

Library Management Phase 10: Final Presentation & Demo Day Documentation

Objective:

Demonstrate the end-to-end functionality of the Library Management , highlight outcomes, and provide documentation for teachers, students, librarians, and library admin.

1. Project Pitch

Goal: Showcase how Smart Library CRM simplifies resource tracking, improves student–teacher interaction, and increases library efficiency.

Key Points to Present:

1. Streamlined student/teacher record management.
 2. Automated book issue/return tracking & fine calculation.
 3. Real-time updates for teachers, students, and librarians.
 4. Analytics through reports and dashboards (usage trends, overdue items, popular books).
-

2. Demo Flow (Click-by-Click Navigation)

Step 1 – Login & Access App

- Open Salesforce → Go to Library Management Lightning App.

Step 2 – Add/Manage Student or Teacher

- Navigate to Students or Teachers Tab → New.
- Fill in details (Name, DOB, Contact, Class/Department).
- Save → System applies any automation (e.g., auto-assign class or default library card number).

Step 3 – Book Selection / Issue

- Navigate to Books Tab → Select Book.
- Create Borrow Request by selecting book, issue date, and due date.

Step 4 – Return & Fine Handling

- Navigate to Borrow Requests Tab → Mark as Returned.

- System auto-calculates Fine Amount (if overdue).
- Librarian records payment of fine if applicable.

Step 5 – Notifications & Feedback

- Confirm book issue/return → Student receives notification (demo simulated without actual emails).
- Navigate to Feedback Tab → New Feedback to record user feedback on library services.

Step 6 – Dashboard Update

- Navigate to Dashboards → Library Overview Dashboard.
 - Verify that new borrow requests, returns, fines, and feedback appear correctly.
-

PHASE 1 Project Title: Library Management

Problem Statement

Managing and tracking books can become cumbersome when done manually or on paper. Small libraries, students, or hobbyists often need a lightweight system to record book details, view them quickly, and make edits without installing heavy software or maintaining a server. The Simple Library Management website addresses this by offering a browser-based CRUD tool using only HTML, CSS, and JavaScript with localStorage.

Requirement Gathering

- Identify what core actions the user needs (add, view, edit, delete books).
- Decide which fields are mandatory (Title, Author, Year, ISBN).
- Determine storage mechanism (localStorage vs. server).
- Collect UI expectations (form + table layout, buttons).
- Plan file structure (index.html, styles.css, script.js).
- Document non-functional needs (works offline, simple design).

Stakeholder Analysis

- Define primary users (students, small libraries, demo users).
- Identify secondary users (developers learning CRUD apps).
- Note stakeholders' technical skills (beginner-friendly, no server).
- Capture feedback on desired features (extra fields, pop-ups).
- Understand constraints (browser-only, small data sets).
- Set expectations for future enhancements (server API, multi-user).

Business Process Mapping

- Map the flow of adding a book (form input → JS → localStorage → table).
- Describe how data persists after refresh (JSON in localStorage).
- Show the edit flow (click edit → prompt/update → save back).
- Map the delete flow (click delete → confirm → remove from storage).
- Define initial data state (empty table).
- Document error handling or confirmations (validation, clearing storage).

Industry-Specific Use Case Analysis

- Compare with small-scale library or personal book lists.
- Explore educational uses (teaching CRUD in classes).
- Consider adaptation for inventory or asset tracking.
- Assess scalability limits (localStorage size, single user).
- Identify privacy and data safety considerations (local only).
- Evaluate user interface needs across devices (desktop, mobile).

AppExchange / Expansion Exploration

- Plan how to extend fields (Genre, Publisher, Ratings).
 - Replace prompts with inline edit forms or modals.
 - Investigate moving from localStorage to a simple backend or API.
 - Consider packaging as a small open-source app or template.
 - Experiment with new UI styles (Tailwind, Bootstrap, animations).
 - Explore publishing on platforms like Salesforce AppExchange or GitHub Pages as a free tool.
-

PHASE 2: Org Setup & Configuration

1. Introduction

This phase sets up a Salesforce Developer Org to model a small library for cataloguing, lending, and returns. It builds on Phase 1 by configuring book records, librarian/member roles, and security so the system reflects real library workflows for adding, editing, deleting, and viewing books.

2. Objective

Create a clean Salesforce Developer Org named “Library Connect” with sample users (Head Librarian, Librarian, Member).

Use OWD and sharing rules to protect book data.

Showcase book entry, editing, and role-based visibility, and lay the groundwork for future fields and external integrations.

Step 1:- • Salesforce Editions Sign up & Login

Signed up for a Salesforce Developer Edition and logged into the Lightning Experience. Confirmed access to the Setup area sing the gear icon.

Seller Home Good afternoon, Manjeet. Let's get selling!

Close Deals
Opportunities owned by me and closing this quarter

- \$0 Total Pipeline
- 0 Open
- 0 Won
- 0 Lost

Plan My Accounts
Accounts owned by me

- 8 Accounts
- 0 Upcoming Activity
- 0 Past Activity
- 8 No Activity

Grow Relationships
Contacts owned by me and created in the last 90 days

- 6 Contacts
- 0 Upcoming Activity
- 0 Past Activity
- 6 No Activity

Build Pipeline
Leads owned by me and created in the last 30 days

- 0 Leads
- 0 Upcoming Activity
- 0 Past Activity
- 0 No Activity

My Goals
Set personal weekly or monthly goals for emails, calls, and meetings.

Today's Events

Today's Tasks

Recent Records

- Manjeet Urmaliya
- Joined report - Acc cont lead
- Exercise Completion by Days to Complete

To Do List

Step 2:- • Company Profile Setup

Company Information

Company Information
Gyan Ganga College Of Technology

The organization's profile is below.

Organization Detail

Organization Name	Gyan Ganga College Of Technology	Phone	
Primary Contact	OrgFarm EPIC	Fax	
Division		Default Locale	English (United States)
Address	United States	Default Language	English
Fiscal Year Starts In	January	Default Time Zone	(GMT-07:00) Pacific Daylight Time (America/Los_Angeles)
Activate Multiple Currencies	<input type="checkbox"/>	Currency Locale	English (United States) - USD
Enable Data Translation	<input type="checkbox"/>	Used Data Space	412 KB (8%) [View]
Newsletter	<input checked="" type="checkbox"/>	Used File Space	17 KB (0%) [View]
Admin Newsletter	<input checked="" type="checkbox"/>	API Requests, Last 24 Hours	0 (15,000 max)
Hide Notices About System Maintenance	<input type="checkbox"/>	Streaming API Events, Last 24 Hours	0 (10,000 max)
Hide Notices About System Downtime	<input type="checkbox"/>	Restricted Logins, Current Month	0 (0 max)
Locale Formats	ICU	Salesforce.com Organization ID	00DGJ000007XhYP
		Organization Edition	Developer Edition
		Instance	CAN98

Created By OrgFarm EPIC 7/18/2025, 8:45 PM Modified By Manjeet Urmaliya 8/12/2025, 8:08 AM

User Licenses

Step 3:- • Business Hours & Holidays

Created " Library Management Hours" Mon-Sat 10 AM-5 PM.

Added relevant Holidays and linked them to Business Hours.

