Phase 6

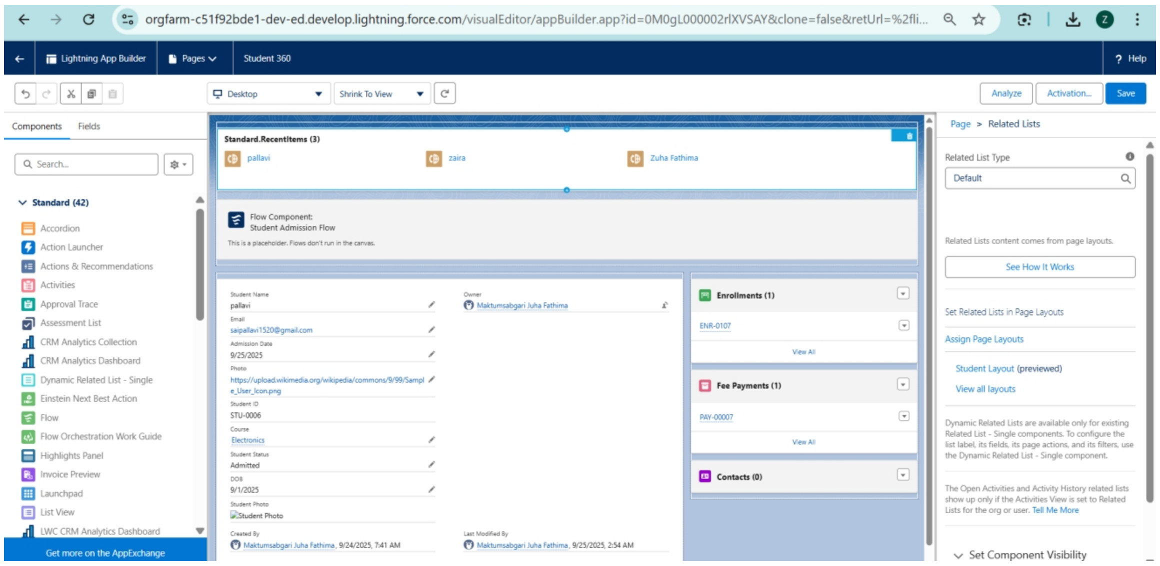
User Interface Development

Goal

Provide a user-friendly interface for managing students, enrollments, and fee payments in Salesforce.

Lightning Record Page for Student (Student 360 Page)

- Navigate to Setup → Object Manager → Student → Lightning Record Pages.
- Create a new App Page or Record Page.
- Add components:
  • Highlights Panel → Student's Name, Email, Status.
  • Related Lists: Enrollments, Fee Payments.
  • Tabs → Info | Related | Fee Calculator (if LWC is added).
- Deliverable → Screenshot of Student 360 Page showing student details + related lists.


orgfarm-c51f92bde1-dev-ed.develop.lightning.force.com/visualEditor/appBuilder.app?id=0M0gL000002rlXVSAY&clone=false&retUrl=%2fli...

Standard.RecentItems (3)
pallavi | zaira | Zuha Fathima

Flow Component:
Student Admission Flow
This is a placeholder. Flows don't run in the canvas.

Student Name: pallavi
Email: saipallavi1520@gmail.com
Admission Date: 9/25/2025
Photo: https://upload.wikimedia.org/wikipedia/commons/9/99/Sample_User_Icon.png
Student ID: STU-0006
Course: Electronics
Student Status: Admitted
DOB: 9/1/2025
Student Photo: Student Photo
Created By: Maktumsabgari Juha Fathima, 9/24/2025, 7:41 AM
Owner: Maktumsabgari Juha Fathima
Last Modified By: Maktumsabgari Juha Fathima, 9/25/2025, 2:54 AM

Enrollments (1)
ENR-0107

Fee Payments (1)
PAY-00007

Contacts (0)

