

# Salesforce Student Admission Management System

## **Problem Statement:**

Many colleges and universities still rely on manual processes, paper forms, and spreadsheets to manage admissions. These methods lead to inefficiencies, errors, and delays in the admission lifecycle. Applicants lack transparency and real-time updates, while admission officers struggle with high volumes, manual evaluations, and fragmented tracking of applications, enrollments, and fees.

Administrators also face challenges with limited analytics, making it hard to forecast trends or optimize resources. Legacy solutions often fail to provide the customization and unified experience institutions need.

This project addresses these issues by offering a cloud-based, automated platform that streamlines applications, supports structured decision workflows, automates enrollment tracking, and delivers actionable insights through reports and dashboards.

## **Phase 1 (Problem Understanding & Industry Analysis):**

### **Problem Understanding & Industry Analysis**

- **Requirement Gathering**  
Collected requirements from the perspective of a university admissions team: online student application, review workflow, enrollment tracking, and reporting needs.
- **Stakeholder Analysis**  
Identified stakeholders such as Admission Officers, Faculty Reviewers, Students/Applicants, and Administrators, along with their roles and expectations in the admission lifecycle.
- **Business Process Mapping**  
Mapped the end-to-end admission process starting from student application submission → review & evaluation → decision publishing → enrollment confirmation.
- **Industry-specific Use Case Analysis**  
Researched admission challenges faced by educational institutions: high application volumes, manual data entry, lack of real-time tracking, and reporting inefficiencies.

## Phase 2 (Org Setup & Configuration):

In this phase, the Salesforce Developer Org was configured to align with the requirements of the Student Admission Management System. Key configuration activities included:

- **Salesforce Edition Selection**

A Salesforce **Developer Edition** was used to build and test the project, as it provides core CRM features, customization, and LWC development capabilities at no cost.

- **Company Profile Setup**

The company profile was configured with institution details such as **College/University name, locale, time zone, currency, and default language**, ensuring consistency across records.

The screenshot shows the Salesforce Setup interface for 'Company Information'. The left sidebar contains a search bar and a list of settings categories: Company Settings, Business Hours, Calendar Settings, Public Calendars and Resources, Company Information (highlighted), Data Protection and Privacy, Fiscal Year, Holidays, Language Settings, and My Domain. Below these is a search prompt: 'Didn't find what you're looking for? Try using Global Search.' The main content area is titled 'Company Information' and shows the profile for 'VIT-AP University'. It includes links for 'User Licenses', 'Permission Set Licenses', 'Feature Licenses', and 'Usage-based Entitlements'. The 'Organization Detail' section is expanded, showing fields for Organization Name, Primary Contact, Division, Address, Fiscal Year Starts In, Activate Multiple Currencies, Enable Data Translation, Newsletter, Admin Newsletter, Hide Notices About System Maintenance, Hide Notices About System Downtime, Locale Formats, Phone, Fax, Default Locale, Default Language, Default Time Zone, Currency Locale, Used Data Space, Used File Space, API Requests, Streaming API Events, Restricted Logins, Salesforce.com Organization ID, Organization Edition, and Instance. The bottom of the page shows 'Created By' and 'Modified By' information.

Organization Detail	
Organization Name	VIT-AP University
Primary Contact	OrgFarm EPIC
Division	
Address	United States
Fiscal Year Starts In	January
Activate Multiple Currencies	<input type="checkbox"/>
Enable Data Translation	<input type="checkbox"/>
Newsletter	<input checked="" type="checkbox"/>
Admin Newsletter	<input checked="" type="checkbox"/>
Hide Notices About System Maintenance	<input type="checkbox"/>
Hide Notices About System Downtime	<input type="checkbox"/>
Locale Formats	ICU
Phone	
Fax	
Default Locale	English (United States)
Default Language	English
Default Time Zone	(GMT-07:00) Pacific Daylight Time (America/Los_Angeles)
Currency Locale	English (United States) - USD
Used Data Space	486 KB (9%) [View]
Used File Space	3.2 MB (16%) [View]
API Requests, Last 24 Hours	0 (15,000 max)
Streaming API Events, Last 24 Hours	0 (10,000 max)
Restricted Logins, Current Month	0 (0 max)
Salesforce.com Organization ID	00DgL00000BGZ9x
Organization Edition	Developer Edition
Instance	CAN96

Created By: OrgFarm EPIC, 9/9/2025, 2:04 PM  
Modified By: OrgFarm EPIC, 9/12/2025, 2:11 AM

- **Business Hours & Holidays**

Defined admission office working hours (9 to 5 and no working hours on sundays) and academic holidays can also be included to be used in workflows (like application submission SLAs or automated responses)

Organization Business Hours

Select the days and hours that your support team is available. These hours, when associated with escalation rules, determine the time your support team is available. If you enter blank business hours for a day, that means your organization does not operate on that day.

Holidays

Business Hours Detail

Business Hours Name

Default

Business Hours

No Hours

0

Sunday

9:00 AM to 5:00 PM

Monday

9:00 AM to 5:00 PM

Tuesday

9:00 AM to 5:00 PM

Wednesday

9:00 AM to 5:00 PM

Thursday

9:00 AM to 5:00 PM

Friday

9:00 AM to 5:00 PM

Saturday

9:00 AM to 5:00 PM

Active

☒

Created By

OroFarm EPIC

9/9/2025, 2:04 PM

Edit

Edit

## Fiscal Year Settings

Configured fiscal year settings to align with the **academic calendar** for reporting admission data semester/annual basis.

Setup

Organization Fiscal Year Edit: VIT-AP University

To specify the fiscal year type for your organization, choose one of the options below.

Standard Fiscal Year

Custom Fiscal Year

Fiscal Year Information

Your organization can change the fiscal year start month, and specify whether the 1 March 2026, your Fiscal Year setting can be either 2025 or 2026.

Changing the fiscal year shifts fiscal periods and impacts opportunities a month will erase existing forecast adjustments and quotas. Consider exp

Change Fiscal Year Period

Name

VIT-AP University

Fiscal Year Start Month

January

Fiscal Year is Based On

☒ The ending month

☐ The starting month

Save

Cancel

## User Setup & Licenses

- Me (Mallela Sri Vaishnavi) → Admissions Director → Full access
- Simran Sahu (Admissions Manager) → Salesforce License
- Venkat Ram (Admissions Officer) → Salesforce License

### Active Users

On this page you can create, view, and manage users.

To get more licenses, use the Your Account app. [Let's Go](#)

View: [Active Users](#) [Edit](#) [Create New View](#)

Action	Full Name	Alias	Username	Role
<input type="checkbox"/>   <a href="#">Edit</a>	Chatter Expert	Chatter	chatty_00d0l00000b0z9xuah.r8xeixnflumf@chatter.salesforce.com	
<input type="checkbox"/>   <a href="#">Edit</a>	EPIC_OroFarm	OEPIC	epic.37e269dcab10@orofarm.salesforce.com	
<input checked="" type="checkbox"/>   <a href="#">Edit</a>	Ram_Venkat	Ram	venkat.admissions@test.com	Admissions Officer
<input checked="" type="checkbox"/>   <a href="#">Edit</a>	Sahu_Simran	simmy	simran.admissions@test.com	Admissions Manager
<input checked="" type="checkbox"/>   <a href="#">Edit</a>	Sri Vaishnavi_Mallela	sri	srivaishnavimallela24213@agentforce.com	Admissions Director
<input type="checkbox"/>   <a href="#">Edit</a>	User_Integration	inteq	integration@00d0l00000b0z9xuah.com	
<input type="checkbox"/>   <a href="#">Edit</a>	User_Security	sec	insightssecurity@00d0l00000b0z9xuah.com	

## Profiles & Roles

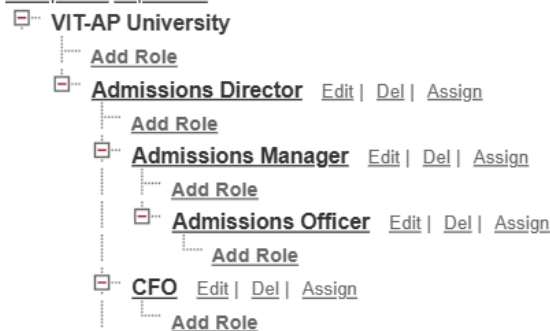
Configured **custom profiles** (e.g., Admission Officer, Student Counselor) and **role hierarchy** (University → Admissions Director → Admission Manager → Admission Officer) to reflect the institution's structure.

### Creating the Role Hierarchy

You can build on the existing role hierarchy shown on this page.

#### Your Organization's Role Hierarchy

[Collapse All](#) [Expand All](#)



## Permission Sets

- Permission Sets provide extra access without changing user profiles.

### Permission Sets

On this page you can create, view, and manage permission sets.


[Created by me.](#) [Edit](#) | [Delete](#) | [Create New View](#)

<input type="checkbox"/>	Action	Permission Set Name ↑	Description
<input type="checkbox"/>	<a href="#">Del</a>   <a href="#">Clone</a>	<a href="#">Admissions Admin Access</a>	Full access to admission records and settings
<input type="checkbox"/>	<a href="#">Del</a>   <a href="#">Clone</a>	<a href="#">Admissions Manager Access</a>	Manage applications and reports
<input type="checkbox"/>	<a href="#">Del</a>   <a href="#">Clone</a>	<a href="#">Admissions Officer Access</a>	Create and process applications
<input type="checkbox"/>	<a href="#">Del</a>   <a href="#">Clone</a>	<a href="#">Application Decision Override</a>	Allow selected users to edit decision fields on Student Application
<input type="checkbox"/>	<a href="#">Del</a>   <a href="#">Clone</a>	<a href="#">Enrollment Management Access</a>	Allows editing enrollment status and fee tracking for Enrolled Students
<input type="checkbox"/>	<a href="#">Del</a>   <a href="#">Clone</a>	<a href="#">Reports &amp; Dashboards Access</a>	Allows running reports and viewing dashboards

## Organization-Wide Defaults (OWD)

Set baseline record-level security:

- Student Applications → **Private** (only owners & managers can see)
- Courses → **Private** (since they're institution-wide)
- Enrolled Students → **Private** (sensitive student details)


<div>  <div> <div>SETUP</div> <div>Sharing Settings</div> </div> </div>			
Academic Record	Public Read/Write	Private	<input checked="" type="checkbox"/>
Course	Public Read/Write	Private	<input checked="" type="checkbox"/>
Enrolled Student	Public Read/Write	Private	<input checked="" type="checkbox"/>
Enrollment Course	Public Read/Write	Private	<input checked="" type="checkbox"/>
Program	Public Read/Write	Private	<input checked="" type="checkbox"/>
Student Application	Public Read/Write	Private	<input checked="" type="checkbox"/>

## Login Access Policies

- Enabled Administrators can Log in as Any User.
- Allows Admin to test features as Admin (Director), Manager, Officer.

### Login Access Policies

Control which support organizations your users can grant login access to.

 Changes Saved

Manage Support Options

Save

Cancel

Setting	Enabled
Administrators Can Log in as Any User	<input checked="" type="checkbox"/>

- **Dev Org Setup**

Installed Salesforce CLI and linked VS Code to the Dev Org for **metadata deployment and retrieval**.