Business Hours Detail

Business Hours Name	Library Management Hours	Time Zone	Default Business Hours														
Business Hours	<table border="1"> <tr><td>Sunday</td><td>No Hours</td></tr> <tr><td>Monday</td><td>10:00 AM to 5:00 PM</td></tr> <tr><td>Tuesday</td><td>10:00 AM to 5:00 PM</td></tr> <tr><td>Wednesday</td><td>10:00 AM to 5:00 PM</td></tr> <tr><td>Thursday</td><td>10:00 AM to 5:00 PM</td></tr> <tr><td>Friday</td><td>10:00 AM to 5:00 PM</td></tr> <tr><td>Saturday</td><td>10:00 AM to 5:00 PM</td></tr> </table>	Sunday	No Hours	Monday	10:00 AM to 5:00 PM	Tuesday	10:00 AM to 5:00 PM	Wednesday	10:00 AM to 5:00 PM	Thursday	10:00 AM to 5:00 PM	Friday	10:00 AM to 5:00 PM	Saturday	10:00 AM to 5:00 PM	(GMT+05:30) India Standard Time (Asia/Kolkata)	✓
Sunday	No Hours																
Monday	10:00 AM to 5:00 PM																
Tuesday	10:00 AM to 5:00 PM																
Wednesday	10:00 AM to 5:00 PM																
Thursday	10:00 AM to 5:00 PM																
Friday	10:00 AM to 5:00 PM																
Saturday	10:00 AM to 5:00 PM																

Created By: Manjeet.Urmalya 9/20/2025, 3:56 AM Last Modified By: Manjeet.Urmalya 9/20/2025, 3:56 AM

Holidays

Holiday Name	Description	Date and Time
Dwali		10/18/2025 All Day
Holi		3/7/2025 All Day

Step 4:- • User Setup & Licenses

- Created 2 sample users:

Display Name	Username (example)	Profile	Role
Librarian Admin	admin_librarian@library.com	Test Profile	Test Role
Teacher Librarian	teacher.librarian@library.com	Identity User	SVP, Human Resources

All Users

Action	Full Name	Alias	Username	Role	Active	Profile
<input type="checkbox"/>	Alay	alay	alay123@test.com	COO	✓	Test profile
<input type="checkbox"/>	Chatter Expert	Chatter	chatty_00000000000000000000000000000000@chatter.salesforce.com		✓	Chatter Free User
<input type="checkbox"/>	EPIC_OrgFarm	GEPIC	epic_c7097595f40@orgfarm.salesforce.com		✓	System Administrator
<input type="checkbox"/>	Librarian Admin	admin	admin_librarian@library.com	Test Role	✓	Test profile
<input type="checkbox"/>	Teacher Librarian	admin	teacher.librarian@library.com	SVP_Human Resources	✓	Identity User
<input type="checkbox"/>	Urmalya_Manjeet	cse	cse22_manjeeturnam@agentforce.com		✓	System Administrator
<input type="checkbox"/>	User_Integration	integ	integrations@00d900000000000000@xyphus0.com		✓	Analytics Cloud Integration User
<input type="checkbox"/>	User_Security	sec	insightssecurity@00dg00000000000000@xyphus0.com		✓	Analytics Cloud Security User

Step 5:- Profiles

Cloned Standard User → created Librarian profile.

Cloned Standard User → created Teacher profile.

Adjusted Object Permissions:

Mentor_c: Doctor = Read/Create/Edit/Delete = Read/Create/Edit/Delete

student_c: Doctor = Read/Edit/Create/Delete = Read/Edit/Create

The screenshot shows the Salesforce Setup interface with the 'Profiles' tab selected under 'Object Manager'. A search bar at the top contains 'profile'. The main area displays the 'Librarian' profile details. The profile has the following settings:

- Name:** Librarian
- User License:** Salesforce
- Description:** SalesForce
- Created By:** Manjeet Urmaliya, 9/20/2025, 5:08 AM
- Modified By:** Manjeet Urmaliya, 9/20/2025, 5:14 AM

Under the 'Custom Profile' section, there is a checked checkbox.

Page Layouts section:

Standard Object Layouts	Global	Location Group Assignment	Location Group Assignment Layout
Email Application	Global Layout [View Assignment]	Macro	Macro Layout [View Assignment]
Home Page Layout	Home Page Default [View Assignment]	Object Milestone	Object Milestone Layout [View Assignment]
Account	Account Layout [View Assignment]	Operating Hours	Operating Hours Layout [View Assignment]
Alternative Payment Method	Alternative Payment Method Layout	Opportunity	Opportunity Layout [View Assignment]
Appointment Invitation	Appointment Invitation Layout [View Assignment]	Opportunity Product	Opportunity Product Layout [View Assignment]
Asset	Asset Layout [View Assignment]	Order	Order Layout [View Assignment]
Asset Action	Asset Action Layout	Order Product	Order Product Layout

Librarian profile

The screenshot shows the Salesforce Setup interface with the 'Profiles' page selected. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar at the top right says 'Search Setup'. The main content area has a header 'Profiles' with a user icon. Below it, there are sections for 'Data Shareflake Connections', 'Work Step Templates', 'Work Types', and 'Work Type Groups', each with a grid of checkboxes. A 'Custom Object Permissions' section follows, with two tabs: 'Basic Access' and 'Data Administration'. Under 'Basic Access', rows are shown for 'Mentors' and 'Students' with columns for Read, Create, Edit, Delete, View All Records, Modify All Records, and View All Fields. A 'Session Settings' section shows 'Session Times Out After' set to '2 hours of inactivity' and 'Session Security Level Required at Login'. A 'Password Policies' section lists various password requirements. At the bottom are 'Edit', 'Clone', 'Delete', and 'View Users' buttons.

Teacher profile

Step 6 — Role Hierarchy

Created simple hierarchy

- Librarian Admin
 - Librarian
 - Teacher

The screenshot shows the Salesforce Setup interface with the 'Roles' page selected. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar at the top right says 'Search Setup'. The main content area has a header 'Roles' with a user icon. Below it, a section titled 'Creating the Role Hierarchy' with a sub-section 'Your Organization's Role Hierarchy'. A tree view shows the following hierarchy under 'Gyan Ganga College Of Technology':

- CEO
- CFO
- COO
- Librarian Admin
- SVP, Customer Service & Support
- SVP, Human Resources
- SVP, Sales & Marketing
- Teacher Librarian

 Each role node has 'Edit | Del | Assign' buttons. A 'Help for this Page' link and a 'Show in tree view' dropdown are also present.

Step 7 Org:- Wide Defaults (OWD)

Setup → Sharing Settings

Mentor_c = Private.

Student_c = Controlled by Parent .

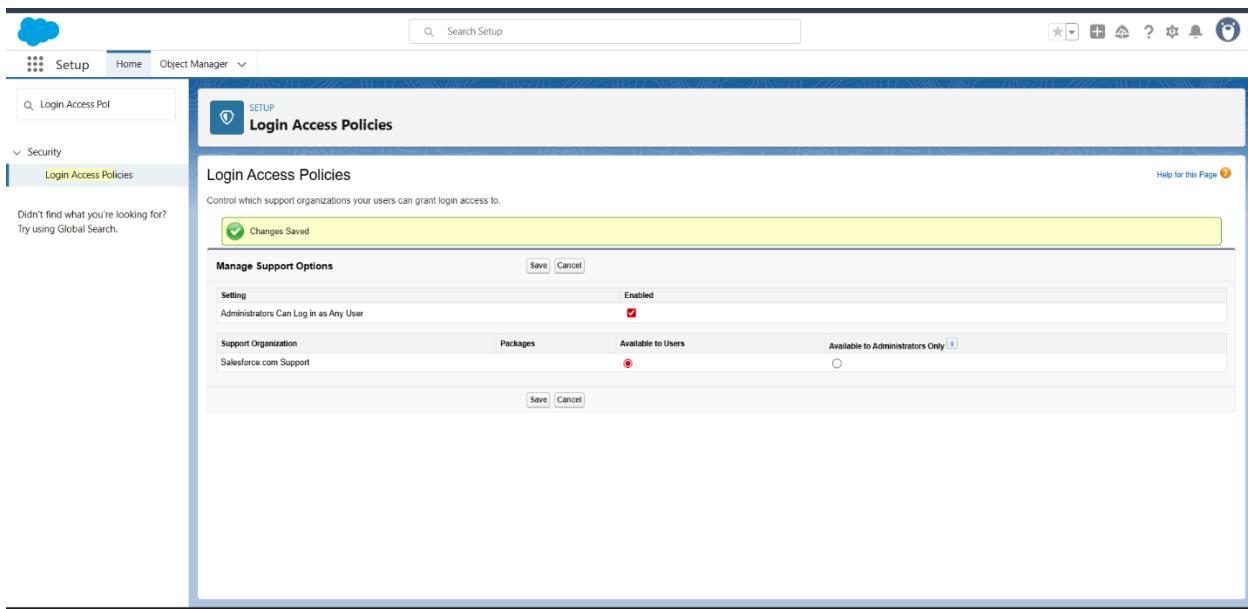
The screenshot shows the 'Sharing Settings' page in the Salesforce Setup. The 'Work Order' object is selected, and its sharing rules are displayed. The 'Mentor' object is set to 'Private', while 'Student' is set to 'Controlled by Parent'. Other settings like Manager Groups and Secure guest user record access are also visible.