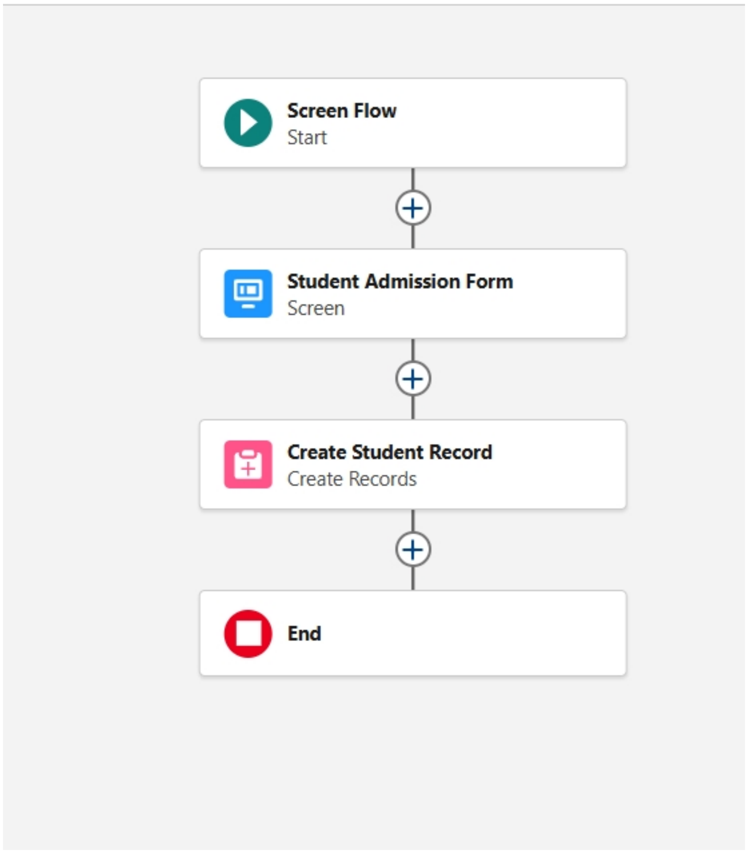
Page > Related Lists
Related List Type: Default
Related Lists content comes from page layouts.
Set Related Lists in Page Layouts
Assign Page Layouts
Student Layout (previewed)
View all layouts
Dynamic Related Lists are available only for existing Related List - Single components. To configure the list label, its fields, its page actions, and its filters, use the Dynamic Related List - Single component.
The Open Activities and Activity History related lists show up only if the Activities View is set to Related Lists for the org or user. Tell Me More

Screen Flow – Student Admission Form

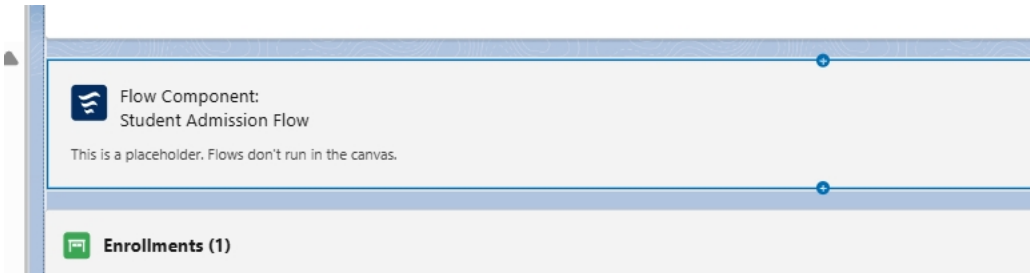
Purpose: Let admins create a new student easily via a guided form.

Steps:
1. Go to Setup → Flow → New Flow → Screen Flow.

2. Screen Element: Add fields for student details (Name, Email, Address, Course selection).
3. Create Records Element:
   - Object = Student
   - Map input fields to Student fields.
4. Connect elements → Save → Activate the Flow.



5. Add Flow to Student App Page using Flow Component.


Flow Component:
Student Admission Flow

This is a placeholder. Flows don't run in the canvas.

☞ Result: Admin can admit students using a simple UI form.

