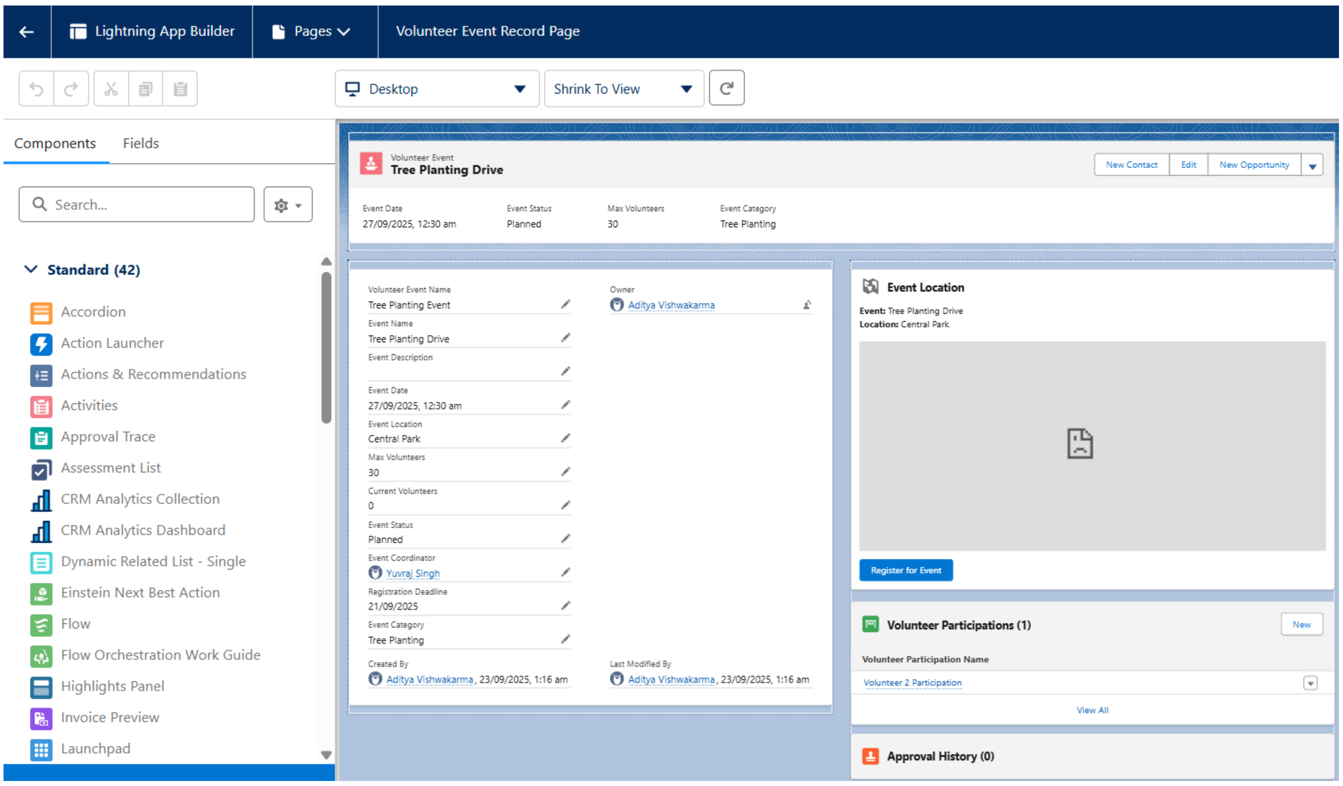
Phase 6 User Interface Development

Lightning App Builder & Record Pages

- ➢ Custom Volunteer Event Record Page
- ➢ In Setup → Lightning App Builder, click New → Record Page
  - ✓ Label: Volunteer Event Record Page
  - ✓ Object: Volunteer_Event__c
- ➢ Drag standard Highlights Panel and Related Lists onto the canvas.
- ➢ Add your custom Lightning Web Component (see Section 3) into a full-width region.
- ➢ Activate for the Org Default and assign to desktop and mobile.