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  HISTORY
PS C:\Users\HP\OneDrive\Desktop\Salesforce crm\StudentAdmissionCRM> sfdx force:org:list

```

	Alias	Username	Org Id	Status
	MySandbox	srivaishnavimallela24213@agentforce.com	00DgL00000BGZ9xUAH	Connected

- **Deployment Basics**


Learned deployment methods:

- **Change Sets** (Admin way)
- **SFDX CLI & GitHub** (Developer way, used in this project)

## Phase 3 (Data Modelling & Relationships):

### Standard & Custom Objects:

- **Standard objects:** User, Contact for student contacts and admission officers.
- **Custom objects:**
  - Student\_Application\_\_c – stores student application details (name, DOB, email, GPA, program applied)
  - Enrolled\_Student\_\_c – tracks enrolled students and links to Student Application
  - Course\_\_c – stores available courses
  - Enrollment\_Course\_\_c – junction object linking students to courses
  - Program\_\_c – stores programs offered by the institution
  - Academic\_Records\_\_c – stores student grades and transcripts

<div><div>SETUP</div><div> <b>Object Manager</b></div><div>2 Items, Sorted by Label</div></div> <div><input type="text" value="stude"/><div>Schema BuilderCreate ▾</div></div>						
LABEL	▲	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Enrolled Student		Enrolled_Student__c	Custom Object		9/17/2025	✓ ▾
Student Application		Student_Application__c	Custom Object		9/12/2025	✓ ▾

LABEL	▲	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Course		Course__c	Custom Object		9/13/2025	✓ ▾
Enrollment Course		Enrollment_Course__c	Custom Object		9/13/2025	✓ ▾

LABEL	▲	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Program		Program__c	Custom Object		9/17/2025	✓ ▾

LABEL	▲	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Academic Record		Academic_Record__c	Custom Object		9/13/2025	✓ ▾

Fields:

- Defined fields for each object using appropriate data types as follows:

Student\_Application\_\_c:

Fields & Relationships 17 Items, Sorted by Field Label		
FIELD LABEL	FIELD NAME	DATA TYPE
Admission Officer	Admission_Officer__c	Lookup(User)
Applicant First Name	Applicant_First_Name__c	Text(80)
Applicant Full Name	Applicant_Full_Name__c	Formula (Text)
Applicant Last Name	Applicant_Last_Name__c	Text(80)
Application Number	Name	Auto Number
Created By	CreatedById	Lookup(User)
Date of Birth	Date_of_Birth__c	Date
Decision Comments	Decision_Comments__c	Long Text Area(32000)
Email	Email__c	Email
High School GPA	High_School_GPA__c	Number(3, 2)
Last Modified By	LastModifiedById	Lookup(User)
Notes	Notes__c	Long Text Area(32768)
Owner	OwnerId	Lookup(User,Group)
Phone	Phone__c	Phone
Program	Program__c	Lookup(Program)
Status	Status__c	Picklist
Submitted Date	Submitted_Date__c	Date/Time

Enrolled\_\_Student\_\_c:

Fields & Relationships 12 Items, Sorted by Field Label		
FIELD LABEL	FIELD NAME	DATA TYPE
Application	Application__c	Lookup(Student Application)
Created By	CreatedById	Lookup(User)
End Date	End_Date__c	Date
Enrollment Number	Name	Auto Number
Enrollment Status	Enrollment_Status__c	Picklist
Fees Paid	Fees_Paid__c	Currency(16, 2)
Last Modified By	LastModifiedById	Lookup(User)
Name	Name__c	Text(30)
Owner	OwnerId	Lookup(User,Group)
Payment Status	Payment_Status__c	Picklist
Start Date	Start_Date__c	Date
Student Contact	Student_Contact__c	Lookup(Contact)

# Course\_\_c:

## Fields & Relationships

8 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Course Code	Course_Code__c	Text(20)
Course Name	Name	Text(80)
Created By	CreatedById	Lookup(User)
Credits	Credits__c	Number(2, 0)
Current Student Count	Current_Student_Count__c	Number(18, 0)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Program	Program__c	Lookup(Program)

# Program\_\_c:

## Fields & Relationships

10 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Active	Active__c	Checkbox
Created By	CreatedById	Lookup(User)
Department	Department__c	Picklist
Description	Description__c	Long Text Area(32768)
Duration (Months)	Duration__c	Number(3, 0)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Program Code	Program_Code__c	Text(20)
Program Name	Name	Text(80)
Tuition Fees	Tuition_Fees__c	Currency(16, 2)

# Academic\_Record\_\_c:

## Fields & Relationships

9 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Academic Record Name	Name	Text(80)
Applicant	Applicant__c	Lookup(Student Application)
Created By	CreatedById	Lookup(User)
Degree	Degree__c	Text(255)
GPA	GPA__c	Number(3, 2)
Institution	Institution__c	Text(255)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Year Completed	Year_Completed__c	Number(4, 0)



## Record Types & Page Layouts:

- Record types to differentiate program types or application types (optional).
- Customized **page layouts** for each object to show key fields:
  - Student\_Application → applicant details, decision picklist, comments, uploaded documents
  - Enrolled\_Student → enrollment info, fees, selected courses
  - Home\_Page\_Default → Home Page layout of my App, consists of rich text, reports, recent records, list view, recent items, quicklinks.
- Compact layouts for mobile and Lightning pages for quick info access

## Lightning record page of Student\_Application:

**Application Details**

Applicant First Name	Vikram
Applicant Last Name	Iyer
Email	vikram.iyer@email.com
Program	BEng Mechanical Engineering
Status	Waitlisted
Submitted Date	9/16/2025, 11:00 AM
Decision Comments	Application Decision

Upload Documents  
[Upload Files](#) Or drop files

**Decision**

Decision  
Select an Option

Comments

[Submit Decision](#)

## Lightning record page of Enrolled\_Student:

**Enroll Student**

Select courses to enroll the student in:

Available Courses

Available

- Business Strategy
- Data Structures
- Financial Management
- Fluid Mechanics
- Operating System
- Thermodynamics

Selected

[Enroll](#)

**Enrollment Information**

Enrollment Number	ENR-00006	Owner	Mallela Sri Vaishnavi
Application	APP-00006		
Student Contact			
Enrollment Status	Active		
Start Date	7/5/2025		
End Date	5/30/2028		
Fees Paid	\$10,000.00		
Payment Status	Pending		
Name			

Created By: Mallela Sri Vaishnavi, 9/17/2025, 10:27 PM

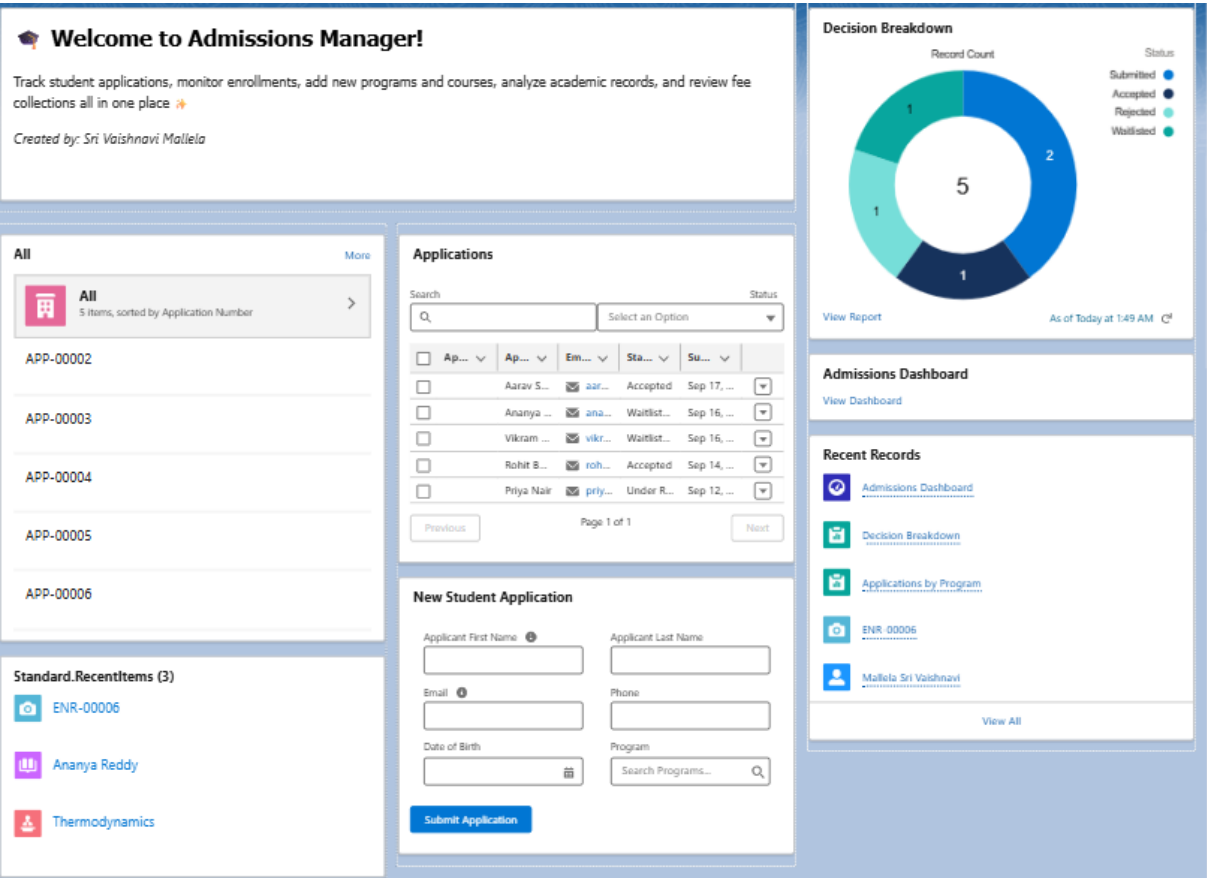
Last Modified By: Mallela Sri Vaishnavi, 9/17/2025, 10:27 PM

**Enrollment Courses (3)**

- Business Strategy
- Fluid Mechanics
- Thermodynamics

[View All](#)

Lightning record of Home\_Page\_Default:



Compact Layout for mobile:

Student Application Compact Layout

Student Application

[« Back to Student Application](#)

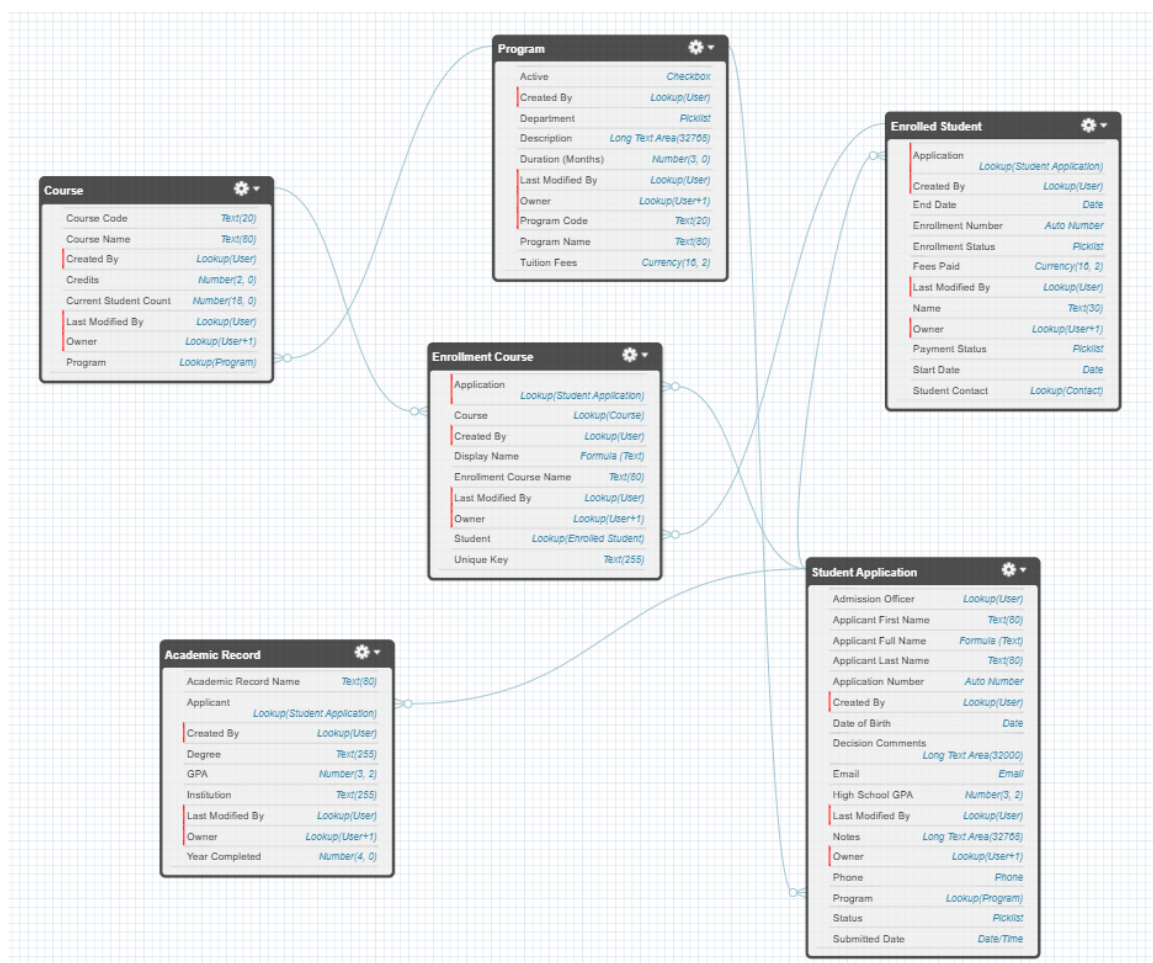
Compact Layout Detail	<div>EditCloneDeleteCompact Layout Assignment</div>
Label	Student Application
API Name	Student_Application
Included Fields	Application Number Applicant First Name Applicant Last Name Applicant Full Name Date of Birth Email Phone High School GPA Program Status
Created By	Mallela Sri Vaishnavi, 9/18/2025, 10:17 PM