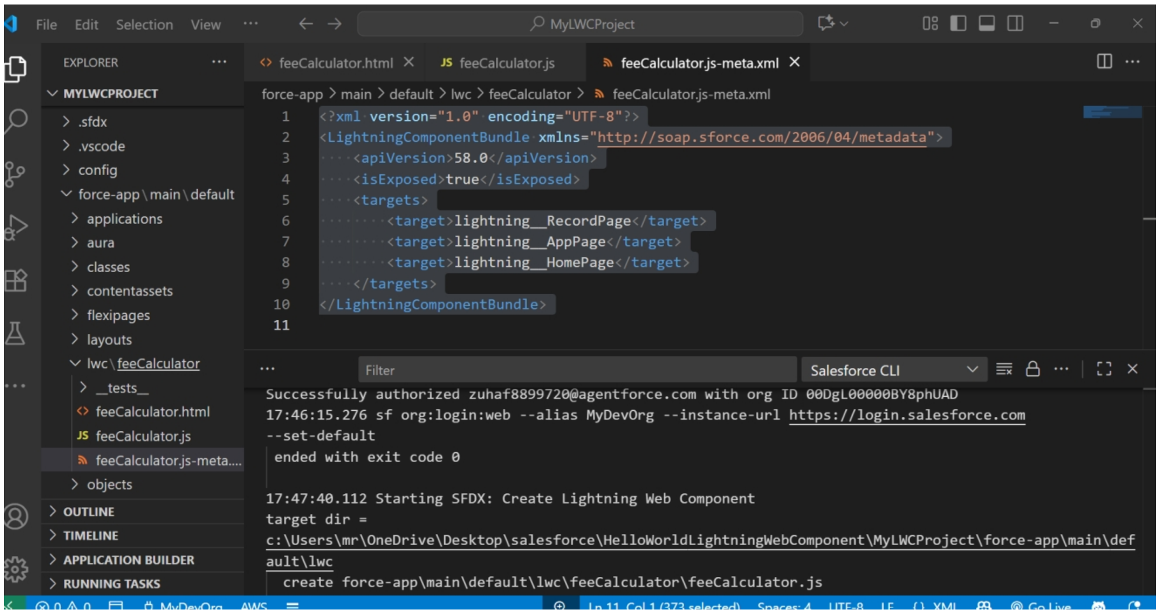
Lightning Web Component – Fee Calculator

Purpose: Allow quick calculation of fees (e.g., Course Fee – Paid Amount = Outstanding Fee).

LWC Development Steps:
1. Create Project (in VS Code Command Palette):
   SFDX: Create Project with Manifest
2. Create LWC:
   SFDX: Create Lightning Web Component
   Name → feeCalculator
3. Edit Files:
   - feeCalculator.html → Create form (inputs for Course Fee & Paid Fee).

Code:

```
<template>
    <lightning-card title="Fee Calculator">
        <div class="slds-m-around_medium">
            <lightning-input
                label="Course Fee"
                type="number"
                value={courseFee}
                onchange={handleCourseFeeChange}>
            </lightning-input>

            <lightning-input
                label="Paid Amount"
                type="number"
                value={paidAmount}
                onchange={handlePaidAmountChange}>
            </lightning-input>

            <lightning-button
                label="Calculate"
                onclick={calculateOutstanding}
                class="slds-m-top_small">
            </lightning-button>

            <template if:true={outstandingFee}>
                <p class="slds-m-top_medium">
                    <b>Outstanding Fee:</b> {outstandingFee}
                </p>
            </template>
        </div>
    </lightning-card>
</template>
```



```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordPage</target>
        <target>lightning__AppPage</target>
        <target>lightning__HomePage</target>
    </targets>
</LightningComponentBundle>
```

```
Successfully authorized zuhaf8899720@agentforce.com with org ID 00DgL00000BY8phUAD
17:46:15.276 sf org:login:web --alias MyDevOrg --instance-url https://login.salesforce.com
--set-default
 ended with exit code 0

17:47:40.112 Starting SFDX: Create Lightning Web Component
target dir =
c:\Users\mr\OneDrive\Desktop\salesforce\HelloWorldLightningWebComponent\MyLWCProject\force-app\main\def
ault\lwc
  create force-app\main\default\lwc\feeCalculator\feeCalculator.js
```

- feeCalculator.js → Logic to calculate outstanding fee.

