

Phase 4

Business Logic & Automation

Step 1: Creating Tabs for Objects

In this step, I created **custom objects** in Salesforce to represent the key entities of the Codesphere Institute Of Technology. For each object, I also created a **tab** so that users can access and manage records easily from the Salesforce app navigation bar.

- Navigated to **Setup** → **Object Manager**.
- Created a **new custom object** for each entity (Student, Faculty, Course, Fee Payment, Enrollment).
- For every object, created a **tab** via:
 - Setup → Tabs → New Tab
 - Selected the object and an icon, then saved.

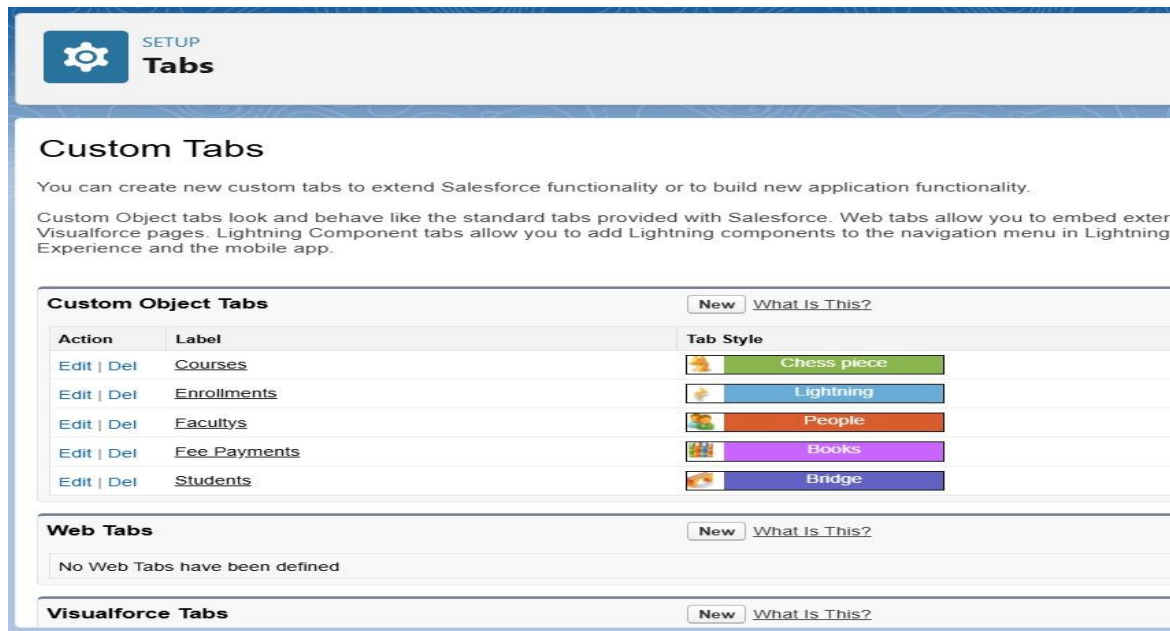


Fig: Creation of Tabs

Step 2: App Creation :

Step 1: Create a New App

- Go to **Setup** → **App Manager**.
- Click on **New Lightning App**.
- Enter **App Name** (Codesphere Institute).
- Upload an **App Logo** to personalize the app.
- Click **Next** to continue.

App Details & Branding

Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.


App Details


*App Name ⓘ
CodeSphere Institute

*Developer Name ⓘ
CodeSphere_Institute

Description ⓘ
Custom app for managing students, courses, faculty, and fees.

App Branding


Image ⓘ


Primary Color Hex Value ⓘ
 #0070D2

[Clear](#)

Org Theme Options
☐ Use the app's image and color instead of the org's custom theme

App Launcher Preview



CodeSphere Institute
Custom app for managing students, courses, faculty, an...

Fig 2:App creation

Step 2: Configure Navigation

- Select **Navigation Style** → **Standard Navigation**.
- Choose whether the app should be available on **Desktop** or **Desktop & Mobile**.
- Click **Next**.

Step 3: Assign User Profiles

- Select the **profiles** that should access this app.
 - Example: System Administrator, Standard User.
- Click **Save and Finish**.