Step 8:- • Sharing Rules

The screenshot shows the 'Sharing Settings' page in the Salesforce Setup. It lists sharing rules for various objects: Work Order Sharing Rules, Work Plan Sharing Rules, Work Plan Template Sharing Rules, Work Step Template Sharing Rules, Work Type Sharing Rules, Work Type Group Sharing Rules, and Mentor Sharing Rules. The Mentor Sharing Rules section shows a rule for 'Mentor: M_Email NOT EQUAL TO' with 'Role: Test Role' having 'Read Only' access and 'Role: Librarian Admin' having 'ReadWrite' access.

Step 9:- Login Access Policy

- Enabled Administrators Can Log in as Any User.



Step 10:- Library Management Lightning App.

- Setup → App Manager → New Lightning App “Library Management”

Accounts									
All Accounts									
21 items • Sorted by Account Name • Updated a minute ago									
	Account Name ↑		Accou... ↓	Billing State/Province ↓	Phone ↓	Type ↓		Account Owner AI... ↓	
1	Ajay			North Carolina	(09703049276)	Customer - Direct	cse	OEPIC	
2	Burlington Textiles Corp of America			Kansas	(336) 222-7000	Customer - Channel	OEPIC		
3	Dickenson plc			Texas	(785) 241-6200	Customer - Direct	OEPIC		
4	Edge Communications			Oregon	(512) 757-6000	Customer - Channel	OEPIC		
5	Express Logistics and Transport			California	(503) 421-7800	Customer - Direct	OEPIC		
6	GenePoint			Illinois	(650) 867-3450	Customer - Channel	OEPIC		
7	Grand Hotels & Resorts Ltd				(312) 596-1000	Customer - Direct	OEPIC		
8	hlo					cse			
9	Pyramid Construction Inc.				(014) 427-4427	Customer - Channel	OEPIC		
10	Ram					cse			
11	Sample Account for Entitlements					autopro			
12	sForce					OEPIC			
13	Test Approval Process					cse			
14	Test Condition formating					cse			
15	Test Report					cse			
16	TestApproval2					cse			
17	TestApproval3					cse			
18	United Oil & Gas Corp.			New York	(212) 842-5500	Customer - Direct	OEPIC		
19	United Oil & Gas, Singapore				(650) 450-8810	Customer - Direct	OEPIC		

Step 11:- Testing

The screenshot shows a Salesforce account detail page for 'Ajay'. The top navigation bar includes links for Home, Calendar, Contacts, Students, Mentors, Accounts (selected), Change Requests, Custom Libraries, and Dashboards. The main content area has tabs for 'Related' and 'Details'. Under 'Details', there are fields for Account Owner (Manjeet Urmaliya), Account Name (Ajay), Parent Account, Account Number, Account Site, Type, Industry, Account Source, Annual Revenue, and Rating. To the right, there's a 'Chatter' section titled 'Activity' with a 'Upcoming & Overdue' feed showing no activities.

PHASE 3: Data Modeling & Relationships

Introduction

Define and implement the data structure of Library Connect to link Members, Book Loans and Books. Configure record types, page layouts, navigation tabs and security so that each role (Admin, Librarian, Member) can perform its tasks.

1. Standard & Custom Objects

- **Contact (Student):** Used to store student information such as **Date of Birth, Roll Number/Student ID, Contact Details**.
- **Teacher (Custom):** Captures teacher information and tracks their classes/sessions. Includes fields for **Subjects Taught, Qualifications, Availability/Timings**, and **lookup to Student (Contact)** for assigned students or mentoring relationships.
- **Student (Custom):** Tracks student academic cases like **Performance, Issues, or Counseling**. Includes fields for **Description, Plan/Action, Status**, and **lookups to Student (Contact) and Teacher (User)**.

2. Fields

- **Student** – Stores key student details (name, DOB, ID, contact, class, status) and links them to book loans, courses, and mentors to track borrowing and enrollment.
- **Teacher** – Stores main teacher details (name, DOB, ID, subject, contact, status) and links them to students, classes, and book loans to show who teaches or approves activities.

The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The left sidebar contains navigation links for Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, Triggers, Flow Triggers, and Validation Rules. The main content area is titled 'Fields & Relationships' and lists 6 items, sorted by Field Label. The table columns are FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The data includes:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Mentor Name	Mentor_Name__c	Master-Detail(Mentor)		✓
Student Email	Student_Email__c	Email		
Student Phone	Student_Phone__c	Phone		
Students Name	Name	Text(80)		✓

The screenshot shows the Salesforce Object Manager interface for the 'Teacher' object. The left sidebar contains the same navigation links as the Student object. The main content area is titled 'Fields & Relationships' and lists 10 items, sorted by Field Label. The table columns are FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The data includes:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Contact	Contact__c	Lookup(Contact)		✓
Created By	CreatedById	Lookup(User)		
Date of Birth	Date_of_Birth__c	Date		
Email	Email__c	Email (Unique)		✓
Employee ID	Employee_ID__c	Text(10)		
Joining Date	Joining_Date__c	Date		
Last Modified By	LastModifiedById	Lookup(User)		
Mentor	Mentor__c	Master-Detail(Mentor)		✓
Subject Specialization	Subject_Specialization__c	Picklist		
Teacher Name	Name	Text(80)		✓

3. Record Types

- Teacher record types: Full-Time Teacher and Part-Time Teacher.
- Separate page layouts exist for each record type to show relevant fields.

The screenshot shows the Salesforce Object Manager interface for the 'Teacher' object. On the left, a sidebar lists various setup options like Details, Fields & Relationships, Page Layouts, and Record Types. The 'Record Types' option is selected and highlighted in blue. The main content area displays a table titled 'Record Types' with two items: 'Full-Time Teacher' and 'Part-Time Teacher'. The table includes columns for 'Record Type Label', 'Description', 'Active', and 'Modified By'. Both records were created by Manjeet Urmaliya on 9/21/2025 at different times.

RECORD TYPE LABEL	DESCRIPTION	ACTIVE	MODIFIED BY
Full-Time Teacher		✓	Manjeet Urmaliya, 9/21/2025, 6:33 AM
Part-Time Teacher		✓	Manjeet Urmaliya, 9/21/2025, 6:36 AM

Teacher Record Types

The screenshot shows the Salesforce Object Manager interface for the 'Teacher' object. The sidebar again lists various setup options, with 'Page Layouts' selected and highlighted in blue. The main content area displays a table titled 'Page Layouts' with three items: 'Full-Time Teacher Layout', 'Part-Time Teacher Layout', and 'Teacher Layout'. The table includes columns for 'Page Layout Name', 'Created By', and 'Modified By'. All layouts were created by Manjeet Urmaliya on 9/21/2025 at different times.

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Full-Time Teacher Layout	Manjeet Urmaliya, 9/21/2025, 6:53 AM	Manjeet Urmaliya, 9/21/2025, 6:54 AM
Part-Time Teacher Layout	Manjeet Urmaliya, 9/21/2025, 6:55 AM	Manjeet Urmaliya, 9/21/2025, 6:55 AM
Teacher Layout	Manjeet Urmaliya, 9/21/2025, 5:08 AM	Manjeet Urmaliya, 9/21/2025, 5:21 AM

Teacher Page Layouts

4. Page Layouts & Related Lists

5. Lightning Record Pages & Compact Layouts

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
Edit Clone Del	Library Record Page - Full-Time	Library_Record_Page_Full_Time			Record Page	9/21/2025, 7:44 AM	9/21/2025, 7:44 AM
Edit Clone Del	Teacher Record Page - Full-Time	Teacher_Record_Page_Full_Time			Record Page	9/21/2025, 7:34 AM	9/21/2025, 7:34 AM

Library

Custom Lightning Record Pages created for Library to give Librarians and Students tailored views.

Compact Layouts show key Library fields (Book Title, Author, Availability, Genre) for quick glance.