Code:

```
import { LightningElement } from 'lwc';

export default class FeeCalculator extends LightningElement {
    courseFee = 0;
    paidAmount = 0;
    outstandingFee;

    handleCourseFeeChange(event) {
        this.courseFee = parseFloat(event.target.value) || 0;
    }

    handlePaidAmountChange(event) {
        this.paidAmount = parseFloat(event.target.value) || 0;
    }

    calculateOutstanding() {
        this.outstandingFee = this.courseFee - this.paidAmount;
    }
}
```

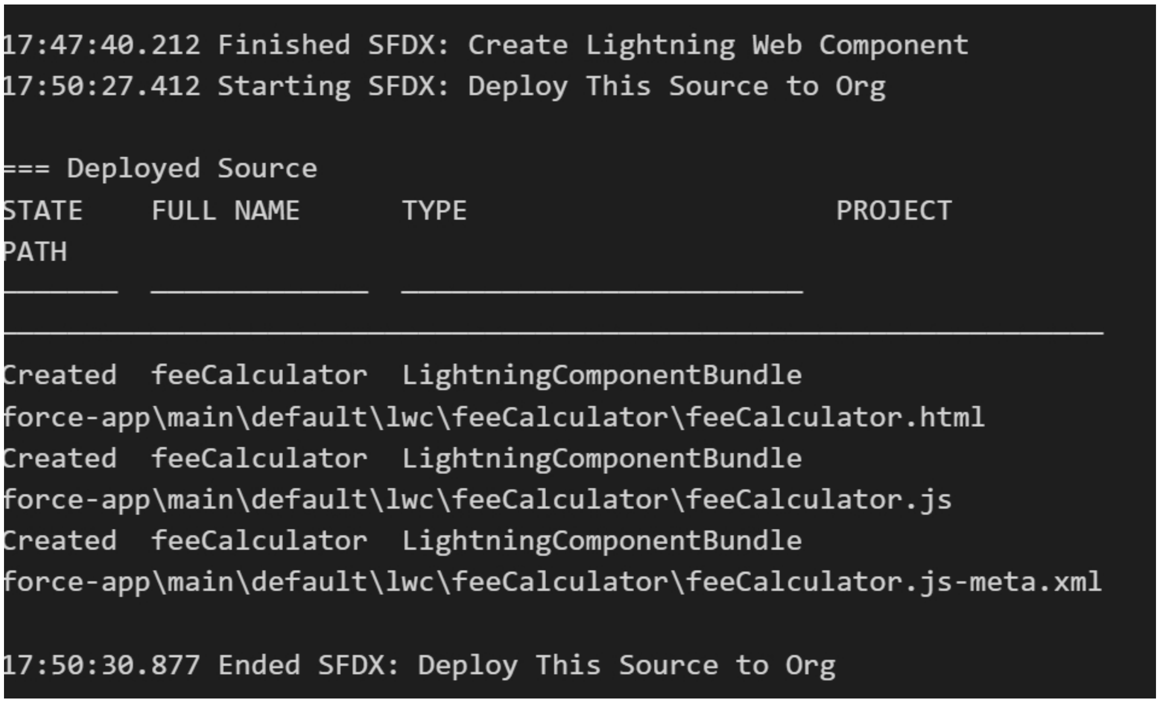
- feeCalculator.js-meta.xml → Expose component to Record Page.