Volunteer Event
Tree Planting Drive

Event Date: 27/09/2025, 12:30 am
Event Status: Planned
Max Volunteers: 30
Event Category: Tree Planting

Volunteer Event Name: Tree Planting Event
Owner: Aditya Vishwakarma
Event Name: Tree Planting Drive
Event Description:
Event Date: 27/09/2025, 12:30 am
Event Location: Central Park
Max Volunteers: 30
Current Volunteers: 0
Event Status: Planned
Event Coordinator: Yuvraj Singh
Registration Deadline: 21/09/2025
Event Category: Tree Planting
Created By: Aditya Vishwakarma, 23/09/2025, 1:16 am
Last Modified By: Aditya Vishwakarma, 23/09/2025, 1:16 am

Event Location
Event: Tree Planting Drive
Location: Central Park

Volunteer Participations (1)

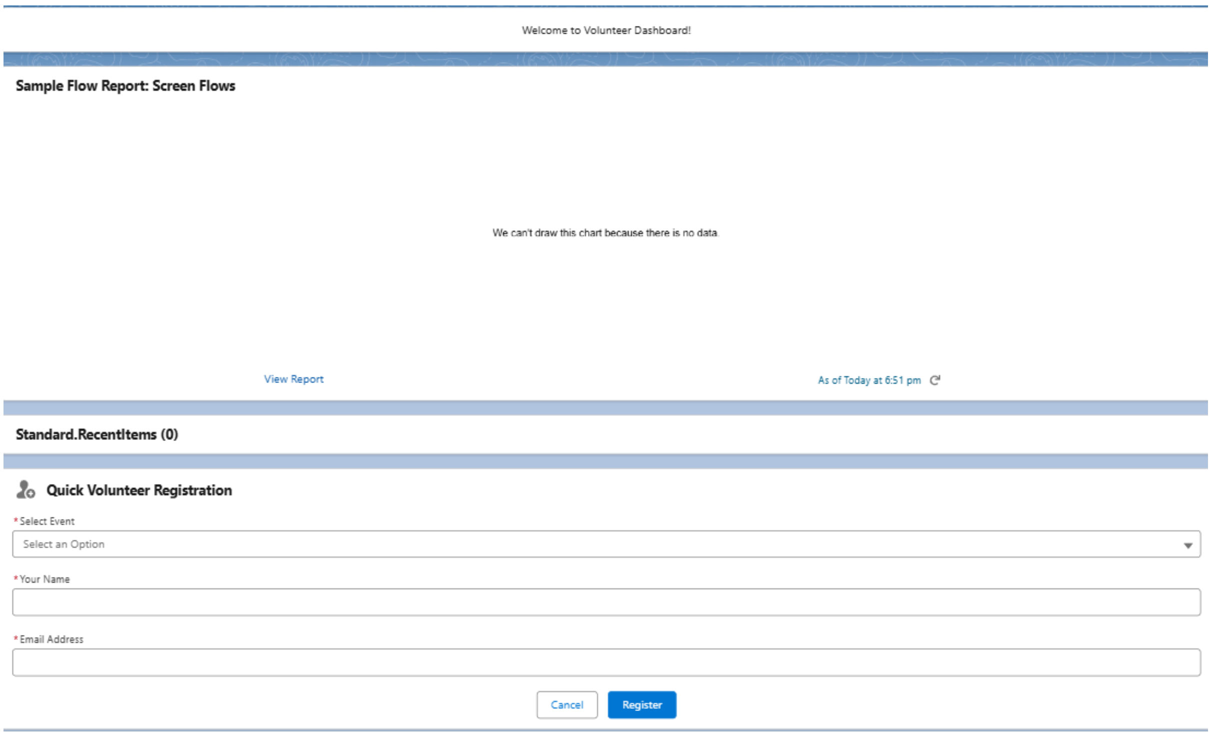
Volunteer Participation Name
Volunteer 2 Participation

Approval History (0)

Tabs & Utility Bar

- ➢ Volunteer Dashboard Tab
- ➢ In Setup → Tabs, create a Lightning Page Tab
  - ✓ Label: Volunteer Dashboard

    - ✓ Lightning Page: a new app page built below
- ➢ Utility Bar Component
- ➢ In Setup → App Manager, edit the Volunteer App
- ➢ Under Utility Bar, click Add, choose your LWC (e.g., Registration Modal)
- ➢ Set Label: Quick Register, Icon: user_add


Welcome to Volunteer Dashboard!

Sample Flow Report: Screen Flows

We can't draw this chart because there is no data.