Lightning Page Detail

Information

Name	Description	Label
Library_Record_Page_Full_Time		Library Record Page – Full-Time

Assignments By App

No Assignments to display

Assignments By App, Record Type, and Profile

App	Record Type	Profile	Form Factor
Library Management	Master	Identify User	Desktop and phone
Library Management	Master	Librarian	Desktop and phone
Library Management	Master	Standard User	Desktop and phone
Library Management	Master	System Administrator	Desktop and phone
Library Management	Master	Teacher	Desktop and phone
Library Management	Master	Test profile	Desktop and phone

Teacher

Custom Lightning Record Pages created for Teacher to give Admins and Staff tailored views.

Compact Layouts show key Teacher fields (Name, Subject, Class, Contact) for quick glance.

Lightning Page Detail

Information

Name	Description	Label
Teacher_Record_Page_Full_Time		Teacher Record Page – Full-Time

Assignments By App

No Assignments to display

Assignments By App, Record Type, and Profile

App	Record Type	Profile	Form Factor
Library Management	Full-Time Teacher	Librarian	Desktop and phone
Library Management	Full-Time Teacher	Standard User	Desktop and phone
Library Management	Full-Time Teacher	System Administrator	Desktop and phone
Library Management	Full-Time Teacher	Teacher	Desktop and phone
Library Management	Full-Time Teacher	Test profile	Desktop and phone

6. Tabs & App Navigation

Teacher

Tabs & Navigation: Teachers, Students, Classes, Attendance, Reports. Custom tab for Teachers.

Library

Tabs & Navigation: Library, Books, Students, Issued Books, Reports. Custom tab for Library.

The screenshot shows the Salesforce Setup interface under the 'Tabs' section. On the left sidebar, 'Custom Tabs' is selected under 'Tableau'. The main area displays 'Custom Object Tabs' with four entries: 'Libraries' (purple), 'Mentors' (yellow), 'Students' (green), and 'Teachers' (red). Below this are sections for 'Web Tabs', 'Visualforce Tabs', and 'Lightning Component Tabs', each stating 'No [type] tabs have been defined'. A note at the bottom says 'Didn't find what you're looking for? Try using Global Search.'

7. Schema Builder (ERD)

Teacher

Schema Builder (ERD): Displays Teacher, Student, Class, and User objects with lookup relationships. Teacher is central, linked to Classes and Students.

Library

Schema Builder (ERD): Displays Library, Book, Student, and User objects with lookup relationships. Library is central, linked to Books and Students.

The screenshot shows the Salesforce Schema Builder tool. The left sidebar lists objects like Account, Activity, Address, etc., with checkboxes. The main area displays an Entity Relationship (ER) diagram on a grid background. It includes objects: Contact, Teacher, Student, Mentor, User, and Library. Relationships are shown as lines connecting objects, such as Teacher linking to Contact, Student, and Mentor. Similarly, User links to Contact and Library, and Library links to Book.

8. Security & Field-Level Visibility

- Teacher profile has Read/Create/Edit on Student and Class records.
- Librarian profile has Read/Create/Edit on Library and Book records, but read-only on Student records.
- Sensitive fields like Salary or Private Notes hidden from Librarian profile.
- Mentor profile can view assigned Students and Classes but cannot edit Library records.

9. Sample Data & Testing

- Sample Students and Teachers created with details like Name, Class, and Contact info.
- Library records (Books) created by Librarian, linked to Students and Teachers with record type “Issued” or “Available.”
- Mentor records created and linked to Students for guidance tracking.
- Teacher login view shows Classes and Students; Student record shows related lists (Books issued, Mentor info).

10. Current System Behaviour

- Library records link both Students and Teachers for issuing/tracking books.
 - Mentors hold guidance/notes information linked to the same Students.
 - Record types like Issued Book and Available Book display appropriate layouts.
 - Student records show all related Classes, Library records, and Mentor assignments.
 - Teachers can view Students and Classes assigned to them.
 - Librarians can create Library records and view limited Student info.
 - The Schema Builder ERD reflects the current relationships between Teacher, Library, Mentor, Student, and User.
-

Phase 4: Process Automation (Admin)

1. Validation Rules

- A validation rule created on Teacher object HireDate_Not_Future Date/ Time Hire Date cannot be blank or in the future.
- A validation rule created on Teacher object Email_Bank Please enter a valid email address.

The screenshot shows the Salesforce Object Manager interface for the 'Teacher' object. On the left, a sidebar lists various setup options like Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, Scoping Rules, Object Access, Triggers, Flow Triggers, and Validation Rules. The 'Validation Rules' option is selected. The main area displays a table titled 'Validation Rules' with two items:

RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY
Email_Bank	Email	Please enter a valid email address.	✓	Manjeet Urmalya, 9/22/2025, 10:39 AM
HireDate_Not_Future	Hire Date	Hire Date cannot be blank or in the future.	✓	Manjeet Urmalya, 9/22/2025, 9:34 AM

The screenshot shows a 'Teachers' list view in the 'Library Management' application. A modal dialog box is open, displaying an error message: 'We hit a snag.' It says 'Review the following fields' and lists 'Hire Date'. The 'Hire Date' field is highlighted with a red border. At the bottom of the dialog are 'Cancel', 'Save & New', and 'Save' buttons.

2. Email Templates for Automation

Two classic email templates were created and stored under Private Email Templates:

- Teacher_HireDate_Confirmation – sends confirmation to the teacher when a date is created or confirmed.

Email Template **Teacher_HireDate_Confirmation**

Details Related

Information

Email Template Name Teacher_HireDate_Confirmation	Related Entity Type Teacher
Description	Folder
Made in Email Template Builder	Private Email Templates

Message Content

Subject Confirmation: Record Created for {{Teacher_c.Name}}	Enhanced Letterhead
HTML Value Hello {{Teacher_c.Name}}, This is a confirmation that your record has been created successfully in our system on {{Teacher_c.CreatedDate}}. If you have any questions, please contact us.	
Thank you, Admin Team	

Additional Information

Created By Manjeet Urmaliya, 9/22/2025, 11:32 AM	Last Modified By Manjeet Urmaliya, 9/22/2025, 11:32 AM
---	---

Email Template **Teacher_HireDate_Reminder**

Details Related

Information

Email Template Name Teacher_HireDate_Reminder	Related Entity Type Teacher
Description	Folder
Made in Email Template Builder	Private Email Templates

Message Content

Subject Reminder: Your Hire Date is {{Teacher_c.Hire_Date_c}}	Enhanced Letterhead
HTML Value Hello {{Teacher_c.Name}}, This is a reminder that your hire date is {{Teacher_c.Hire_Date_c}}. Thank you, HR Team	

Additional Information

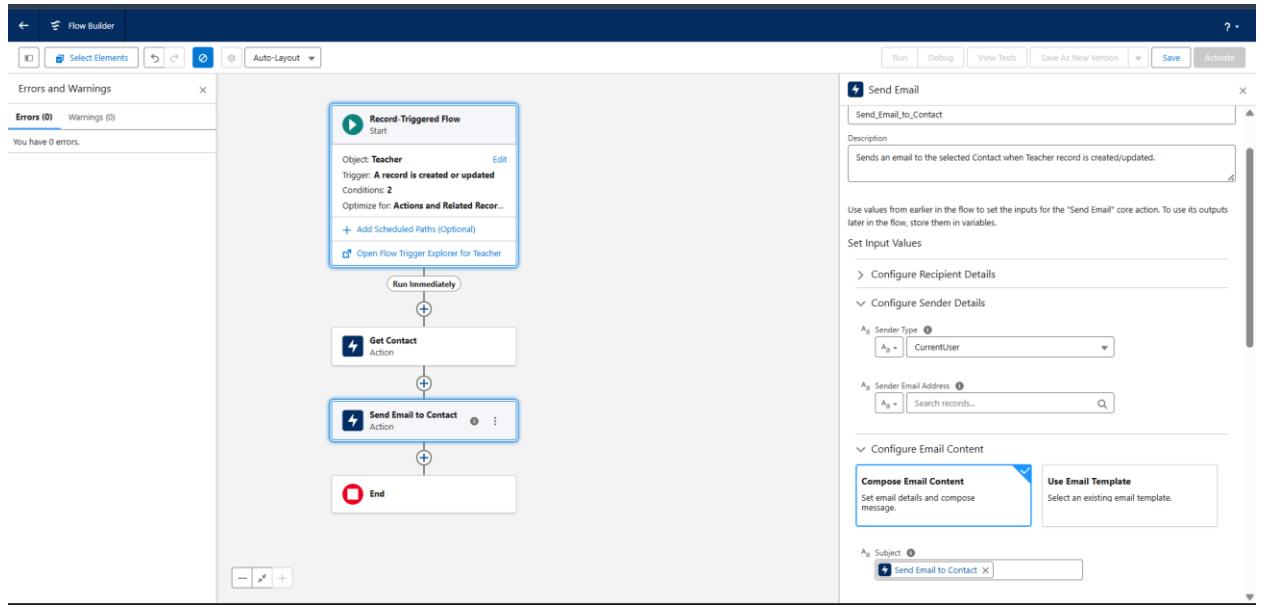
Created By Manjeet Urmaliya, 9/22/2025, 11:20 AM	Last Modified By Manjeet Urmaliya, 9/22/2025, 11:21 AM
---	---