Step 4: Verify the App

- Open the **App Launcher** (grid icon in top-left).
- Search for your app (Codesphere Institute Of Technology).
- Open it and confirm that all required **tabs** (Students, Faculty, Courses, Fee Payment, Enrollments) are visible.
- Test by creating a sample record in each tab to ensure everything works properly.

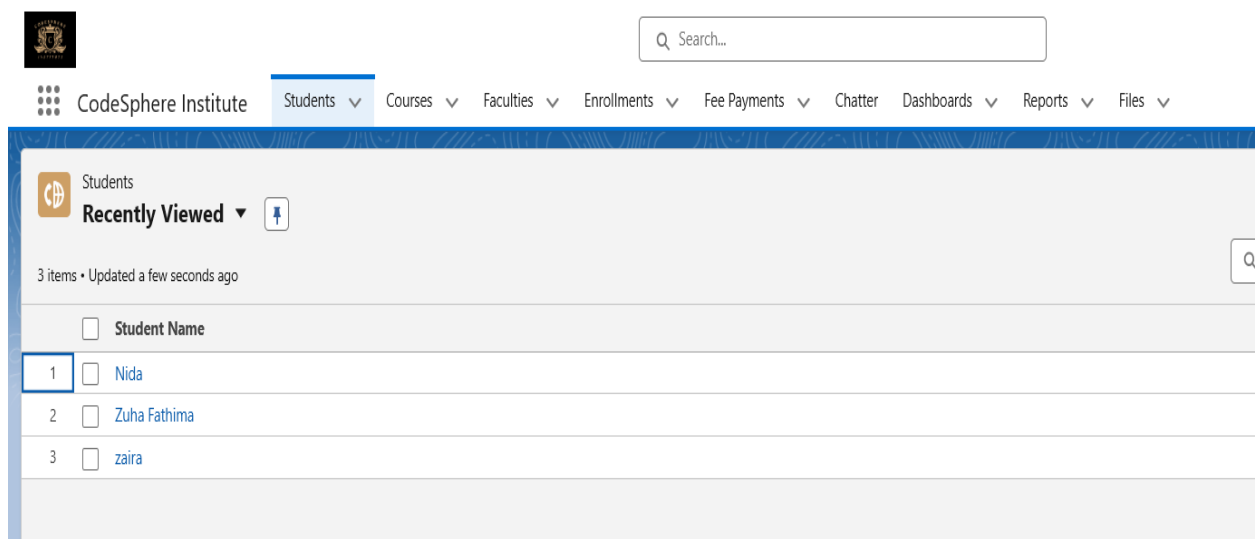


Fig 3:Codesphere app Image

Step 3: Create Validation Rules

Validation Rules are designed to maintain data integrity by ensuring that only valid data is entered in Salesforce objects.

Student Object

- **Rule Name:** DOB_Check
- **Formula:** Date_of_Birth__c >= TODAY()
- **Error Message:** “Date of Birth must be in the past.”
- **Purpose:** Prevents users from entering a future date as a student’s date of birth.

The screenshot shows a Salesforce record for a student named Zuha Fathima. The form includes the following fields and values:

- Student Name:** Zuha Fathima
- Email:** zuhaf8899@gmail.com
- Admission Date:** 9/19/2025
- Photo:** (Empty)
- Student ID:** STU-0001
- Course:** Electronics
- Student Status:** Admitted
- DOB:** 11/14/2025 (Highlighted in yellow)
- Owner:** Maktumsat
- Created By:** Maktumsabqari Juha Fathima, 9/19/2025, 11:05 AM

A red error message box is displayed over the DOB field, stating: "We hit a snag. Review the errors on this page. Date of Birth must be in the past." The form also includes "Cancel" and "Save" buttons at the bottom right.

Fig 4:Checking Validation rule for DOB

Course Object

- **Rule Name:** StartDate_Check

- **Formula:**
- Start_Date__c < TODAY()
- **Error Message:** “Course Start Date cannot be in the past.”
- **Purpose:** Ensures course start dates are valid and in the future.