EditCloneDeleteCompact Layout Assignment

## Relationships:

- Lookup and Master-Detail relationships:
  - Enrolled\_Student\_\_c → Student\_Application\_\_c : Lookup (required)
  - Enrollment\_Course\_\_c → Enrolled\_Student\_\_c : Lookup
  - Enrollment\_Course\_\_c → Course\_\_c : Lookup
  - Academic\_Records\_\_c → Student\_\_c : Lookup
  - Academic\_Records\_\_c → Course\_\_c : Lookup
- Junction Objects:
  - Enrollment\_Course\_\_c enables many-to-many tracking between students and courses.

## Schema Builder :



## Phase 4 (Process Automation (Admin)):

**Objective:** Automate repetitive administrative tasks in the admission lifecycle to ensure accuracy, save time, and provide a seamless applicant experience.

### Validation Rules:

Ensured data quality and consistency in admission workflow.

- Implemented rules in **Student\_Application\_\_c**, **Academic\_Record\_\_c**, **Enrollment\_Course** object:
  1. **Require\_Ddecision\_Comments\_On\_Reject** – Prevents rejecting an application unless decision comments are added.
    - Example: If Admission Officer tries to reject without filling **Comments\_\_c**, an error appears.
  2. **Require\_Program\_When\_Accepted** – Forces the officer to select a **Program\_\_c** before marking the application as Accepted.
    - Example: Officer cannot approve without assigning a program.
  3. **Showsubmitteddate** – Ensures **Enrollment\_Date\_\_c** is entered before marking the application as Accepted.
    - Example: Students cannot be accepted without enrollment date filled.
  4. **Validate\_GPA\_Range** – Ensures GPA entered is between **0.0** and **4.0**.
    - Example: Prevents saving record if GPA = 4.5 or GPA = -1.
  5. **EndDate\_on\_or\_after\_StartDate** -- Used in **Courses / Programs** to validate date ranges.
    - Example: Prevents creating a course/program with an **End Date** earlier than **Start Date**.

Validation Rules					New
1 Items, Sorted by Rule Name					
RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY	
Validate GPA Range	GPA	GPA must be between 0.0 and 4.0	✓	Mallela Sri Vaishnavi, 9/13/2025, 3:19 AM	▼

Validation Rules					New
3 Items, Sorted by Rule Name					
RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY	
<a href="#">Require Decision Comments On Reject</a>	Top of Page	Please add decision comments when rejecting an application.	✓	Mallela Sri Vaishnavi, 9/12/2025, 4:38 AM	▼
<a href="#">Require Program When Accepted</a>	Status	Program must be selected before marking application as Accepted.	✓	Mallela Sri Vaishnavi, 9/15/2025, 1:56 AM	▼
<a href="#">Showsubmitteddate</a>	Top of Page	Enrollment Date must be entered before marking the application as Accepted.	✓	Mallela Sri Vaishnavi, 9/15/2025, 1:58 AM	▼

Validation Rules					New
1 Items, Sorted by Rule Name					
RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY	
<a href="#">EndDate on or after StartDate</a>	End Date	End Date must be the same as or after Start Date.	✓	Mallela Sri Vaishnavi, 9/12/2025, 11:28 PM	▼


# Workflow Rules:

## Set Enrollment Status to Confirmed

- **Objective:** Automatically update a student's enrollment status once the admission fee is paid.
- **Object:** Enrolled\_Student\_\_c
- **Evaluation Criteria:** Rule runs when a record is created, and every time it's edited to meet the condition.
- **Rule Criteria:**
  - Fee\_Paid\_\_c = TRUE

## Field Update Action: Enrollment Status Confirmed

- **Action Name:** Enrollment Status Confirmed
- **Field to Update:** Enrollment\_Status\_\_c
- **New Value:** Confirmed


**Workflow Rules**

Workflow Rule

Set Enrollment Confirmed

Help for this Page

Go with the flow! With Flow Builder, the future of low-code automation, you can do everything you do with workflow rules — and more! Salesforce plans to retire workflow rules and recommends building automation in Flow Builder.
[Tell Me More](#)
[Migrate your workflow rules to flows](#)

Workflow Rule Detail

edit

clone

deactivate

Rule Name	Set Enrollment Confirmed	Object	Enrolled Student
Active	✓	Evaluation Criteria	Evaluate the rule when a record is created, and any time it's edited to subsequently meet criteria.
Description			
Rule Criteria	Enrolled Student: Fee Paid NOT EQUAL TO 0		
Created by	Mallela Sri Vaishnavi, 9/19/2025, 12:28 AM	Modified by	Mallela Sri Vaishnavi, 9/19/2025, 12:29 AM

Workflow Actions

edit

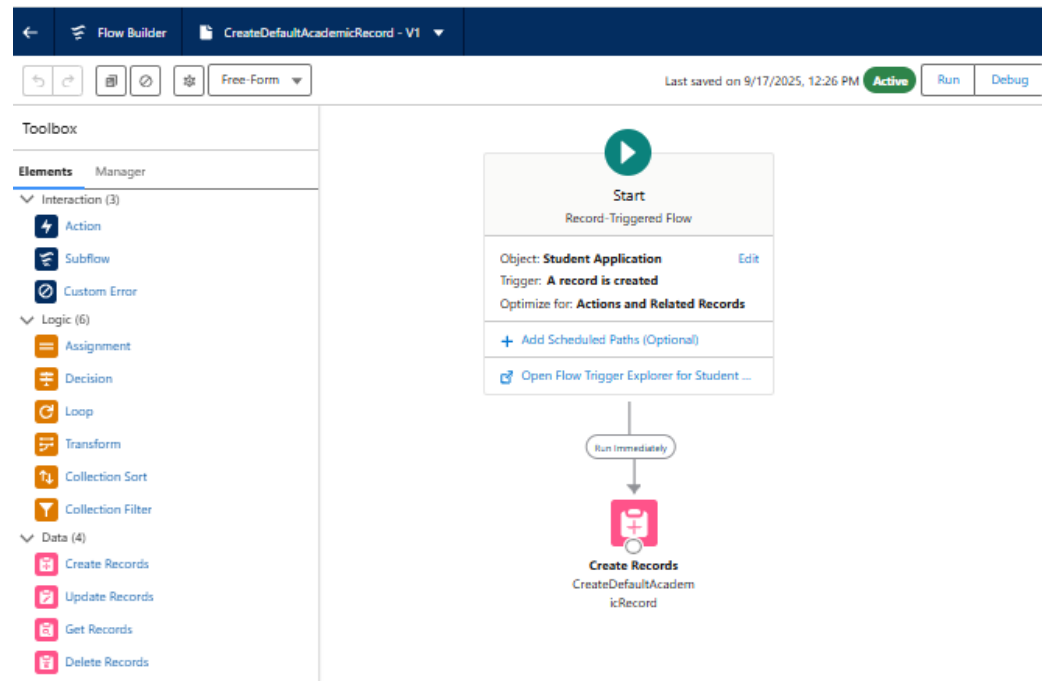
Immediate Workflow Actions

Type	Description
Field Update	Enrollment Status Confirmed

## Flow Builder:

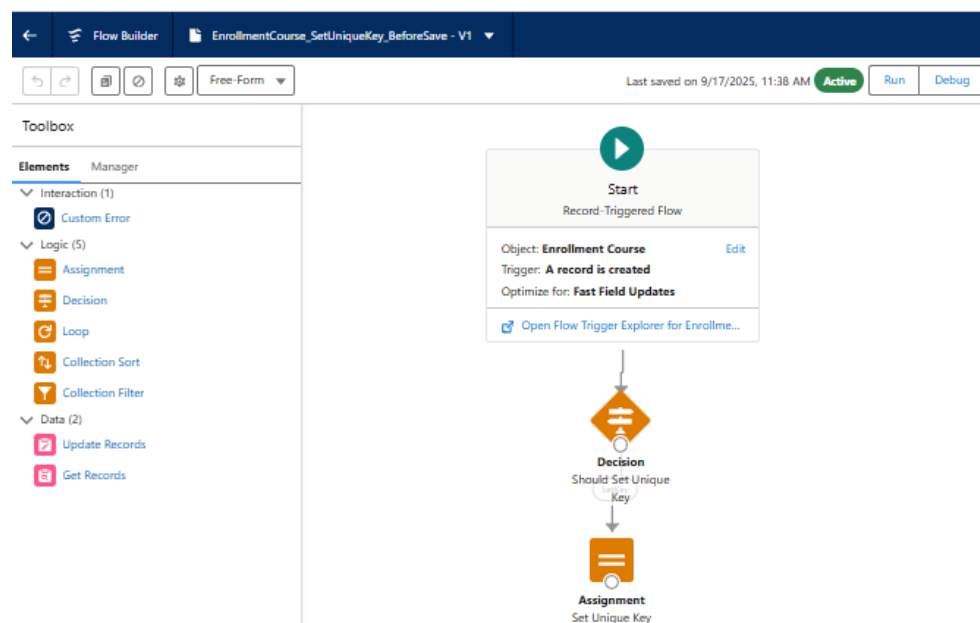
### 1) CreateDefaultAcademicRecord (Autolaunched Flow)

- Automatically creates a default academic record when a new student application is submitted.



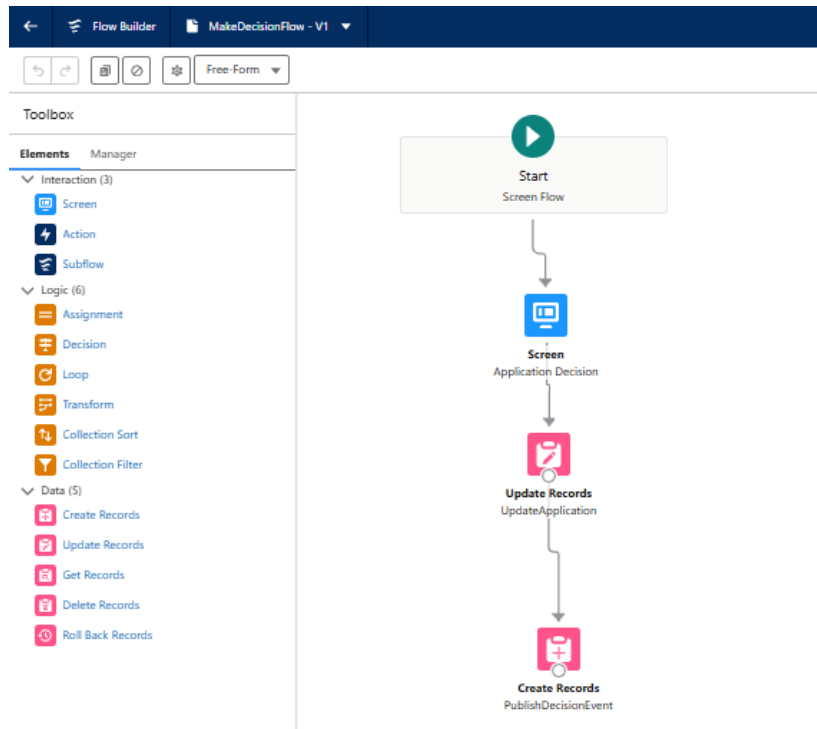
### 2) EnrollmentCourse\_SetUniqueKey\_BeforeSave (Autolaunched Flow)

- Generates a unique key for each enrollment-course record before save.
- Ensures data integrity and prevents duplicates.



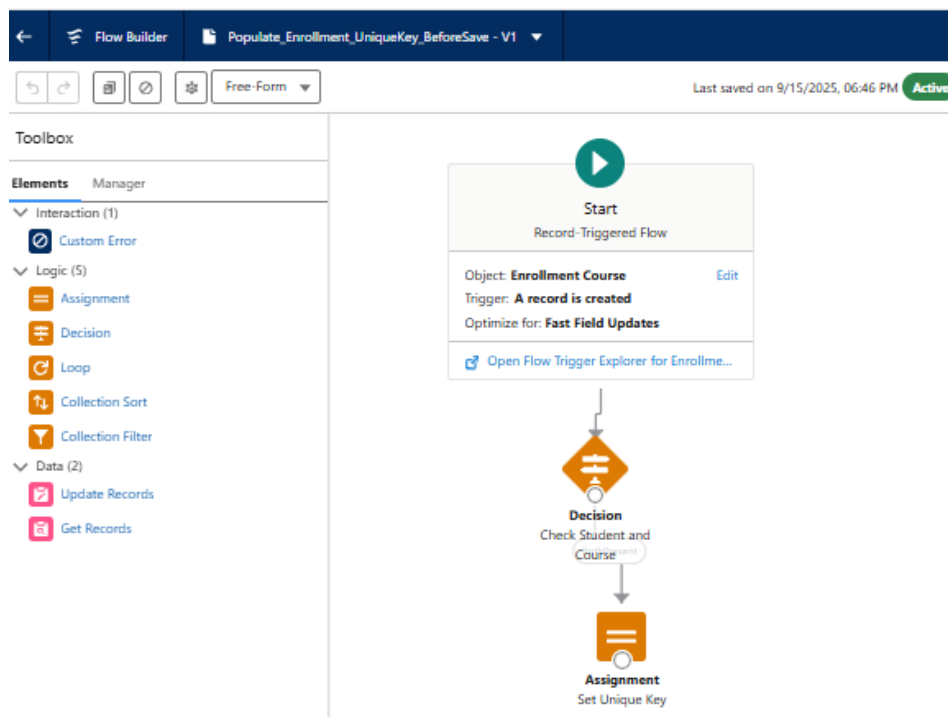
### 3) MakeDecisionFlow (Screen Flow)

- Guided screen flow for admission officers to mark applications as Accepted, Rejected, or Waitlisted.
- Collects decision comments and updates the record.



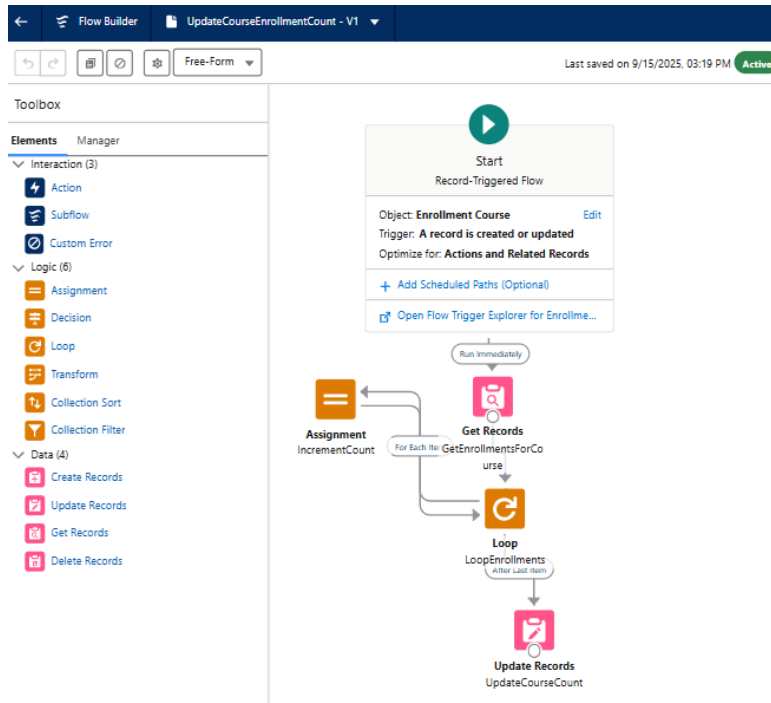
### 4) Populate\_Enrollment\_UniqueKey\_BeforeSave (Autolaunched Flow)

- Populates a unique identifier on enrollment records at save time.



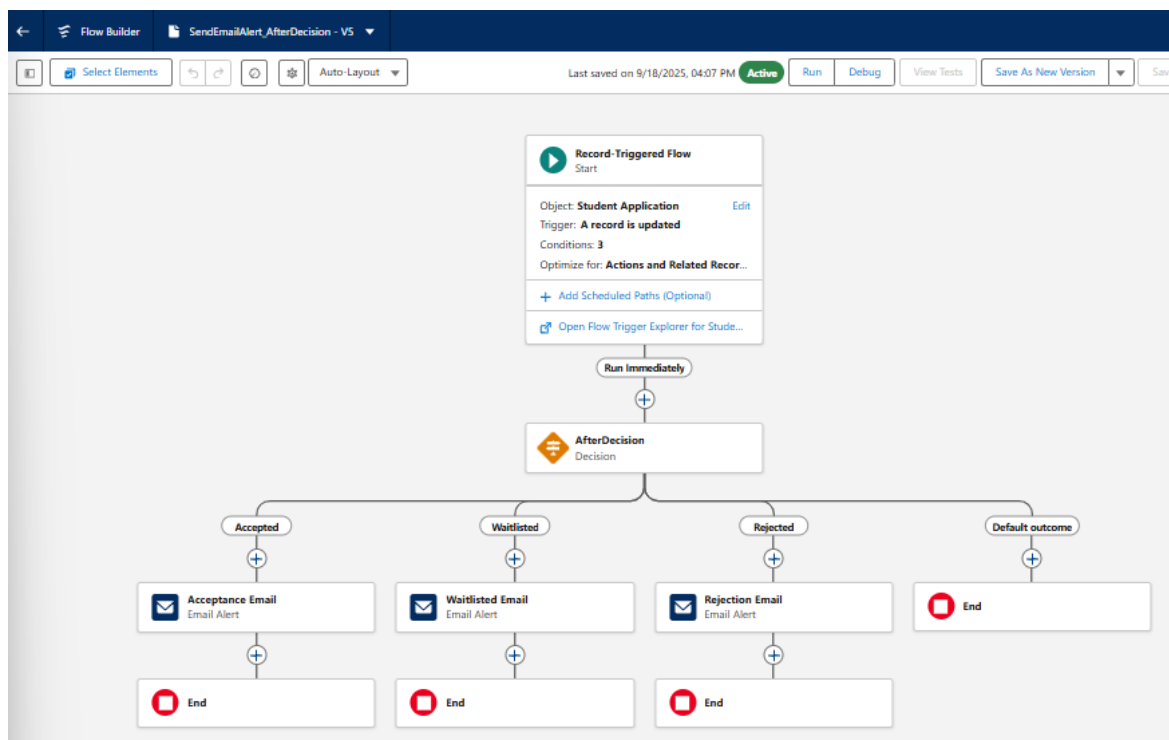
## 5) UpdateCourseEnrollmentCount (Autolaunched Flow)

- Updates the number of enrolled students whenever enrollment changes.
- Useful for reporting and capacity management.



## 6) SendEmailAlert\_AfterDecision (Autolaunched Flow)

- Sends an **acceptance** / **rejection** / **waitlist** email to applicants after a decision is made.





# Email Templates in Project:

To ensure timely and consistent communication with applicants, **Email Templates** were configured and connected to Flows/Email Alerts.

## All Email Alerts

Email alerts are used to send emails from a flow or other automation.

View: All Email Alerts Create New View

New Email Alert				
Action	Description ↑	Email Template Name	Object	Last Modified Date
<a href="#">Edit</a>   <a href="#">Del</a>	<a href="#">Alert: send acceptance email to applicant</a>	<a href="#">Admission Acceptance Notification</a>	Student Application	9/18/2025
<a href="#">Edit</a>   <a href="#">Del</a>	<a href="#">Rejected Student Notification</a>	<a href="#">Admission Rejection Notification</a>	Student Application	9/18/2025
<a href="#">Edit</a>   <a href="#">Del</a>	<a href="#">Waitlisted Student Notification</a>	<a href="#">Admission Waitlisted Notification</a>	Student Application	9/18/2025

## Implemented Templates

- Admission Acceptance Notification
  - Type: Text
  - Purpose: Sent automatically when an applicant is marked Accepted

## Email Template:

Email Template