3. Flow for Teacher Confirmation :-

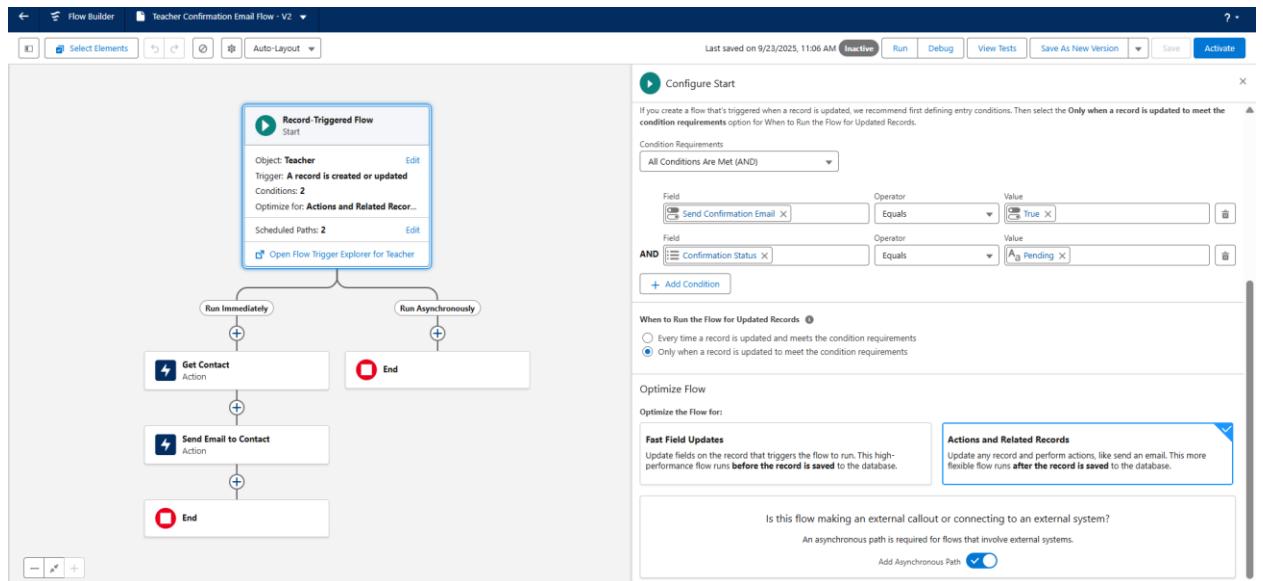
A record-triggered flow “Teacher_Confirmation_Flow” built on Teacher object:

Trigger: When record is created or when Status changes to “Confirmed.”

- Action:
- Update “Confirmation Sent” checkbox on the Teacher.



4. Reminder Notification & Email (Scheduled Path)



6. Quick Action – “Mark as Completed”

A Quick Action added Teacher object:

- Action Type: Update a Record.
- Predefined Field Values: Status c = “Completed.”

8. Testing & Results

Sample Appointments were created to test the automation:

Phase 5: Apex Programming (Developer)

1. Classes & Objects

Created an Apex utility class **TeacherService**:

- Inner Comparator: **TeacherHireDateComparator** – sorts teachers by **Hire_Date__c**.
- Static Method:

- **preventDuplicateHireDates(List<Teacher__c> incoming, Map<Id, Teacher__c> oldMap)** – checks new/updated teacher records against existing teacher records to ensure no conflicting hire dates for the same department (or other criteria) and uses **addError()** to block duplicates.

File → New → Apex Class.

Name it TeacherServiceTest → click **OK**.

Paste test code:

Save.

The screenshot shows the Salesforce IDE interface with the code editor open. The code is a test class named TeacherServiceTestApex. It contains logic to insert two teacher records (t1 and t2) and then run a test block. Inside the test block, it inserts t2 and asserts that an exception is thrown (expected duplicate hire date). A modal dialog titled "Running tests synchronously..." shows a "Success" message with the text "A teacher already has this hire date.". The API version is set to 64.

```

1 Teacher__c t1 = new Teacher__c(
2     Name = 'John Doe',
3     Department__c = 'Math',
4     Hire_Date__c = Date.today()
5 );
6
7 Teacher__c t2 = new Teacher__c(
8     Name = 'Jane Smith',
9     Department__c = 'Science',
10    Hire_Date__c = Date.today()
11 );
12
13 Test.startTest();
14 try {
15     insert t2;
16     System.assert(false, 'Expected duplicate hire date');
17 } catch (DmlException e) {
18     System.assert(e.getMessage().contains('A teacher already has this hire date.'));
19 }
20 Test.stopTest();
21
22 }
```

2. Apex Trigger

File → New → Apex Trigger.

Enter:

- Name: TeacherTrigger
- sObject: Teacher__c

Click **Submit**.

Save.

```

1 * trigger TeacherTrigger on Teacher__c (before insert) {
! 2 *     trigger TeacherTrigger on Teacher__c (before insert, before update) {
3 *         if (Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
4             TeacherService.preventDuplicateHireDates(Trigger.new, Trigger.oldMap);
5         }
6     }
7
8
9 }

```

Logs Tests Checkpoints Query Editor View State Progress Problems 1

Name	Line	Problem
TeacherTrigger	2	Expecting ')' but was: 'trigger'

- **3.Trigger Design Pattern:**

Business logic is kept in the TeacherService class; the trigger on Teacher__c simply calls the class method.

```

1 * public class Teacher_Service {
2
3     // Comparator to sort teachers by Hire Date
4     public class TeacherHireDateComparator implements Comparator<Teacher__c> {
5         public Integer compare(Teacher__c t1, Teacher__c t2) {
6             if (t1.Hire_Date__c == t2.Hire_Date__c) return 0;
7             if (t1.Hire_Date__c > t2.Hire_Date__c) return 1;
8             return -1;
9         }
10    }
11
12    // Static method to prevent duplicate hire dates in same department
13    public static void preventDuplicateHireDates(
14        List<Teacher__c> newTeachers,
15        Map<Id, Teacher__c> oldMap
16    ) {
17        // Collect all teachers from same department
18        Set<String> deptDateCombo = new Set<String>();
19
20        // Overlay existing teacher records

```

File Edit Debug Test Workspace Help < >

TeacherTrigger.apxt TeacherServiceTest.apxc TeacherServiceHandle.apxc TeacherServiceHandleTest.apxc Teacher_Service.apxc

Code Coverage: None API Version: 64

Enqueuing test run... Success

4. SOQL & Collections

Inside preventOverlap:

- **SOQL** queries existing Teacher for relevant new Teacher.

```

1 @isTest
2 public class TeacherServiceHandleTest {
3
4     @isTest
5     static void testPreventDuplicateHireDates() {
6         // Insert first teacher
7         Teacher__c t1 = new Teacher__c(
8             Name = 'John Doe',
9             Department__c = 'Science',
10            Hire_Date__c = Date.today()
11        );
12        insert t1;
13
14        // Insert second teacher with same hire date in same department
15        Teacher__c t2 = new Teacher__c(
16            Name = 'Jane Smith',
17            Department__c = 'Science',
18            Hire_Date__c = Date.today()
19        );
20
21        Test.startTest();
22        try {
23            insert t2;
24            System.assert(false, 'Expected duplicate hire date error.');
25        } catch (DmlException e) {
26            System.assert(e.getMessage().contains('A teacher in this department already has this hire date.'));
27        }
}

```

5. Control Statements

for loops, if conditions and **addError()** to prevent overlaps.