Fee Payment Object

- **Rule Name:** Amount_Check
- **Formula:**
- Amount__c <= 500
- **Error Message:** “Payment amount must be greater than Five hundred.”
- **Purpose:** Prevents entry of invalid payment amounts.

Fee Payment Validation Rule

[Back to Fee Payment](#)

Validation Rule Detail		Edit	Clone
Rule Name	Amount_Check		
Error Condition Formula	Amount__c <= 500		
Error Message	Payment amount must be greater than Five hundred		
Description			
Created By	<u>Maktumsabgari Juha Fathima</u> , 9/19/2025, 11:17 AM		

Edit
Clone

Fig 5: Validation rule Formula

Step 4: Flow 1 – Admission Process (Auto-create Enrollment + Fee Payment)

Purpose:

Automatically create Enrollment and Fee Payment records when a Student is admitted.

Configuration Steps:

- **Navigate to Flows:**
 - Setup → Flows → New Flow
- **Select Flow Type:**
 - Record-Triggered Flow
- **Object:**
 - Student__c
- **Trigger:**
 - When a record is **created**
- **Entry Condition:**
 - Status__c = "Admitted"
- **Add Elements:**
 - a. Create Enrollment Record**
 - Object: Enrollment__c
 - Set Fields:
 1. Student__c = {!\$Record.Id}
 2. Course__c = {!\$Record.Course__c}
 - b. Create Fee Payment Record**
 - Object: Fee_Payment__c
 - Set Fields:
 1. Student__c = {!\$Record.Id}
 2. Status__c = "Pending"
- **Click On Save → Label = Admission Process**
- **And Activate** the flow

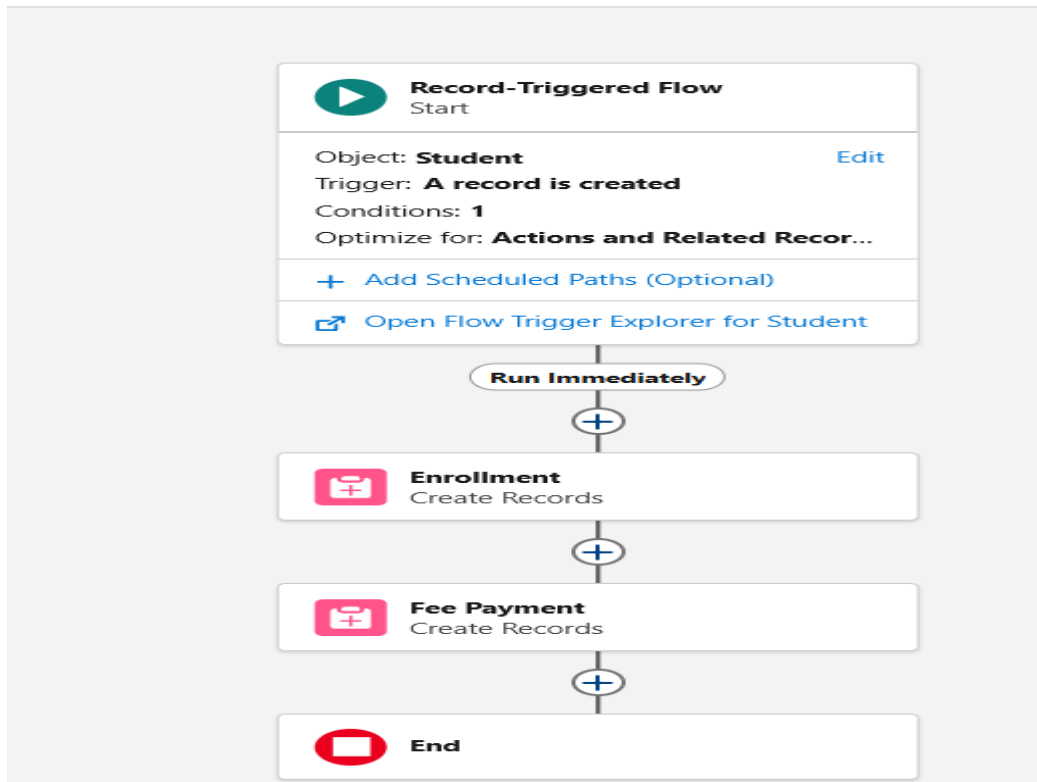


Fig 6: Flow of Admission Process

Outcome:

When a Student record is created with Enrollment_Status__c = "Admitted", corresponding Enrollment and Fee Payment records are automatically generated, reducing manual data entry and ensuring process consistency.