Send Test and Verify Merge Fields

Subject

Congratulations – You're Accepted!

Plain Text Preview

Dear {Student\_Application\_\_c.Applicant\_First\_Name\_\_c},  
Congratulations! We are thrilled to inform you that you have been officially accepted into the {Student\_Application\_\_c.Program\_\_c} program. Your hard work, dedication, and achievements have truly set you apart, and we are excited to welcome you into our academic community.  
  
This is a remarkable milestone, and we are confident that you will thrive in this program while making valuable contributions along the way.  
  
Next Steps:  
1)Log in to the student portal and complete your enrollment forms.  
2)Submit your program fees to secure your seat.  
  
If you have any questions or need assistance, simply reply to this email or reach out to our admissions office—we're here to help.  
  
Warm regards,  
Admissions Team  
Sri Vaishnavi Mallela

Attachments

Attach File

Action	File Name	Size	Last Modified
<a href="#">Edit</a>   <a href="#">Del</a>   <a href="#">View</a>	<a href="#">Congratulations Letter.png</a>	1.02MB	9/18/2025, 2:

## Email Alert:

Email Alert

Alert: send acceptance email to applicant

[Rules Using This Email Alert \(0\)](#) | [Approval Processes Using This Email Alert \(0\)](#) | [Enrollment Processes Using This Email Alert \(0\)](#)

Email Alert Detail

EditDeleteClone

Description

Alert: send acceptance email to applicant

Email Template

[Admission Acceptance Notification](#)

Unique Name

Alert\_send\_acceptance\_email\_to\_applicant

Object

Student Application

From Email Address

Current User's email address

Recipients

Additional Emails

[sriVaishnavimallela24@gmail.com](#)

Created by

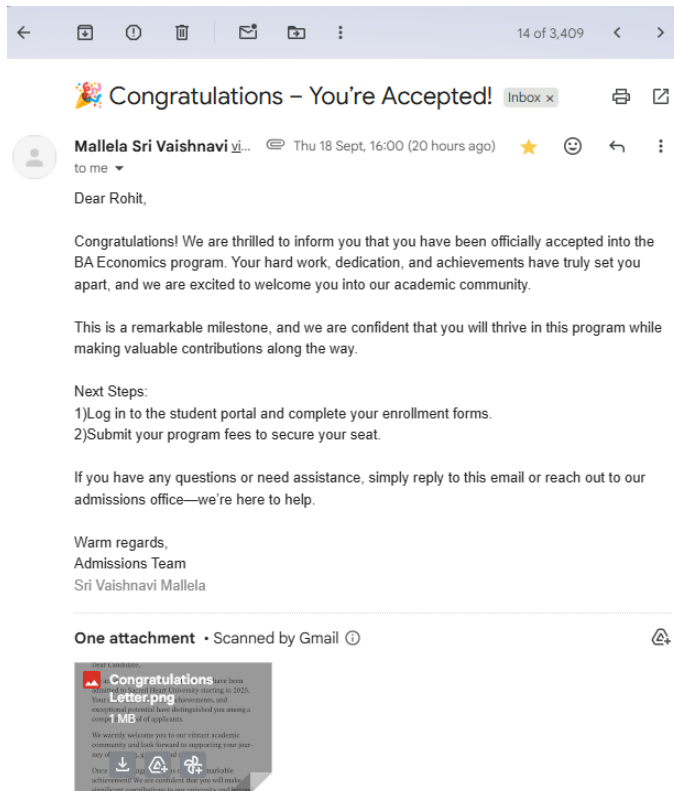
Mallela Sri Vaishnavi, 9/18/2025, 2:24 AM

Modified by

Mallela Sri Vaishnavi, 9/18/2025, 2:24 AM

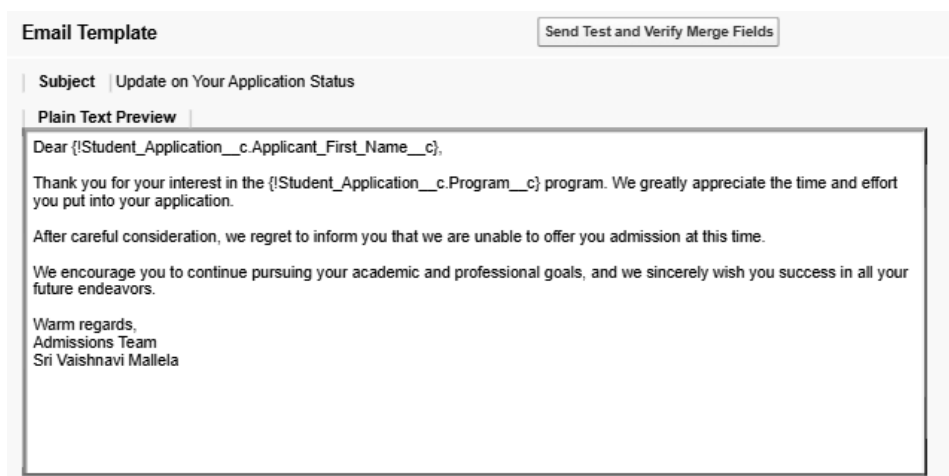
EditDeleteClone

## Email Notification:



- **Admission Rejection Notification**
  - **Type:** Text
  - **Purpose:** Sent when an applicant's status changes to **Rejected**.

## Email Template:



Email Alert:

Email Alert

Rejected Student Notification

Rules Using This Email Alert (0)

Approval Processes Using This Email Alert (0)

Enrollment Processes Using This Email Alert (0)

Email Alert Detail

Edit

Delete

Clone

Description	Rejected Student Notification	Email Template	Admission Rejection Notification
Unique Name	Rejected_Student_Notification	Object	Student Application
From Email Address	Current User's email address		
Recipients			
Additional Emails	sriVaishnavimallela24@gmail.com		
Created By	Mallela Sri Vaishnavi, 9/18/2025, 2:49 AM	Modified By	Mallela Sri Vaishnavi, 9/18/2025, 2:49 AM

Edit

Delete

Clone

Email Notification:

Update on Your Application Status

Inbox x

Print

Link

Mallela Sri Vaishnavi via 2gp... Thu 18 Sept, 16:12 (20 hours ago)

★

😊

↩

⋮

to me ▼

Dear Vikram,

Thank you for your interest in the BEng Mechanical Engineering program. We greatly appreciate the time and effort you put into your application.

After careful consideration, we regret to inform you that we are unable to offer you admission at this time.

We encourage you to continue pursuing your academic and professional goals, and we sincerely wish you success in all your future endeavors.

Warm regards,  
Admissions Team  
Sri Vaishnavi Mallela

- Admission Waitlisted Notification
  - Type: Text
  - Purpose: Sent when an applicant is **Waitlisted**, giving transparency to students about their status.

Email Template:

Email Template

Send Test and Verify Merge Fields

Subject

Update on Your Application Status

Plain Text Preview

Dear {!Student\_Application\_\_c.Applicant\_First\_Name\_\_c},

Thank you for applying to the {!Student\_Application\_\_c.Program\_\_c} program. After careful review, we would like to inform you that your application is currently on our waitlist.

This means that while your profile is strong, we are awaiting final enrollment confirmations before extending additional offers.

We appreciate your patience and will notify you as soon as an update is available.

If you have any questions, feel free to reach out to our admissions office.

Warm regards,  
Admissions Team  
Sri Vaishnavi Mallela

## Email Alert:

Email Alert

Waitlisted Student Notification

Rules Using This Email Alert (0) | Approval Processes Using This Email Alert (0) | Enrollment Processes Using This Email Alert (0)

Email Alert Detail

edit

delete

clone

Description	Waitlisted Student Notification	Email Template	Admission Waitlisted Notification
Unique Name	Waitlisted_Student_Notification	Object	Student Application
From Email Address	Current User's email address		
Recipients			
Additional Email	srivaishnavimallela24@gmail.com		
Created by	Mallela Sri Vaishnavi, 9/18/2025, 2:48 AM	Modified by	Mallela Sri Vaishnavi, 9/18/2025, 3:47 AM

edit

delete

clone

## Email Notification:

Update on Your Application Status 

Inbox x

S

Sri Vaishnavi Mallela <srivaishnavimallel... 12:42 (0 minutes ago) to me

Dear Ananya,

Thank you for applying to the MBA Business Administration program. After careful review, we would like to inform you that your application is currently on our waitlist.

This means that while your profile is strong, we are awaiting final enrollment confirmations before extending additional offers.

We appreciate your patience and will notify you as soon as an update is available.

If you have any questions, feel free to reach out to our admissions office.

Warm regards,  
Admissions Team  
Sri Vaishnavi Mallela

## Field Updates:

- Enrollment Status Confirmed
  - Action Name: Enrollment Status Confirmed
  - Object: Enrolled\_Student\_\_c
  - Field to Update: Enrollment\_Status\_\_c
  - Operation: Update field value
  - New Value: Completed

### Purpose:

- Ensures that once the workflow condition (e.g., Fee Paid = true) is satisfied, the student's enrollment record is automatically updated.
- Reduces manual intervention for the admissions team.

## Edit Subscription

Schedule dashboard refreshes and subscribe to receive results.

### Settings

Frequency

[Daily](#) [Weekly](#) [Monthly](#)

Days

[Sun](#) [Mon](#) [Tue](#) [Wed](#) [Thu](#) [Fri](#) [Sat](#)

Time

4:00 PM ▼

### Recipients

⚠ Recipients see the same report data as the person running the report.

☒ Receive new results by email when dashboard is refreshed. ⓘ

Send email to

Me, OrgFarm EPIC

[Edit Recipients](#)

[Unsubscribe](#) [Cancel](#) [Save](#)

## Phase 5 (Apex Programming)- Developer

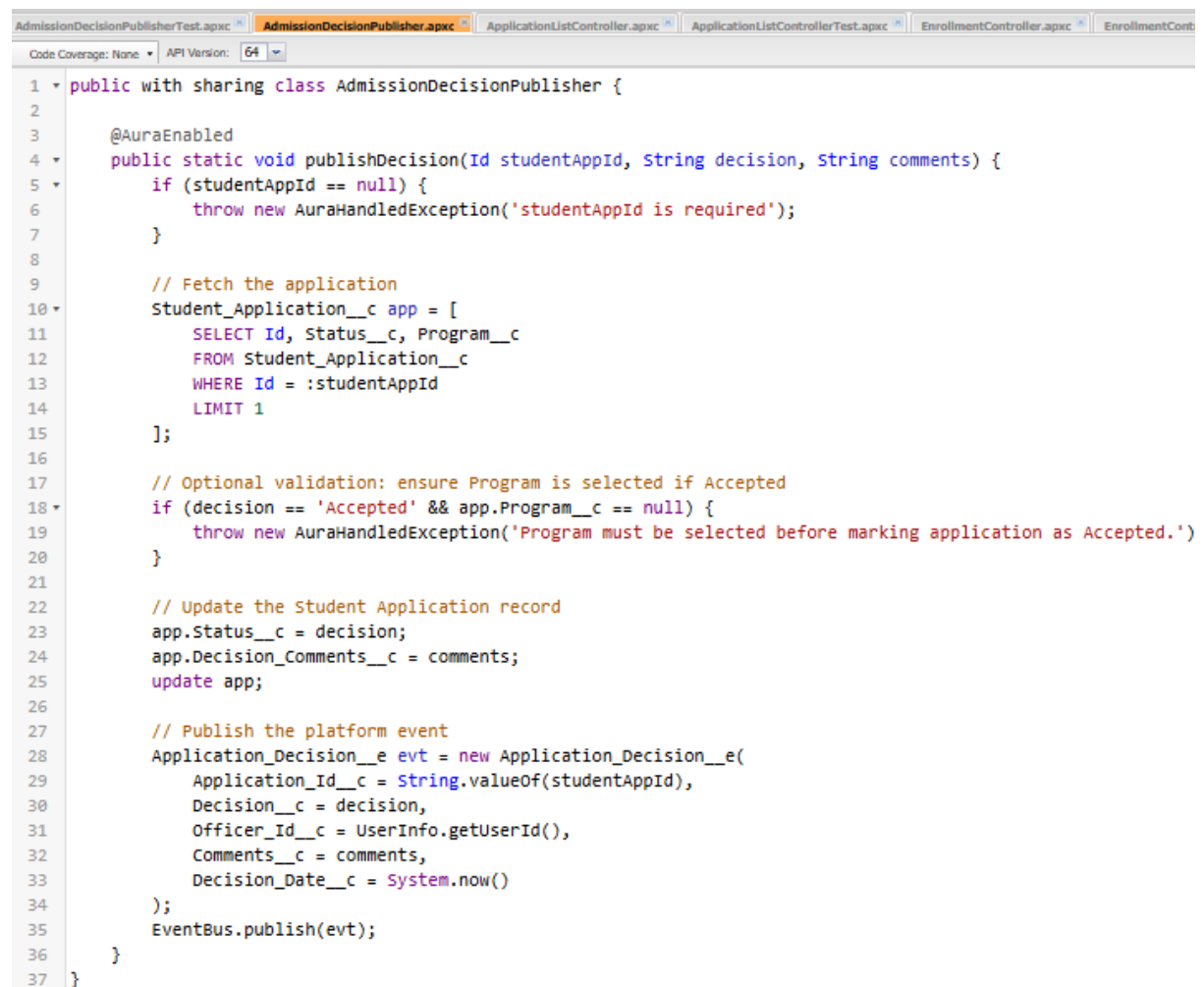
### Classes & Objects:

- **Classes** are blueprints for logic, containing methods and variables.

In this project, we created:

- **AdmissionDecisionPublisher** → Publishes admission decisions and updates student application records.

### Apex Code:



```
1 public with sharing class AdmissionDecisionPublisher {
2
3     @AuraEnabled
4     public static void publishDecision(Id studentAppId, String decision, String comments) {
5         if (studentAppId == null) {
6             throw new AuraHandledException('studentAppId is required');
7         }
8
9         // Fetch the application
10        Student_Application__c app = [
11            SELECT Id, Status__c, Program__c
12            FROM Student_Application__c
13            WHERE Id = :studentAppId
14            LIMIT 1
15        ];
16
17        // Optional validation: ensure Program is selected if Accepted
18        if (decision == 'Accepted' && app.Program__c == null) {
19            throw new AuraHandledException('Program must be selected before marking application as Accepted.');
20        }
21
22        // Update the Student Application record
23        app.Status__c = decision;
24        app.Decision_Comments__c = comments;
25        update app;
26
27        // Publish the platform event
28        Application_Decision__e evt = new Application_Decision__e(
29            Application_Id__c = String.valueOf(studentAppId),
30            Decision__c = decision,
31            Officer_Id__c = UserInfo.getUserId(),
32            Comments__c = comments,
33            Decision_Date__c = System.now()
34        );
35        EventBus.publish(evt);
36    }
37 }
```

### Test class for AdmissionDecisionPublisher:

```

AdmissionDecisionPublisherTest.apex | AdmissionDecisionPublisher.apex | ApplicationListController.apex | ApplicationListControllerTest.apex | Enrollme
Code Coverage: None | API Version: 64
1  @IsTest
2  public class AdmissionDecisionPublisherTest {
3
4      @IsTest
5      static void testPublishDecision_updatesRecordAndPublishesEvent() {
6          // Create Program
7          Program__c prog = new Program__c(Name='Test Program');
8          insert prog;
9
10         // Create Student Application
11         Student_Application__c app = new Student_Application__c(
12             Applicant_First_Name__c = 'Test',
13             Applicant_Last_Name__c = 'User',
14             Email__c = 'test@example.com',
15             Program__c = prog.Id,
16             Submitted_Date__c = System.now()
17         );
18         insert app;
19
20         Test.startTest();
21         AdmissionDecisionPublisher.publishDecision(
22             app.Id,
23             'Accepted',
24             'Good performance'
25         );
26         Test.stopTest();
27
28         Student_Application__c updatedApp = [
29             SELECT Id, Status__c, Decision_Comments__c
30             FROM Student_Application__c
31             WHERE Id = :app.Id
32         ];
33
34         System.assertEquals('Accepted', updatedApp.Status__c);
35         System.assertEquals('Good performance', updatedApp.Decision_Comments__c);
36     }
37 }

```

- **ApplicationListController** → Provides list view functionality for Student Applications in LWC.

## Apex Code:

```

onDecisionPublisherTest.apex | AdmissionDecisionPublisher.apex | ApplicationListController.apex | ApplicationListControllerTest.apex | EnrollmentController.apex | EnrollmentCont
Coverage: None | API Version: 64
public with sharing class ApplicationListController {

    public class PaginatedResult {
        @AuraEnabled public List<Student_Application__c> records;
        @AuraEnabled public Integer totalSize;
    }

    @AuraEnabled(cacheable=true)
    public static PaginatedResult getApplications(Integer pageSize, Integer pageNumber, String statusFilter) {
        if (pageSize == null || pageSize <= 0) pageSize = 10;
        if (pageNumber == null || pageNumber <= 0) pageNumber = 1;
        Integer offsetVal = (pageNumber - 1) * pageSize;

        List<Student_Application__c> apps;
        Integer total = 0;

        if (String.isBlank(statusFilter)) {
            apps = [
                SELECT Id, Applicant_First_Name__c, Applicant_Last_Name__c,
                       Email__c, Status__c, Submitted_Date__c
                FROM Student_Application__c
                ORDER BY Submitted_Date__c DESC
                LIMIT :pageSize OFFSET :offsetVal
            ];
            total = [SELECT COUNT() FROM Student_Application__c];
        } else {
            apps = [
                SELECT Id, Applicant_First_Name__c, Applicant_Last_Name__c,
                       Email__c, Status__c, Submitted_Date__c
                FROM Student_Application__c
                WHERE Status__c = :statusFilter
                ORDER BY Submitted_Date__c DESC
                LIMIT :pageSize OFFSET :offsetVal
            ];
            total = [SELECT COUNT() FROM Student_Application__c WHERE Status__c = :statusFilter];
        }

        PaginatedResult res = new PaginatedResult();
        res.records = apps;
        res.totalSize = total;
        return res;
    }
}

```

### Test class for ApplicationController:

```

ApplicationDecisionPublisherTest.apjx  AdmissionDecisionPublisher.apjx  ApplicationListController.apjx  ApplicationListControllerTest.apjx
de Coverage: None  API Version: 64

@Test
private class ApplicationListControllerTest {

    @Test static void testGetApplications_pagination() {
        List<Student_Application_c> apps = new List<Student_Application_c>();
        Program_c prog = new Program_c(Name='Test Program');
        insert prog;

        for (Integer i = 0; i < 12; i++) {
            apps.add(new Student_Application_c(
                Applicant_First_Name_c = 'Fn' + i,
                Applicant_Last_Name_c = 'Ln' + i,
                Email_c = 'test' + i + '@example.com',
                Program_c = prog.Id,
                Status_c = 'Submitted',
                Submitted_Date_c = System.now()
            ));
        }
        insert apps;

        Test.startTest();
        ApplicationListController.PaginatedResult res =
            ApplicationListController.getApplications(5, 1, null);
        Test.stopTest();

        System.assertNotEquals(null, res);
        System.assert(res.records.size() <= 5, 'Should return at most 5 apps');
        System.assert(res.totalSize >= 12, 'Total size should count all apps');
    }
}

```

- **EnrollmentController** → Manages enrolling accepted students into courses.

### Apex Code:

sonDecisionPublisherTest.apxc | AdmissionDecisionPublisher.apxc | ApplicationListController.apxc | ApplicationListControllerTest.apxc | EnrollmentController.apxc

Coverage: None | API Version: 64

```
public with sharing class EnrollmentController {
    @AuraEnabled
    public static List<Enrollment_Course__c> enrollCourses(Id studentId, List<Id> courseIds) {

        if (studentId == null || courseIds == null || courseIds.isEmpty()) {
            return new List<Enrollment_Course__c>();
        }