6. Test Classes (Teacher Example)

A test class TeacherTriggerTest was created to:

- Insert a valid teacher (no duplicate hire date in the same department) and verify it saves successfully.
- Attempt to insert a teacher with a duplicate hire date in the same department and verify a **DmlException** with “already has this hire date” in the message is thrown.

This test indirectly executes the logic in

TeacherServiceHandle.preventDuplicateHireDates and gives code coverage for both the trigger (**TeacherTrigger**) and the service class (**TeacherServiceHandle**).

The screenshot shows the Salesforce Apex code editor with the following code:

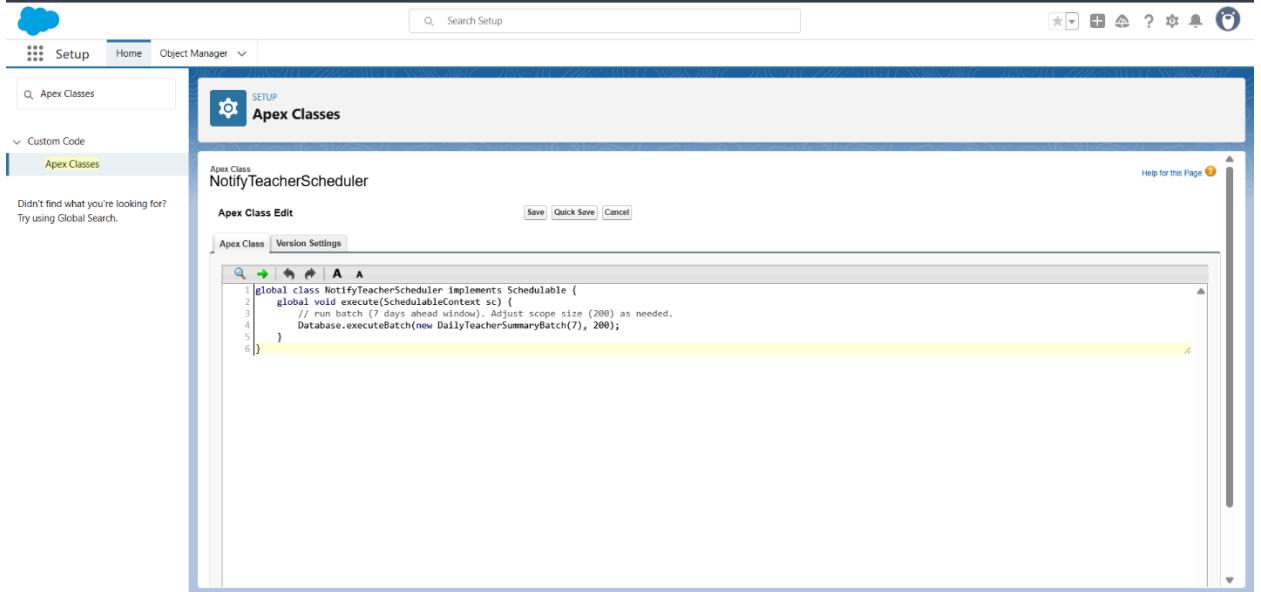
```

1 static void testPreventDuplicateHireDates() {
2     // Insert a valid teacher (no duplicate hire date)
3     Teacher__c t1 = new Teacher__c(
4         Name = 'John Doe',
5         Department__c = 'Science',
6         Hire_Date__c = Date.today()
7     );
8     insert t1;
9
10    // Attempt to insert a teacher with duplicate hire date in same department
11    Teacher__c t2 = new Teacher__c(
12        Name = 'Jane Smith',
13        Department__c = 'Science',
14        Hire_Date__c = Date.today()
15    );
16
17    Test.startTest();
18    try {
19        insert t2;
20        System.assert(false, 'Expected duplicate hire date error.');
21    } catch (DmlException e) {
22        System.assert(e.getMessage().contains('already has this hire date'));
23    }
24    Test.stopTest();
25 }
26
27 }
```

7. Batch Apex and Scheduled Jobs

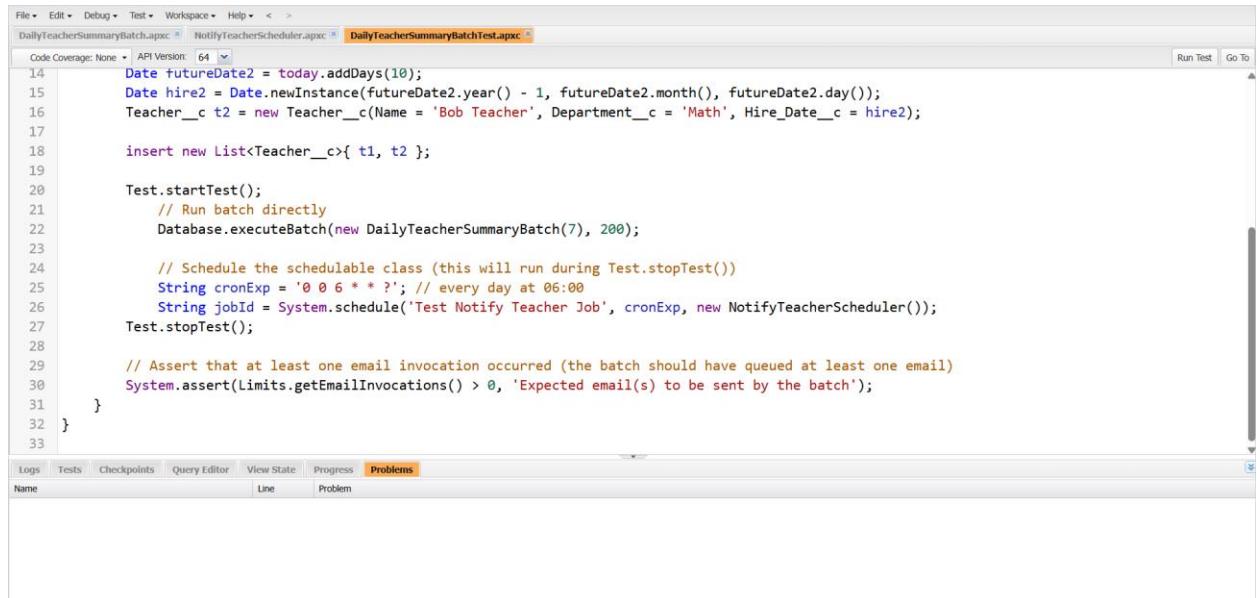
A Batch Apex class TeacherSummaryBatch was created to:

- Query recently hired teachers by department.
- Generate and log summary reports (or prepare reminders for admin review).
- Run daily at 6:00 AM using a Scheduled Apex job.



8. Exception Handling

Used addError() on SObjects to stop DML with a user-friendly message.