Standard.RecentItems (0)

Quick Volunteer Registration

*Select Event
Select an Option

*Your Name

*Email Address

»» Lightning Web Components (LWC)

Component: volunteerEventMap

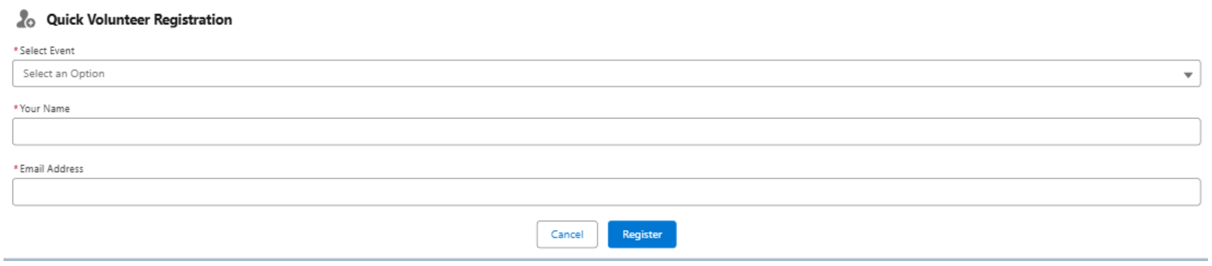
- ➢ Purpose: Display event location on Google Map
- ➢ Files:
    - ✓ volunteerEventMap.html
    - ✓ volunteerEventMap.js
    - ✓ volunteerEventMap.js-meta.xml

Component: volunteerRegistrationModal

- ➢ Purpose: Let users register for an event in a modal
- ➢ Files:
    - ✓ volunteerRegistrationModal.html

✓ volunteerRegistrationModal.js

✓ volunteerRegistrationModal.js-meta.xml

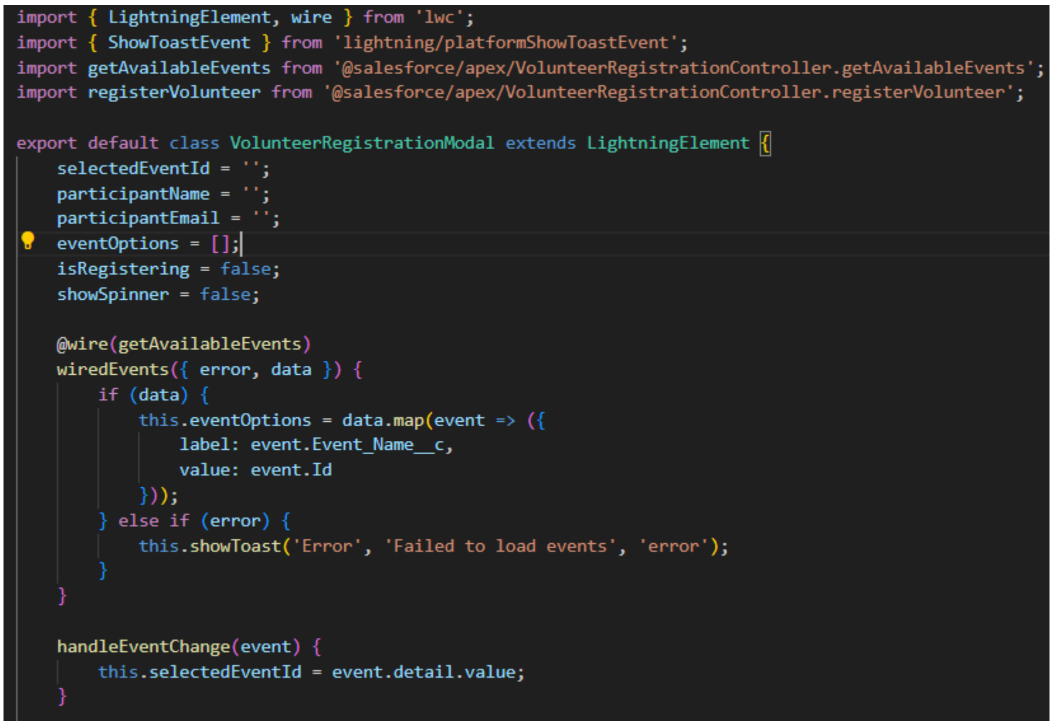


⋙ Apex with LWC & Imperative Apex Calls

Apex Class: VolunteerRegistrationController

```
public with sharing class VolunteerRegistrationController {
  @AuraEnabled
  public static Id registerVolunteer(Id eventId, String participantEmail) {
    Volunteer_Participation__c vp = new Volunteer_Participation__c(
      Event__c = eventId,
      Participant_Email__c = participantEmail,
      Participation_Status__c = 'Registered'
    );
    insert vp;
    return vp.Id;
  }
}
```