        // Get the Enrolled Student record to find its Application__c
        Enrolled_Student__c enrolled = [
            SELECT Id, Application__c
            FROM Enrolled_Student__c
            WHERE Id = :studentId
            LIMIT 1
        ];

        List<Enrollment_Course__c> inserts = new List<Enrollment_Course__c>();
        for (Id cId : courseIds) {
            inserts.add(new Enrollment_Course__c(
                Student__c = studentId,
                Course__c = cId,
                Application__c = enrolled.Application__c
            ));
        }
        insert inserts;
        return inserts;
    }

    @AuraEnabled(cacheable=true)
    public static List<Course__c> getCourses() {
        return [SELECT Id, Name FROM Course__c ORDER BY Name LIMIT 500];
    }
}
```



## Test class for EnrollmentController:



```
@isTest
public class EnrollmentControllerTest {

    @isTest
    static void testEnrollCourses_createsEnrollments() {
        // Step 1: Create Enrolled Student (Name auto-number, no value needed)
        Enrolled_Student__c student = new Enrolled_Student__c();
        insert student;

        // Step 2: Create Courses
        Course__c c1 = new Course__c(Name='Course 1');
        Course__c c2 = new Course__c(Name='Course 2');
        insert new List<Course__c>{c1, c2};

        // Step 3: Call the method under test
        Test.startTest();
        List<Enrollment_Course__c> enrollments = EnrollmentController.enrollCourses(
            student.Id,
            new List<Id>{c1.Id, c2.Id}
        );
        Test.stopTest();

        // Step 4: Verify enrollments were created
        System.assertEquals(2, enrollments.size(), 'Two enrollments should be created');
        for (Enrollment_Course__c ec : enrollments) {
            System.assertEquals(student.Id, ec.Student__c, 'Student ID should match');
        }
    }
}
```

Overall Code Coverage		
Class	Percent	Lines
Overall	94%	
AdmissionDecisionPublisher	100%	17/17
ApplicationListController	77%	14/18
EnrollmentController	100%	15/15

## Apex Triggers (before/after insert/update/delete)

- **Triggers** allow automation when records are created/updated.
- Example (not in my project at the moment, but possible extension):
  - A trigger on **Student\_Application\_\_c** can set Status\_\_c = 'Under Review' when a new record is created.
  - Another trigger can enforce validation (e.g., program must be selected before status = Accepted).

## Trigger Design Pattern

- To avoid logic directly in triggers, we use **Trigger Handler Classes**.

- Example: Instead of putting logic in a before update trigger for Student\_Application\_\_c, call methods in a separate handler class like ApplicationTriggerHandler.
- Benefits: Cleaner code, easier testing, reusability.

## SOQL & SOSL

- **SOQL (Salesforce Object Query Language):**
  - Used in controllers to fetch records.
  - Example in ApplicationListController and in other classes also.

```
List<Student_Application__c> apps;
Integer total = 0;

if (String.isBlank(statusFilter)) {
    apps = [
        SELECT Id, Applicant_First_Name__c, Applicant_Last_Name__c,
            Email__c, Status__c, Submitted_Date__c
        FROM Student_Application__c
        ORDER BY Submitted_Date__c DESC
        LIMIT :pageSize OFFSET :offsetVal
    ];
    total = [SELECT COUNT() FROM Student_Application__c];
} else {
    apps = [
        SELECT Id, Applicant_First_Name__c, Applicant_Last_Name__c,
            Email__c, Status__c, Submitted_Date__c
        FROM Student_Application__c
        WHERE Status__c = :statusFilter
        ORDER BY Submitted_Date__c DESC
        LIMIT :pageSize OFFSET :offsetVal
    ];
    total = [SELECT COUNT() FROM Student_Application__c WHERE Status__c = :statusFilter];
}
```

- **SOSL (Salesforce Object Search Language):**
  - Useful for searching across multiple objects at once.
  - Example: Searching applications by applicant name or email.

## Collections: List, Set, Map

- **List:** Ordered collection (used to store multiple application records).
- **Set:** Stores unique values (used to avoid duplicate student IDs).

- **Map:** Key-value pairs (e.g., mapping Course Id → Course Name).

In my project:

- **EnrollmentController** might use a `List<Enrollment_Course__c>` for bulk insert.
- **ApplicationListController** can use `Map<Id, Student_Application__c>` to optimize queries.

## Control Statements

- Apex supports if-else, for, while, and switch.
- Example from tests:

```
if (studentId == null || courseIds == null || courseIds.isEmpty()) {
    return new List<Enrollment_Course__c>();
}

for (Id cId : courseIds) {
    inserts.add(new Enrollment_Course__c(
        Student__c = studentId,
        Course__c = cId,
        Application__c = enrolled.Application__c
    ));
}
```

## Batch Apex

- Used for processing large data in chunks.
- Example extension for your project:
  - Batch job to update status of all applications pending for more than 30 days.

## Queueable Apex

- Supports async jobs with chaining.
- Example extension:
  - Queueable job to send bulk acceptance/rejection emails after decisions

## Scheduled Apex

- Run jobs at specific times.
- Example extension:
  - A scheduled job every Sunday to generate weekly admission statistics.

## Future Methods

- Used for async processing of callouts or heavy tasks.
- Example extension:
  - @future method to send an external notification (to an external system) when an application is accepted.

## Exception Handling

- Apex supports try-catch-finally.
- In AdmissionDecisionPublisher, if update fails due to missing program, an exception is thrown.