The screenshot shows the Salesforce IDE interface with the following details:

- File Bar:** File, Edit, Debug, Test, Workspace, Help.
- Code Coverage:** None
- API Version:** 64
- Open Files:** DailyTeacherSummaryBatch.apxc, NotifyTeacherScheduler.apxc, DailyTeacherSummaryBatchTest.apxc
- Code Content:**

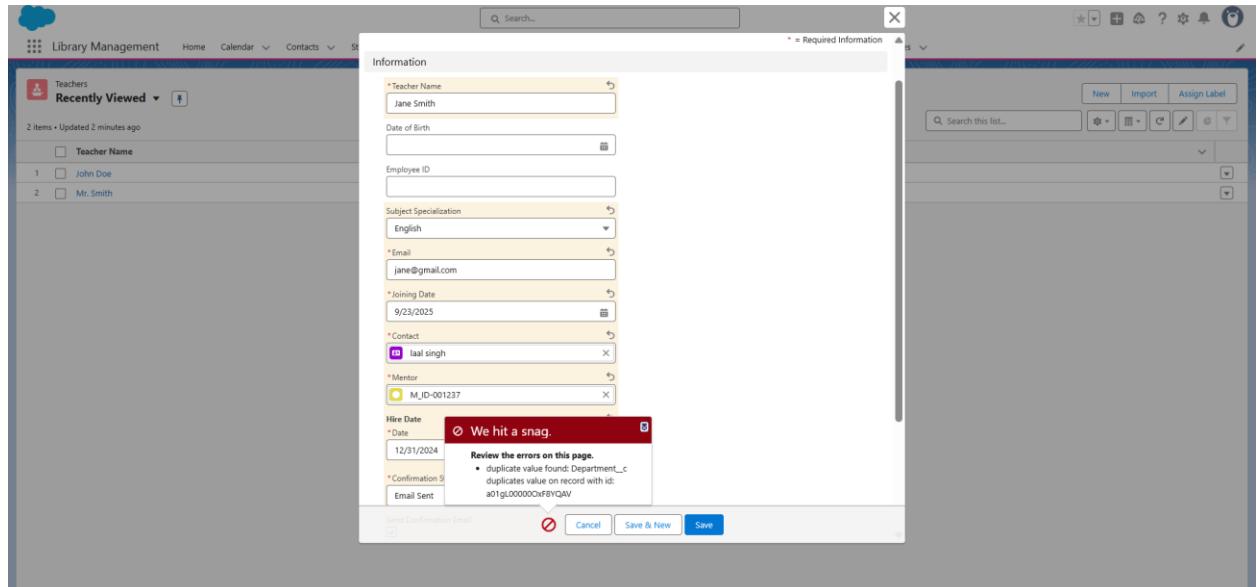
```

14 Date futureDate2 = today.addDays(10);
15 Date hire2 = Date.newInstance(futureDate2.year() - 1, futureDate2.month(), futureDate2.day());
16 Teacher__c t2 = new Teacher__c(Name = 'Bob Teacher', Department__c = 'Math', Hire_Date__c = hire2);
17
18 insert new List<Teacher__c>{ t1, t2 };
19
20 Test.startTest();
21     // Run batch directly
22     Database.executeBatch(new DailyTeacherSummaryBatch(7), 200);
23
24     // Schedule the schedulable class (this will run during Test.stopTest())
25     String cronExp = '0 0 6 * * ?'; // every day at 06:00
26     String jobId = System.schedule('Test Notify Teacher Job', cronExp, new NotifyTeacherScheduler());
27     Test.stopTest();
28
29     // Assert that at least one email invocation occurred (the batch should have queued at least one email)
30     System.assert(Limits.getEmailInvocations() > 0, 'Expected email(s) to be sent by the batch');
31 }
32 }
33 
```
- Toolbars:** Run Test, Go To.
- Tabs:** Logs, Tests, Checkpoints, Query Editor, View State, Progress, Problems (selected).
- Bottom Navigation:** Name, Line, Problem.

9. Skipped / Not Implemented

Queueable Apex, Future Methods, Asynchronous Processing beyond the daily batch job were not implemented at this phase.

10. Testing & Results



Phase 6: User Interface Development

Introduction

This phase focuses on building an intuitive Lightning experience for different library roles (Teacher, Student, Librarian/Admin).

Using Lightning App Builder, Record Pages, Tabs, Home Page Layouts, Utility Bar, plus a custom LWC and Apex integration, a seamless UI was delivered to every user type.

1. Lightning App Builder

- Customized Pages for Each Role: Different Lightning Record Pages were designed for Book/Resource – Teacher View and Book/Resource – Student View (and optionally Librarian/Admin View).

Role-Based Components: Each page displays only the components relevant to that specific role, e.g.

- Teacher: can upload resources, create assignments, view students' borrowing history.
- Student: can search books, borrow/return, view personal borrowing history.
- Librarian/Admin: can add/edit/delete books, manage stock, approve requests.

- Tabs & Home Layouts: Tabs and Home Page Layouts were set up separately for Teacher, Student, and Librarian/Admin so each profile lands on the correct workspace.

Utility Bar: Added quick-access actions and shortcuts in the Utility Bar for commonly used features by each role, e.g.

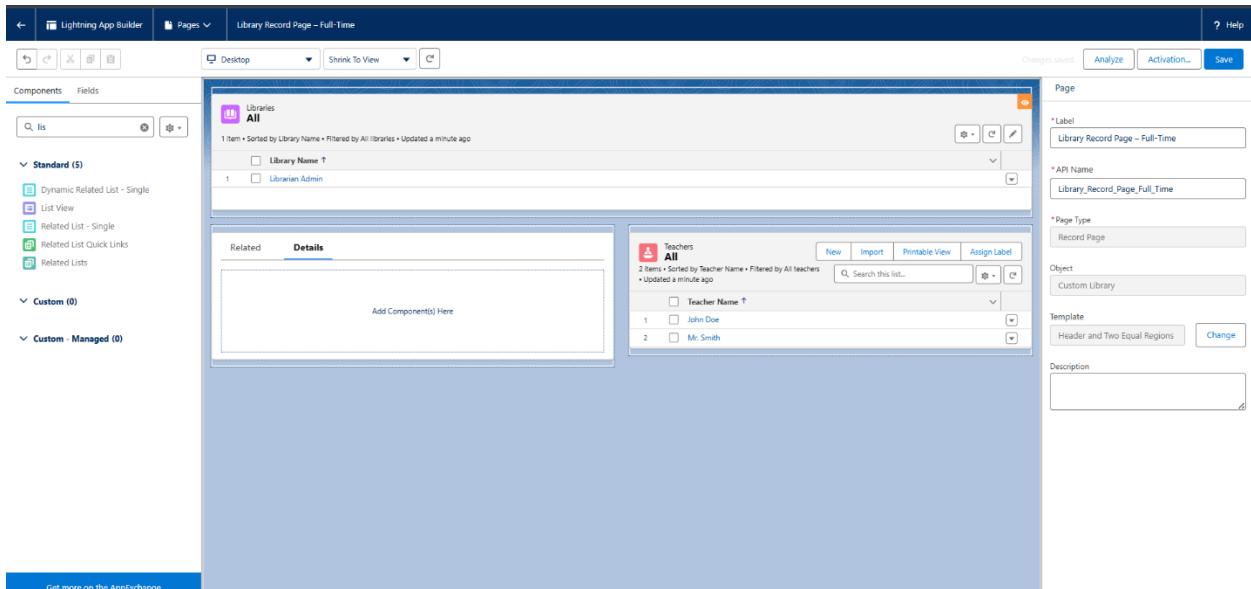
Librarian/Admin: "Add New Book," "View Overdue Items," "Send Notifications."

The screenshot shows the Salesforce Lightning App Builder interface. At the top, there's a navigation bar with icons for Setup, Home, Object Manager, and a search bar labeled "Search Setup". Below the navigation is a sidebar titled "User Interface" with a "Lightning App Builder" section. A message says "The Lightning App Builder provides an easy to use graphical interface for creating custom Lightning pages for Salesforce Lightning Experience and mobile app. Lightning pages are built using Lightning components—compact, configurable, and reusable elements that you can drag and drop into regions of the page in the Lightning App Builder." There's also a "Create New View" button. The main content area is titled "Lightning Pages" and contains a table with the following data:

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
Edit Clone Del	Full-Time Teacher Layout	Full_Time_Teacher_Layout			Record Page	cse 9/22/2025, 3:32 AM	cse 9/22/2025, 3:32 AM
Edit Clone Del	Library Home	Library_Home			Record Page	cse 9/23/2025, 9:43 PM	cse 9/23/2025, 9:43 PM
Edit Clone Del	Library Record Page – Full-Time	Library_Record_Page_Full_Time			Record Page	cse 9/21/2025, 7:44 AM	cse 9/21/2025, 7:44 AM
Edit Clone Del	Teacher Record Page – Full-Time	Teacher_Record_Page_Full_Time			Record Page	cse 9/21/2025, 7:34 AM	cse 9/23/2025, 9:25 PM

2. Record Pages (Teacher & Student Views)

- Teachers need quick access to class resources and student borrowing history. Students need an easy way to browse and request library materials. Two record pages were created with appropriate components:
-
- Library Resource – Teacher View: shows resource summary, related student borrow/return history, and an upcoming assignments/resources LWC.
-
- Library Resource – Student View: optimized for quick browsing, book request/renewal, and viewing personal borrowing history.
-
- Activated per profile/record type so each user sees the correct page automatically.



