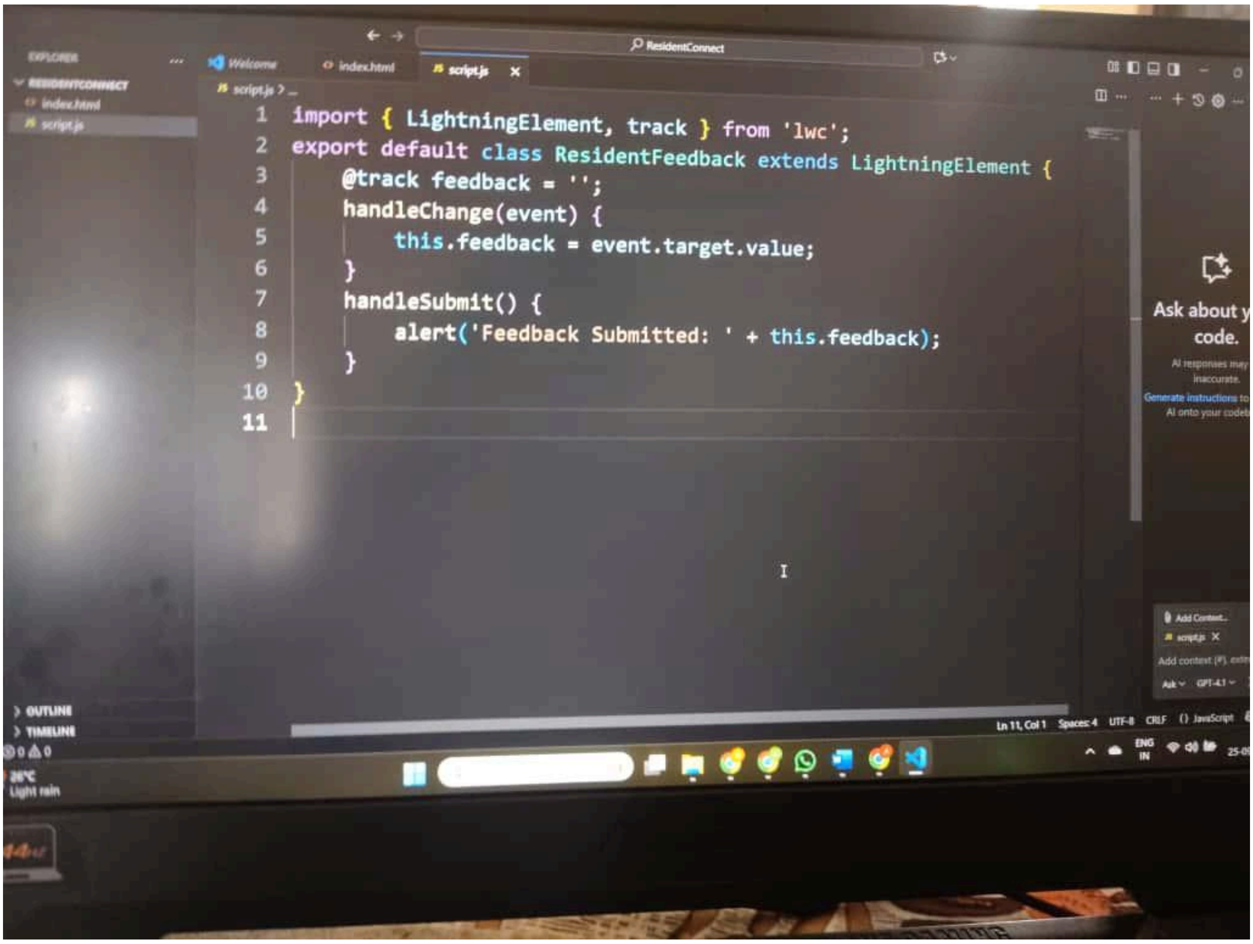
```
import { NavigationMixin } from 'lightning/navigation';
export default class ResidentNavigator extends
NavigationMixin(LightningElement) {
    navigateToRecordPage(recordId) {
        this[NavigationMixin.Navigate]({
            type: 'standard__recordPage',
            attributes: {
                recordId: recordId,
                objectApiName: 'Contact',
                actionName: 'view'
            }
        });
    }
}
```

residentFeedback.html:



```
<template>
    <lightning-card title="Resident Feedback">
        <lightning-textarea label="Enter Feedback" onchange={handle
        <lightning-button label="Submit" onclick={handleSubmit}></]
    </lightning-card>
</template>
```

residentFeedback.js:



```
import { LightningElement, track } from 'lwc';
export default class ResidentFeedback extends LightningElement {
    @track feedback = '';
    handleChange(event) {
        this.feedback = event.target.value;
    }
    handleSubmit() {
        alert('Feedback Submitted: ' + this.feedback);
    }
}

```