The screenshot shows the 'Student zaira' interface. It has two tabs: 'Related' and 'Details'. Under the 'Related' tab, there are three sections: 'Enrollments (1)', 'Fee Payments (1)', and 'Contacts (0)'. Each section has a 'New' button. The 'Enrollments (1)' section shows 'Enrollment Number' with the value 'ENR-0105' and a 'View All' link. The 'Fee Payments (1)' section shows 'Fee Payment Number' with the value 'PAY-00005' and a 'View All' link. The 'Contacts (0)' section is currently empty.

Fig 7:Output

Step 5: Scheduled Flow – Daily Fee Reminders

Purpose:

Send daily reminders to students for pending fee payments.

Configuration Steps:

- **Navigate to Flows:**
 - Setup → Flows → New Flow
- **Select Flow Type:**
 - Scheduled-Triggered Flow
- **Set Schedule:**
 - Frequency: Daily
 - Start Time: 4.30 PM
- **Object / Records to Process:**
 - Fee_Payment__c
 - Condition: Status__c = "Pending"
- **Add Elements:**
 - Action → **Send Email Alert**
 - Recipient: Student__c.Email
- **Save & Activate**
- **Click On Save → Label = Fee remainder flow**
- **And Activate the flow**

Outcome:

All students with pending fees receive automated daily reminders, ensuring timely payments.

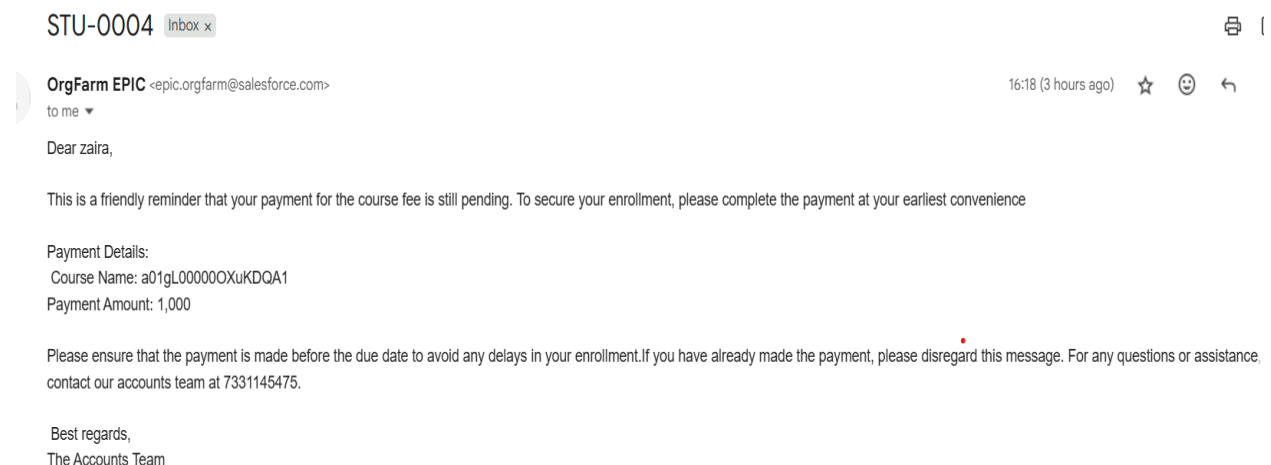


Fig 8:Email Alert

Step 6: Record-Triggered Flows

Enrollment Status Update When Grade Is Entered

Purpose: Automatically update Enrollment status to “Completed” once a grade is entered.