```
try {  
    update app;  
} catch(DmlException e) {  
    System.debug('Error while updating: ' + e.getMessage());  
}
```

## Test Classes

- Required to validate logic and ensure coverage.
- In your repo:
  - **AdmissionDecisionPublisherTest** → Tests decision publishing logic.
  - **ApplicationListControllerTest** → Tests pagination and retrieval of applications.
  - **EnrollmentControllerTest** → Tests course enrollment logic.

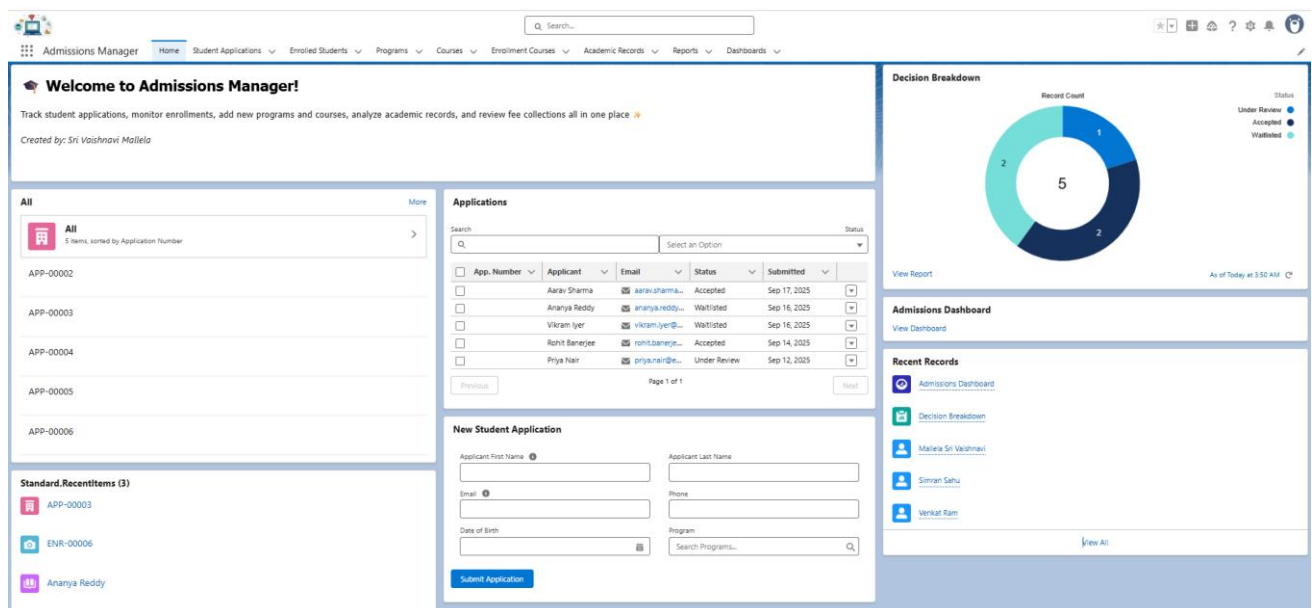
Salesforce requires **≥75% code coverage** for deployment to production.

## Phase 6 (User Interface Development):

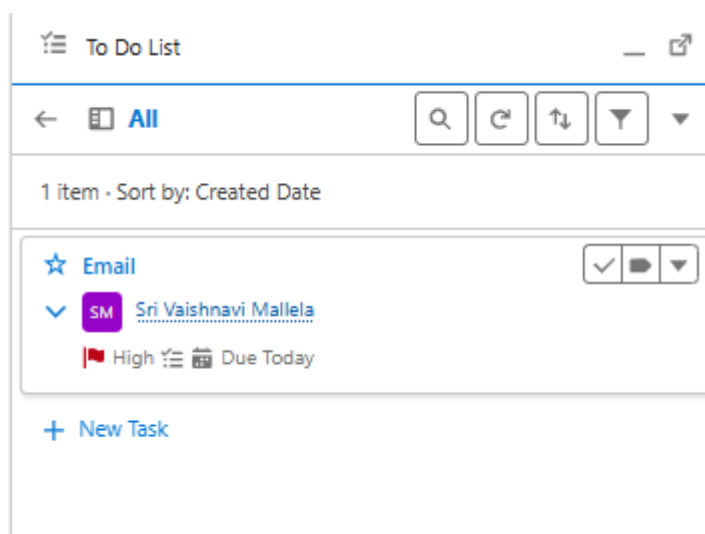
### Lightning App Builder

- Built a **custom Admission Management App** with navigation for Applications, Students, Courses, and Enrollment, Academic Records etc.
- Added **Record Pages, Home Pages, Tabs, and Utility Bar** using Lightning App Builder.

### Home Page Layout:



### Utility Bar – To Do List:



## Record Pages

- Created a **custom record page** for **Student Application** object.
- Components: Application details, related list (enrollment, decisions), and highlights panel.
- **Lightning record page of Student\_Application:**

**Application Details**

Applicant First Name	Vikram
Applicant Last Name	Iyer
Email	<a href="mailto:vikram.iyer@email.com">vikram.iyer@email.com</a>
Program	BEng Mechanical Engineering
Status	Waitlisted
Submitted Date	9/16/2025, 11:00 AM
Decision Comments	Application Decision

Upload Documents

[Upload Files](#) Or drop files

**Decision**

Decision

Select an Option

Comments

[Submit Decision](#)

- **Lightning record page of Enrolled\_Student:**

**Enroll Student**

Select courses to enroll the student in:

**Available Courses**

Available

- Business Strategy
- Data Structures
- Financial Management
- Fluid Mechanics
- Operating System
- Thermodynamics

**Selected**

[Enroll](#)

**Enrollment Details**

Enrollment Number: ENR-00006

Owner: [Mallela Sri Vaishnavi](#)

Application: [APP-00006](#)

Student Contact: [\[Edit\]](#)

Enrollment Status: [Active](#)

Start Date: 7/5/2025

End Date: 5/30/2028

Fees Paid: \$10,000.00

Payment Status: [Pending](#)

Name: [\[Edit\]](#)

Created By: [Mallela Sri Vaishnavi](#), 9/17/2025, 10:27 PM

Last Modified By: [Mallela Sri Vaishnavi](#), 9/17/2025, 10:27 PM

**Enrollment Courses (3)**

- [Business Strategy](#)
- [Fluid Mechanics](#)
- [Thermodynamics](#)

[View All](#)

Tabs:

. Student Applications tab

Admissions Manager

Home

Student Applications

Enrolled Students

Programs

Courses

Enrollment Courses

\* More

Student Applications

Recently Viewed

New

Import

5 items • Updated a few seconds ago

Star

Grid

Refresh

Print

Share

Filter

Submitted (0)

Under Review (1)

Accepted (2)

Rejected (0)

Waitlisted (2)

Enrolled (0)

APP-00005

APP-00006

APP-00002

APP-00003

APP-00004

Admissions Manager

Home

Student Applications

Enrolled Students

Programs

Courses

Enrollment Courses

More

Application Details

Decision

Applicant First Name

Priya

Applicant Last Name

Nair

Email

priya.nair@email.com

Program

MSc Data Science

Status

Under Review

Submitted Date

9/12/2025, 8:00 AM

Decision Comments

Upload Documents

Upload Files

Or drop files

Decision

Decision

Select an Option

Comments

Submit Decision

. Enrolled Students tab

Admissions Manager

Home

Student Applications

Enrolled Students

Programs

Courses

Enrollment Courses

\* More

Enrolled Students

Recently Viewed

New

Import

Change Owner

Assign Label

5 items • Updated a few seconds ago

Search this list...

Star

Grid

Refresh

Print

Share

Filter

Enrollment Number

1

2

3

4

5

ENR-00006

ENR-00005

ENR-00004

ENR-00003

ENR-00002

Admissions Manager

HomeStudent ApplicationsEnrolled StudentsProgramsCoursesEnrollment CoursesAcademic RecordsReportsDashboards

Enroll Student

Select courses to enroll the student in:

Available Courses

Available

Business StrategyData StructuresFinancial ManagementFluid MechanicsOperating SystemThermodynamics

Enroll

Selected

Enrollment Number

ENR-00006

Application

APP-00006

Student Contact

Enrollment Status

Active

Start Date

7/5/2025

End Date

5/30/2028

Fees Paid

\$10,000.00

Payment Status

Pending

Name

Created By

Mallela Sri Vaishnavi 9/17/2025, 10:27 PM

Last Modified By

Mallela Sri Vaishnavi 9/17/2025, 10:27 PM

Owner

Mallela Sri Vaishnavi

Enrollment Courses (3)

Business StrategyFluid MechanicsThermodynamics

View All

## . Programs tab

Admissions Manager

HomeStudent ApplicationsEnrolled StudentsProgramsCoursesEnrollment CoursesMore

Programs

Recently Viewed

NewImportChange OwnerAssign Label

5 items • Updated a few seconds ago

Program Name

1BA Economics

2BSc Computer Science

3MSc Data Science

4BEng Mechanical Engineering

5MBA Business Administration

## . Courses tab

Admissions Manager

HomeStudent ApplicationsEnrolled StudentsProgramsCoursesEnrollment CoursesMore

Courses

Recently Viewed

NewImportChange OwnerAssign Label

6 items • Updated a few seconds ago

Course Name

1Fluid Mechanics

2Thermodynamics

3Financial Management

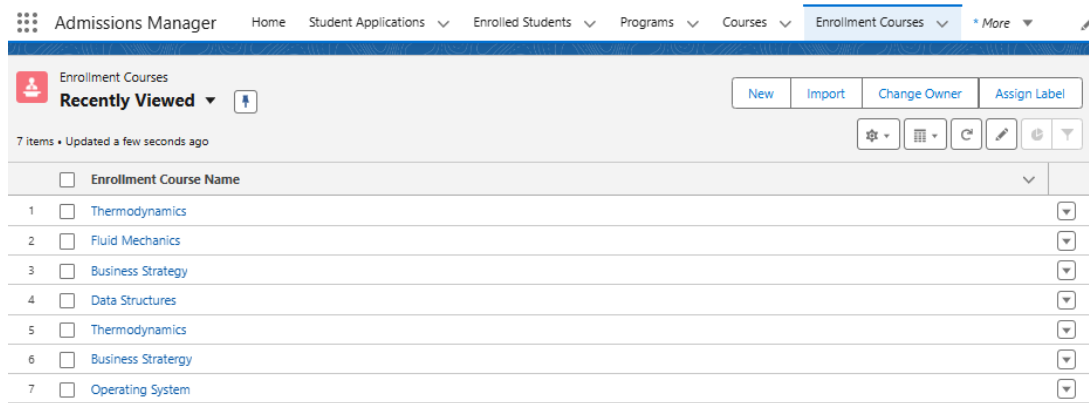
4Business Strategy

5Data Structures

6Operating System

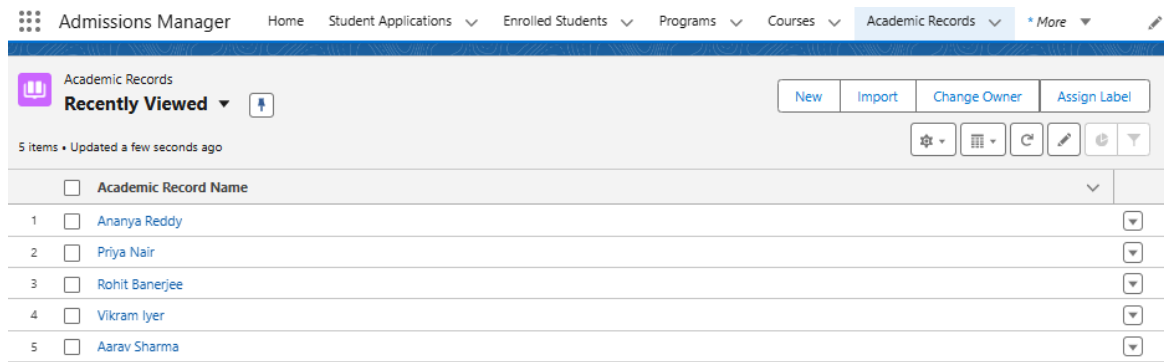


## . Enrollment Courses tab



<input type="checkbox"/>	Enrollment Course Name	
1 <input type="checkbox"/>	Thermodynamics	▼
2 <input type="checkbox"/>	Fluid Mechanics	▼
3 <input type="checkbox"/>	Business Strategy	▼
4 <input type="checkbox"/>	Data Structures	▼
5 <input type="checkbox"/>	Thermodynamics	▼
6 <input type="checkbox"/>	Business Strategy	▼
7 <input type="checkbox"/>	Operating System	▼

## . Academic Records tab



<input type="checkbox"/>	Academic Record Name	
1 <input type="checkbox"/>	Ananya Reddy	▼
2 <input type="checkbox"/>	Priya Nair	▼
3 <input type="checkbox"/>	Rohit Banerjee	▼
4 <input type="checkbox"/>	Vikram Iyer	▼
5 <input type="checkbox"/>	Aarav Sharma	▼

Related

Details

Files (0)

Add Files

Upload Files

Or drop files

## Apex with LWC

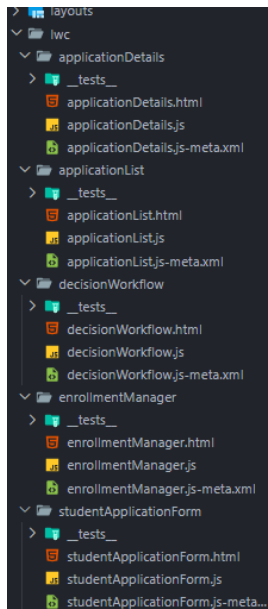
LWCs were integrated with **Apex controllers** to handle complex business logic and data operations that go beyond declarative tools.

### Examples in Project:

- **studentApplicationForm** → Calls Apex to insert new applications into Salesforce.
- **decisionWorkflow** → Invokes Apex to publish platform events and send notifications after a decision is made.
- **enrollmentManager** → Uses SOQL in Apex to fetch related enrollments and update fee statuses.

## Lightning Web Components (LWC)

In this project, **Lightning Web Components (LWCs)** were built to provide a modern, responsive, and user-friendly interface for managing the student admission lifecycle. LWCs leverage standard web technologies (HTML, JS, CSS) integrated with Salesforce data.



Action	Name ↑	Label	Type
Del	<a href="#">applicationDetails</a>	<a href="#">applicationDetails</a>	LWC
Del	<a href="#">applicationList</a>	<a href="#">applicationList</a>	LWC
Del	<a href="#">decisionWorkflow</a>	<a href="#">decisionWorkflow</a>	LWC
Del	<a href="#">enrollmentManager</a>	<a href="#">enrollmentManager</a>	LWC
Del	<a href="#">studentApplicationForm</a>	<a href="#">studentApplicationForm</a>	LWC

### Key LWCs Developed:

- **studentApplicationForm** → Allows applicants to submit personal, academic, and program details.

#### New Student Application

Applicant First Name ⓘ	Applicant Last Name
<input type="text"/>	<input type="text"/>
Email ⓘ	Phone
<input type="text"/>	<input type="text"/>
Date of Birth	Program
<input type="text"/>	<input type="text" value="Search Programs..."/>
<input type="button" value="Submit Application"/>	

- **applicationDetails** → Shows detailed application records for review.

#### Application Details

Upload Documents

Or drop files

- **decisionWorkflow** → Enables admission officers to accept, reject, or waitlist applications.

**Decision**

Decision

Select an Option

Comments

Submit Decision

- **applicationList** → Displays submitted applications with filtering and sorting for officers.

**Applications**

Search

Status

Select an Option

<input type="checkbox"/>	App. Number	Applicant	Email	Status	Submitted	
<input type="checkbox"/>		Aarav Sharma	<a href="#">aarav.sharma...</a>	Accepted	Sep 17, 2025	<input type="checkbox"/>
<input type="checkbox"/>		Ananya Reddy	<a href="#">ananya.reddy...</a>	Waitlisted	Sep 16, 2025	<input type="checkbox"/>
<input type="checkbox"/>		Vikram Iyer	<a href="#">vikram.iyer@...</a>	Waitlisted	Sep 16, 2025	<input type="checkbox"/>
<input type="checkbox"/>		Rohit Banerjee	<a href="#">rohit.banerje...</a>	Accepted	Sep 14, 2025	<input type="checkbox"/>
<input type="checkbox"/>		Priya Nair	<a href="#">priya.nair@e...</a>	Under Review	Sep 12, 2025	<input type="checkbox"/>

Previous Page 1 of 1 Next

- **enrollmentManager** → Helps track enrollments, fee status, and course allocations.

**Enroll Student**

Select courses to enroll the student in:

Available Courses

Available

Business Strategy

Data Structures

Financial Management

Fluid Mechanics

Operating System

Thermodynamics

Selected

Enroll

Reports:

Reports

Recent

7 items

Q

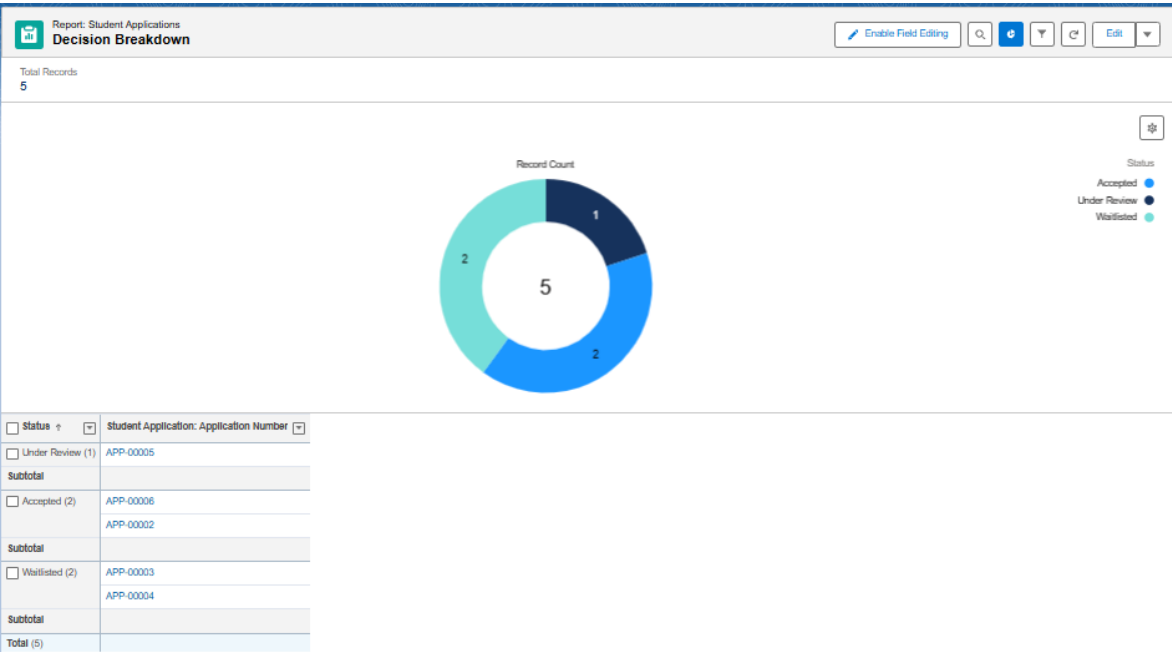
Search recent reports...

New Report

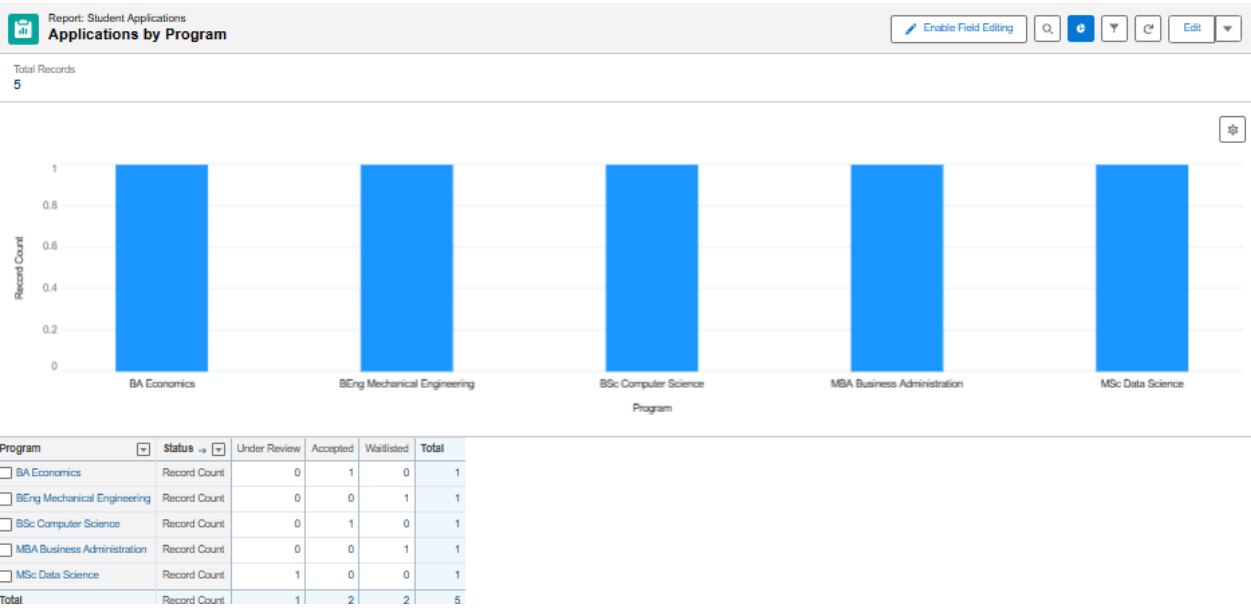
New Folder

REPORTS	Report Name	Description	Folder	Created By	Created On	Subscribed
Recent	Decision Breakdown		Admissions Reports	Mallela Sri Vaishnavi	9/17/2025, 11:39 PM	
Created by Me	Applications by Program		Admissions Reports	Mallela Sri Vaishnavi	9/17/2025, 11:30 PM	
	Enrollments per Course		Admissions Reports	Mallela Sri Vaishnavi	9/17/2025, 11:49 PM	
Private Reports						