➢ Exposed as an imperative call from volunteerRegistrationModal.js.



```
import { LightningElement, wire } from 'lwc';
import { ShowToastEvent } from 'lightning/platformShowToastEvent';
import getAvailableEvents from '@salesforce/apex/VolunteerRegistrationController.getAvailableEvents';
import registerVolunteer from '@salesforce/apex/VolunteerRegistrationController.registerVolunteer';

export default class VolunteerRegistrationModal extends LightningElement {
    selectedEventId = '';
    participantName = '';
    participantEmail = '';
    eventOptions = [];
    isRegistering = false;
    showSpinner = false;

    @wire(getAvailableEvents)
    wiredEvents({ error, data }) {
        if (data) {
            this.eventOptions = data.map(event => ({
                label: event.Event_Name__c,
                value: event.Id
            }));
        } else if (error) {
            this.showToast('Error', 'Failed to load events', 'error');
        }
    }

    handleEventChange(event) {
        this.selectedEventId = event.detail.value;
    }
```

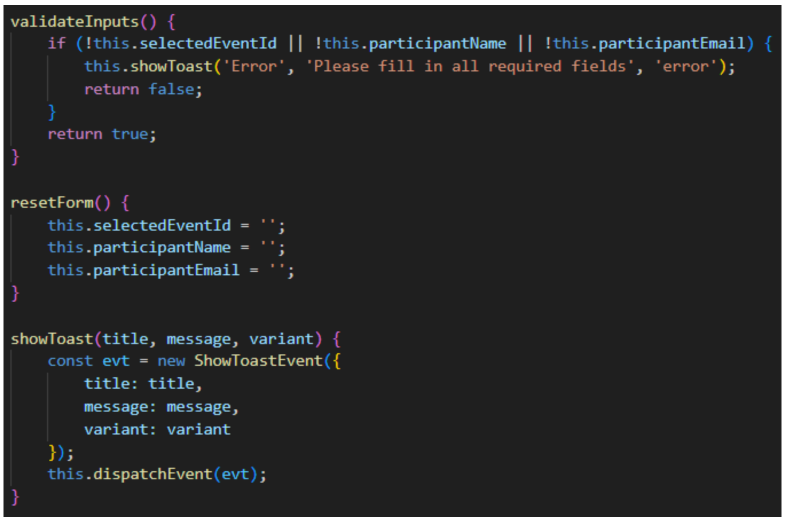
Events in LWC

Publish an Event After Registration

- Use a custom DOM event in volunteerRegistrationModal.js:

```
const registrationEvent = new CustomEvent('registered', {
  detail: { eventId: this.recordId }
});
this.dispatchEvent(registrationEvent);
```

- Consume in volunteerEventMap.js to refresh the map or count badge.

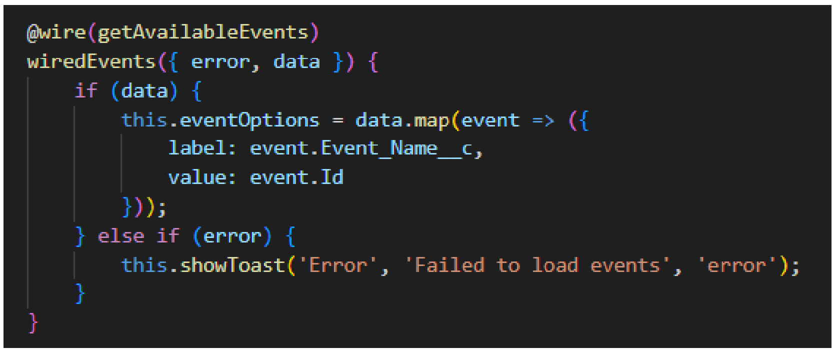


```
validateInputs() {
    if (!this.selectedEventId || !this.participantName || !this.participantEmail) {
        this.showToast('Error', 'Please fill in all required fields', 'error');
        return false;
    }
    return true;
}

resetForm() {
    this.selectedEventId = '';
    this.participantName = '';
    this.participantEmail = '';
}

showToast(title, message, variant) {
    const evt = new ShowToastEvent({
        title: title,
        message: message,
        variant: variant
    });
    this.dispatchEvent(evt);
}
```

Wire Adapters

Fetch Event Record Data

- In volunteerEventMap.js, use @wire(getRecord, { recordId: '$recordId', fields: [EVENT_NAME_FIELD, LOCATION_FIELD] }) record;
- Display record.data.fields.Event_Location__c.value on the map.



```
@wire(getAvailableEvents)
wiredEvents({ error, data }) {
    if (data) {
        this.eventOptions = data.map(event => ({
            label: event.Event_Name__c,
            value: event.Id
        }));
    } else if (error) {
        this.showToast('Error', 'Failed to load events', 'error');
    }
}
```

Navigation Service

Navigate to Registration Modal via Button

- In volunteerEventMap.js, import NavigationMixin and call:

```
this[NavigationMixin.Navigate]({

  type: 'standard__navItemPage',

  attributes: { apiName: 'Volunteer_Dashboard' }

});
```