- **Configuration Steps:**
- Setup → Flow → New Flow → Record-Triggered Flow
- Object: Enrollment__c
- Trigger: When a record is **created or updated**
- Condition Requirements:
 - Grade__c **Is Changed**
 - Grade__c **Is Not Null**
- Optimize the Flow for: **Actions and Related Records**
- Add Element → **Update Records**
 - Update the same Enrollment record
 - Set Enrollment_Status__c = "Completed"
- **Click On Save → Label** = Status completed
- **And Activate** the flow

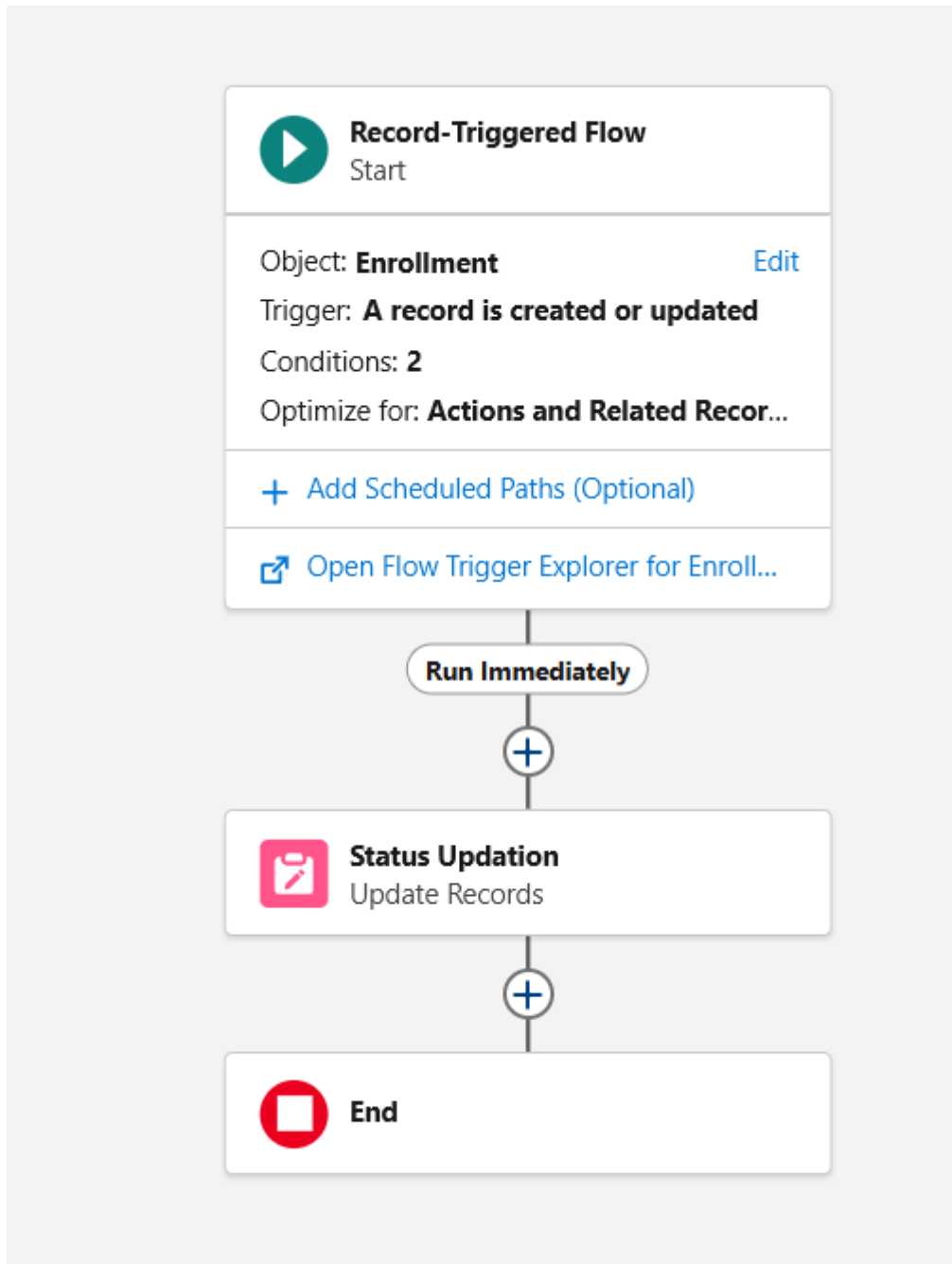



Fig 9:Flow construction

Outcome: Enrollment records are automatically updated to “Completed” when a grade is recorded.