Public Reports	Sample Flow Report: Screen Flows	Which flows run, what's the status of each interview, and how long do users take to complete the screens?	Public Reports	Automated Process	9/9/2025, 2:04 PM	
All Reports						
FOLDERS	Total Fees Collected		Admissions Reports	Mallela Sri Vaishnavi	9/18/2025, 1:25 AM	
	GPA Distribution Report		Admissions Reports	Mallela Sri Vaishnavi	9/18/2025, 1:19 AM	
All Folders	Total Applicaitons		Admissions Reports	Mallela Sri Vaishnavi	9/17/2025, 11:33 PM	
Created by Me						

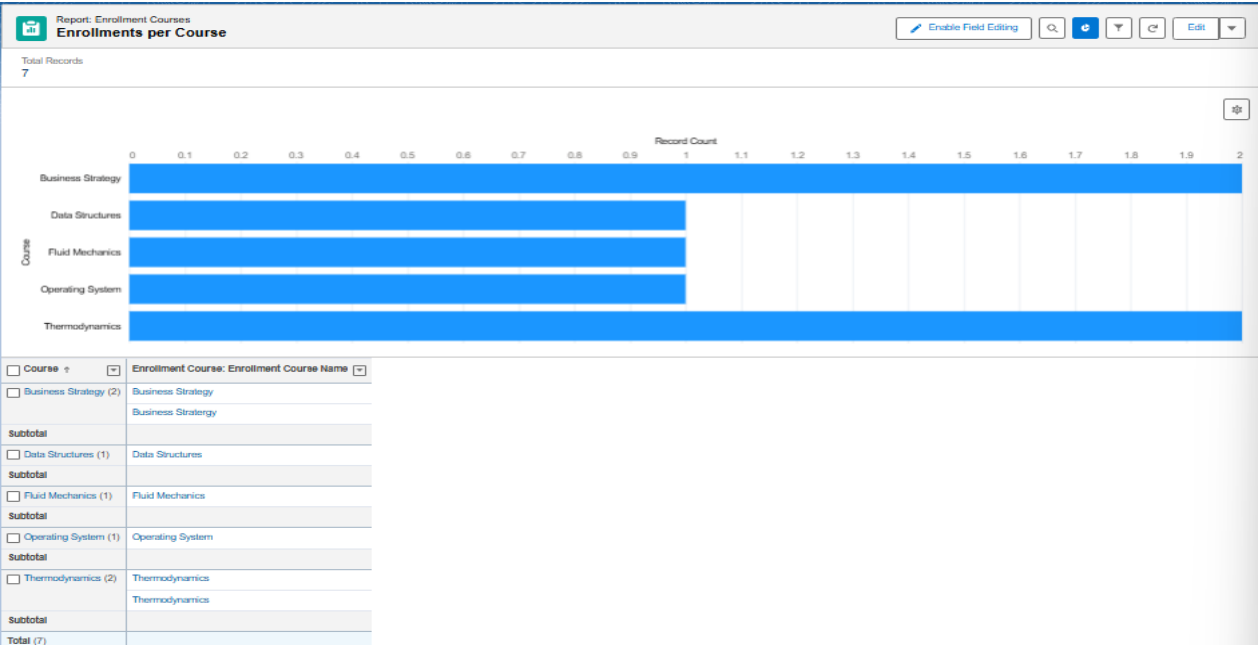
Decision Breakdown report:



Applications by Program report:



Enrollments per course report:



Total fees collected report:

Report: Enrolled Students

Total Fees Collected

Total Records

5

Total Fees Paid

\$265,000.00

	Enrolled Student: Enrollment Number	Fees Paid
1	ENR-00002	\$25,000.00
2	ENR-00003	\$60,000.00
3	ENR-00004	\$20,000.00
4	ENR-00005	\$150,000.00
5	ENR-00006	\$10,000.00
6		\$265,000.00

GPA Distribution report:

Report: Academic Records

GPA Distribution Report

Total Records

5

Total GPA

18.20

	GPA	Academic Record: Academic Record Name	GPA Range
1	3.40	Vikram Iyer	Bucket 4
2	3.50	Rohit Banerjee	Bucket 4
3	3.90	Ananya Reddy	Bucket 3
4	3.60	Aarav Sharma	Bucket 3
5	3.80	Priya Nair	Bucket 3
6	18.20		

## **Phase 7 (Integration & External Access) –**

### **Not integrated in Project**

#### **Named Credentials**

Named Credentials provide a secure and simplified way to call external APIs without storing authentication details in Apex code. They bundle the endpoint URL and authentication in one place, so developers only need to reference them in callouts.

- **Usage:** Used for integrating with third-party services such as payment gateway APIs to verify student fee payments.
- **Key Benefit:** Centralized management of authentication and endpoints, reducing hardcoding and increasing security.

#### **External Services**

External Services allow Salesforce to connect with APIs described by an OpenAPI/Swagger specification and automatically generate Apex actions or Flow actions. This enables both developers and admins to leverage APIs

- **Usage:** Used to integrate with external services such as document verification APIs (for verifying student identity, certificates, or test scores).
- **Key Benefit:** Exposes external system functionality inside Salesforce, which can then be directly used in Flows to automate admission processes.

#### **Web Services (REST/SOAP)**

Salesforce supports consuming and exposing web services using REST and SOAP protocols. REST is commonly used for lightweight and modern integrations, while SOAP is still supported for legacy systems.

- **Usage:** REST services can be used for sending student admission data to external analytics platforms. SOAP services can be used for exchanging admission decisions with traditional ERP systems.

- **Key Benefit:** Enables Salesforce to work seamlessly with both modern and legacy applications.

## Callouts

Apex callouts allow Salesforce to send HTTP requests to external services and process responses. They can be synchronous (real-time) or asynchronous.

- **Usage:** Used for confirming online payment transactions, checking scholarship eligibility, or fetching course details from external academic systems.
- **Key Benefit:** Enables Salesforce to stay connected with external data sources, ensuring accurate information during admission workflows.

## Platform Events

Platform Events enable an event-driven architecture by allowing apps to publish and subscribe to custom events.

- **Usage:** When an admission decision (Accept, Reject, Waitlist) is made, a Platform Event can notify the student portal and trigger external notification services like SMS or email systems.
- **Key Benefit:** Provides real-time, asynchronous communication across Salesforce and external systems.

## Change Data Capture (CDC)

Change Data Capture publishes real-time events whenever records in Salesforce are created, updated, deleted, or undeleted. External systems can subscribe to these events.

- **Usage:** When a student updates their contact details or when an application status changes, the external systems like the university's master student database are updated immediately.
- **Key Benefit:** Ensures synchronization of data across Salesforce and external systems without manual intervention.

## Salesforce Connect

Salesforce Connect allows Salesforce to access external data sources in real-time without importing or storing the data in Salesforce.

- **Usage:** Used to access a student's past academic history from the university's external database without duplicating records in Salesforce.
- **Key Benefit:** Reduces storage costs, avoids redundancy, and provides up-to-date external data on demand.

## API Limits

Salesforce enforces API request limits based on the edition and user licenses to maintain performance and system stability.

- **Usage:** The Student Admission CRM must carefully monitor API usage when performing frequent callouts for payment verification, document checks, and integration with third-party apps.
- **Key Benefit:** Helps manage integrations efficiently while staying within Salesforce's governor limits.

## OAuth & Authentication

OAuth 2.0 is a secure, token-based authentication mechanism used for connecting Salesforce with external apps. It allows apps to access Salesforce data without exposing user credentials.

- **Usage:** Used when connecting Salesforce with student self-service portals, learning management systems (LMS)
- **Key Benefit:** Provides secure, scalable, and industry-standard authentication for integrated apps.

## Remote Site Settings

Remote Site Settings whitelist external endpoints so Salesforce can send callouts to them. This was the traditional approach before Named Credentials.

- **Usage:** Used for integrations with legacy services such as SMS gateways or APIs that do not require advanced authentication.
- **Key Benefit:** Protects Salesforce from sending data to unauthorized or untrusted endpoints.



## Phase 8 (Data Management & Deployment)

### Data Import Wizard

- Browser-based tool to import up to 50,000 records.
- Supports standard/custom objects with field mapping and duplicate prevention.
- Used by Admissions Manager to upload new applications, academic records, or program/course details received offline.

### Data Loader

- Desktop client to handle large volumes of records (millions).
- Supports insert, update, upsert, delete, and export.
- Used for migrating historical applications, bulk updating fee statuses, or exporting enrollment data for audits.

### Duplicate Rules

- Works with Matching Rules to prevent duplicate records.
- Can block duplicates or allow with alerts.
- Ensures no duplicate student records are created when the same applicant reapplies with the same email or ID.

All Duplicate Rules

What Are Duplicate Rules?

View: All Duplicate Rules ▾

A   B   C   D   E   F   G   H   I   J   K   L   M   N   O   P   Q   R					
New Rule ▾					
Rule Name ↑	Description	Object	Matching Rule	Active	Last Modified By
<a href="#">Standard Account Duplicate Rule</a>	Identify accounts that duplicate other accounts.	Account	<a href="#">Standard Account Matching Rule</a>	✓	OEPIG
<a href="#">Standard Contact Duplicate Rule</a>	Identify contacts that duplicate other contacts and leads.	Contact	<a href="#">Standard Contact Matching Rule</a> <a href="#">Standard Lead Matching Rule</a>	✓	OEPIG
<a href="#">Standard Lead Duplicate Rule</a>	Identify leads that duplicate other leads and contacts.	Lead	<a href="#">Standard Lead Matching Rule</a> <a href="#">Standard Contact Matching Rule</a>	✓	OEPIG

### Data Export & Backup

- Provides manual and scheduled export of Salesforce data.
- Exports records into CSV files bundled in .zip.
- Weekly/monthly backups ensure secure storage of student applications, enrollments, and course allocations for compliance.

## Monthly Export Service

Data Export lets you prepare a copy of all your data in salesforce.com. From this page you can start the export process manually or schedule it to run automatically. When an export is ready for download you will receive an email containing a link that allows files are also available on this page for 48 hours, after which time they are deleted.

Next scheduled export: None		
<a href="#">Export Now</a> <a href="#">Schedule Export</a>		
Scheduled By	Mallela Sri Vaishnavi	
Schedule Date	9/21/2025	
Export File Encoding	ISO-8859-1 (General US & Western European, ISO-LATIN-1)	
Action	File Name	File Size
<a href="#">download</a>	WE_00DgL00000BGZ9xUAH_1.ZIP	3.1M

## Change Sets

- Moves metadata (not data) between environments like Sandbox → Production.
- Ensures tested configurations are promoted safely.
- Used to deploy workflows, approval processes, and Lightning Pages like Application Details from Sandbox to Production.

## Unmanaged vs Managed Packages

- *Unmanaged*: Customizable after installation, used internally.
- *Managed*: Locked, used for distributing polished apps (AppExchange).
- Unmanaged Package used to share Admission CRM internally; Managed Package if released commercially as an educational solution.

## ANT Migration Tool

- Java/command-line tool to script deployments.
- Used to automate deployment of LWCs, triggers, and Apex classes (e.g., decisionWorkflow) from Sandbox to Production.

## VS Code & SFDX

- Modern development environment for Salesforce.
- SFDX enables scratch orgs, push/pull metadata, testing, and deployments.
- Used to build and deploy LWCs like studentApplicationForm, applicationList, and enrollmentManager with version control integration.

## **Phase 9 (Reporting, Dashboards & Security Review)**