Code:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
   <apiVersion>58.0</apiVersion>
   <isExposed>true</isExposed>
   <targets>
      <target>lightning__RecordPage</target>
      <target>lightning__AppPage</target>
      <target>lightning__HomePage</target>
   </targets>
</LightningComponentBundle>
```

4. Deploy Component:

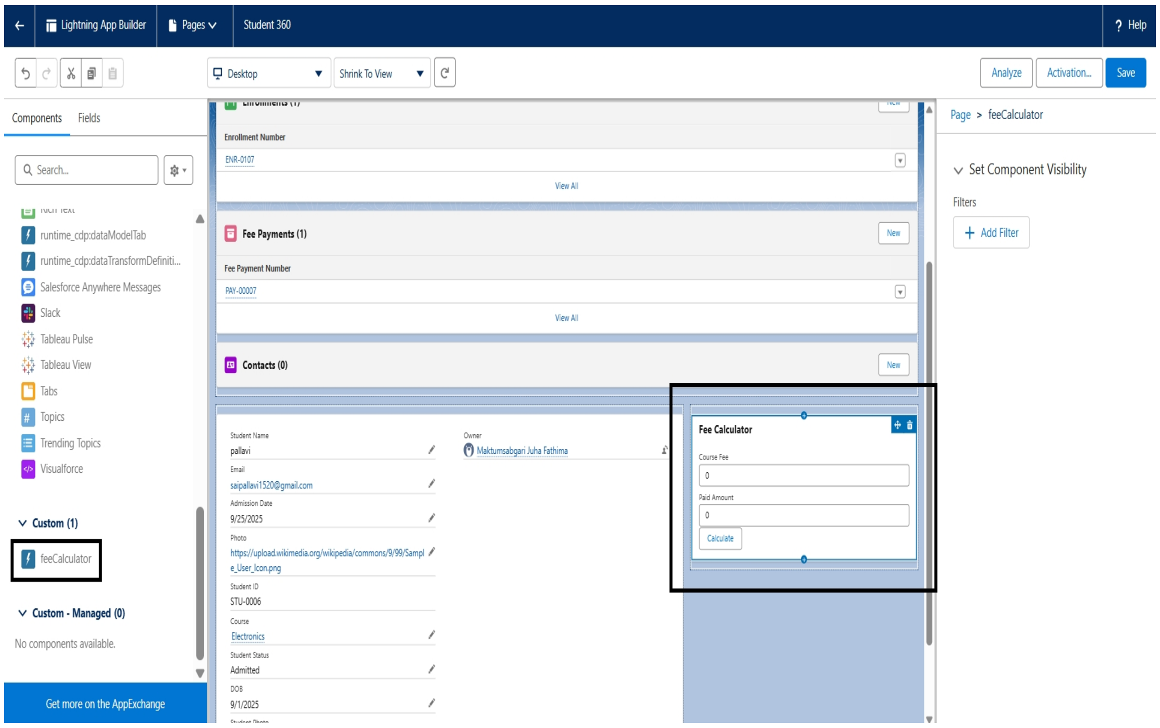
```
sf project deploy start --metadata LightningComponentBundle:feeCalculator -o
```



```
17:47:40.212 Finished SFDX: Create Lightning Web Component
17:50:27.412 Starting SFDX: Deploy This Source to Org

=== Deployed Source
STATE    FULL NAME      TYPE                       PROJECT
PATH
_______  _____________  _________________________
____________________________________________________________________

Created  feeCalculator  LightningComponentBundle
force-app\main\default\lwc\feeCalculator\feeCalculator.html
Created  feeCalculator  LightningComponentBundle
force-app\main\default\lwc\feeCalculator\feeCalculator.js
Created  feeCalculator  LightningComponentBundle
force-app\main\default\lwc\feeCalculator\feeCalculator.js-meta.xml

17:50:30.877 Ended SFDX: Deploy This Source to Org
```

MyDevOrg

5. Add to Lightning Page:
   - Go to Lightning App Builder → Drag feeCalculator onto Student Page.



Enrollment Number
ENR-0107

Fee Payments (1)
Fee Payment Number
PAY-00007

Contacts (0)

Student Name: pallavi
Owner: Maktumsabgari Juha Fathima
Email: saipallavi1520@gmail.com
Admission Date: 9/25/2025
Photo: https://upload.wikimedia.org/wikipedia/commons/9/99/Sample_User_Icon.png
Student ID: STU-0006
Course: Electronics
Student Status: Admitted
DOB: 9/1/2025

Fee Calculator
Course Fee: 0
Paid Amount: 0

Result: Admins can instantly calculate and display outstanding fees inside the Student 360 page.

Results

- **Student 360 Page:** Displays student info and related records in a tabbed layout.
- **Screen Flow:** Enables guided student admission.
- **Fee Calculator LWC:** Interactive tool for fee management.
- **Overall Outcome:** Clean, intuitive, and functional UI for managing student-related data in Salesforce.