Enrollment
ENR-0101

Related
Details




Enrollment Number	ENR-0101	Owner	 Maktumsabgari Juha Fathima
Student	Zuha Fathima		
Course	Electronics		
Enrollment Date	9/20/2025		
Status	Completed		
Grade	C		
Created By	 Maktumsabgari Juha Fathima , 9/19/2025, 11:39 AM	Last Modified By	 Maktumsabgari Juha Fathima , 9/22/2025, 4:13 AM

Fig 10:Status automatically changed

Fee Payment Status Update When Amount Received

Purpose: Automatically update Fee Payment status to “Paid” when a payment is received.

Configuration Steps:

- Setup → Flow → New Flow → Record-Triggered Flow
- Object: Fee_Payment__c
- Trigger: When a record is **created or updated**
- Condition Requirements:
 - Amount__c > 0
 - Status__c != "Paid"
- Add Element → **Update Records**
 - Update the same Fee Payment record
 - Set Status__c = "Paid"
- Add **Email Alert** → Notify Student of payment received
- **Click On Save → Label** = Fee status Updated
- **And Activate** the flow

Outcome: Fee Payment records are automatically updated, and students are notified, ensuring accurate financial tracking.



Fig 11:Email for clearing of fees

Step 6: Email Alert:

Purpose:

Automatically notify students via email when their enrollment status becomes **Active**, ensuring timely communication about their successful enrollment in the institute.

Configuration Steps:

- Setup → Flow → New Flow → Record-Triggered Flow
- Object: Enrollment__c
- Trigger: When a record is **created or updated**
- Condition Requirements:
 - Status__c = "Active"
- Add **Email Alert** → Notify Student who are enrolled in institute
- **Click On Save → Label = Enroll students**
- **And Activate** the flow

Outcome:

When an enrollment record is created or updated with Status__c = "Active", the

student automatically receives an enrollment confirmation email, improving communication and reducing manual follow-ups.

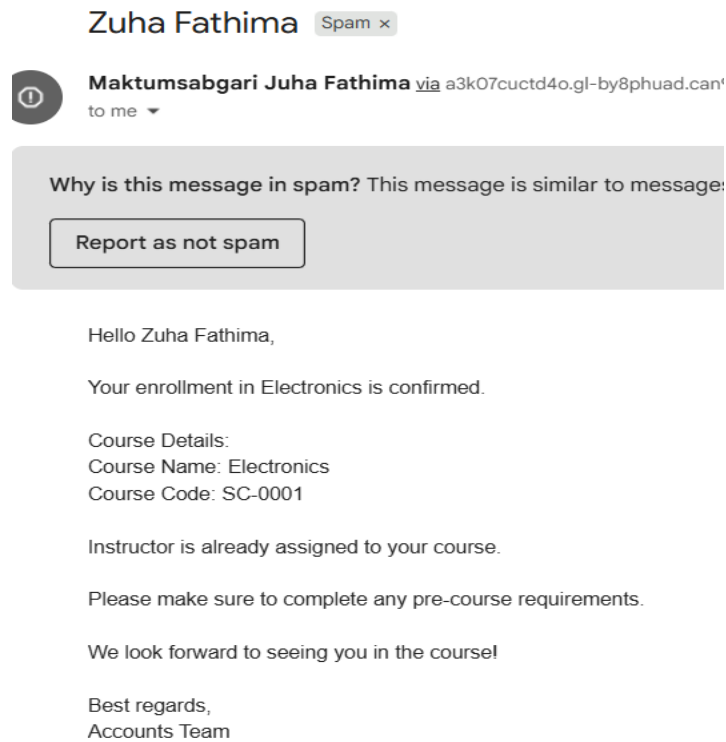


Fig 12:Enrollment Email