Library(Record Page)

The screenshot shows the Lightning App Builder interface. At the top, it says "Lightning App Builder" and "Pages" with "Teacher Record Page – Full-Time". On the left, there's a sidebar titled "Components" with sections for "Standard (42)" and "More Components". The main area shows a page layout with two sections: "Related" and "Details". The "Details" section contains fields for "Teacher Name" (John Doe), "Employee ID", "Subject Specialization" (English), "Email" (john@gmail.com), "Joining Date" (9/9/2025), "Contact" (good Ajay), "Mentor" (M_ID-001237), "Hire Date" (12/12/2024, 10:45 AM), "Confirmation Status", "Email Sent", "Send Confirmation Email" (checked), "Last Email Sent Date" (9/10/2025, 12:00 PM), "Department" (Science), and "Onboarding Stage". Below these fields are "Created By" (Manjeet Urmaliya) and "Last Modified By" (Manjeet Urmaliya). On the right, there's a "Page" configuration panel with fields for "Label" (Teacher Record Page – Full-Time), "API Name" (Teacher_Record_Page_Full_Time), "Page Type" (Record Page), "Object" (Teacher), "Template" (Header and Two Equal Regions), and "Description". There's also a checkbox for "Enable page-level dynamic actions for the Salesforce mobile app".

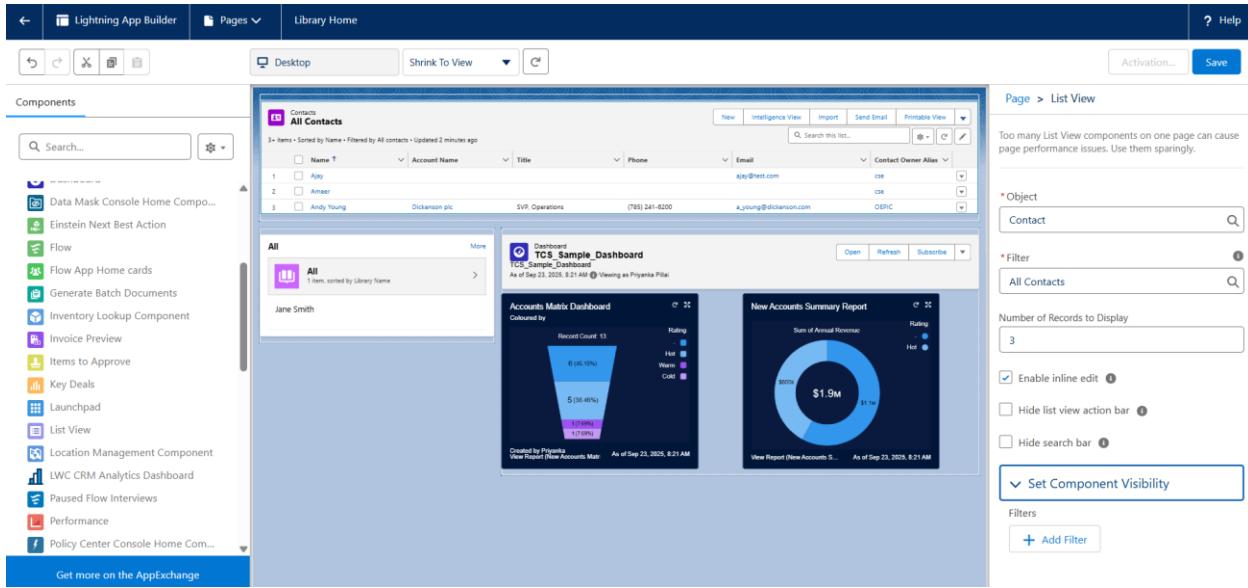
3. Tabs

All major library objects (Books/Resources, Borrow Requests, Students/Teachers, Classes) were exposed as tabs in the Library Management Lightning App so Teachers, Students, and Librarians can navigate quickly to the areas they need.

The screenshot shows the Library Management Lightning App Home Page. At the top, there's a navigation bar with tabs for "Home", "Calendar", "Contacts", "Students", "Mentors", "Accounts", "Change Requests", "Custom Libraries", "Dashboards", "Teachers", and "Libraries". Below the navigation bar, there's a section titled "Quarterly Performance" and an "Assistant" button.

4. Home Page Layouts

A custom Library Management Home Page was built that displays dashboard components such as Upcoming Book Returns, Recent Borrow Requests, Popular Books, and Overdue Items Report Chart to give Teachers, Students, and Librarians a real-time view of library activity.



Phase 7 — Integration & External Access (Implemented Parts)

Step 1: Create an External Credential (one time)

Go to **Setup** → **Quick Find** → type **External Credentials** → click **External Credentials**.

Click **New External Credential**.

Fill the form:

- Label:** GoogleMapsAPIAuth
- Name:** GoogleMapsAPIAuth
- Authentication Protocol:** No Authentication

Click **Save**.

You now have an External Credential called **GoogleMapsAPIAuth**.

Step 2: Create the Named Credential (link to External Credential)

Go to **Setup** → **Quick Find** → **Named Credentials** → click **Named Credentials** → click **New**.

Fill details:

- **Label:** GoogleMapsAPI
- **Name:** GoogleMapsAPI
- **External Credential:** pick **GoogleMapsAPIAuth**
- **URL:** <https://maps.googleapis.com>
- **Generate Authorization Header:** leave checked
- **Authentication:** nothing extra (because using API key)

Click **Save**.

The screenshot shows the Salesforce Setup interface for creating a Named Credential. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main area is titled 'SETUP > NAMED CREDENTIALS' and shows a record for 'GoogleMapsAPIAuth'. The 'Label' field is set to 'GoogleMapsAPIAuth', 'Name' to 'GoogleMapsAPIAuth', and 'URL' to 'https://maps.googleapis.com'. The 'Authentication' section shows 'External Credential' selected as 'GoogleMapsAPIAuth'. The 'Callout Options' section has 'Generate Authorization Header' checked. The 'Principals' section is currently empty, indicated by the message 'You don't have any principals configured.'

This screenshot shows the continuation of the Named Credential setup for 'GoogleMapsAPI'. The 'Callout Options' section is expanded, displaying three checked checkboxes: 'Generate Authorization Header', 'Allow Formulas in HTTP Header', and 'Allow Formulas in HTTP Body'. The 'Authentication' section shows 'External Credential' selected as 'GoogleMapsAPIAuth'. The 'Managed Package Access' section is present at the bottom. The rest of the interface is identical to the previous screenshot, showing the 'SETUP > NAMED CREDENTIALS' header and the 'GoogleMapsAPIAuth' record.

```
public with sharing class LibraryMapsService {  
    // Returns raw JSON string or you can parse as needed  
    public static String getLocationData(String branchAddress) {  
        HttpRequest req = new HttpRequest();  
        req.setEndpoint(  
            'callout:GoogleMapsAPI/maps/api/geocode/json?address=' +  
            EncodingUtil.urlEncode(branchAddress,'UTF-8') +  
            '&key=YOUR_KEY'  
        );  
        req.setMethod('GET');  
  
        HttpResponse res = new Http().send(req);  
  
        // Parse JSON if needed  
        Map<String, Object> result =  
            (Map<String, Object>) JSON.deserializeUntyped(res.getBody());  
  
        // For now just return raw response  
        return res.getBody();  
    }  
}
```

The screenshot shows the Salesforce Developer Console and the Apex Classes page for the class `LibraryMapsService`.

Developer Console - Google Chrome

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

DailyTeacherSummaryBatch.apxc NotifyTeacherScheduler.apxc DailyTeacherSummaryBatchTest.apxc LibraryMapsService.apxc

Code Coverage: None API Version: 64

```

1 public with sharing class LibraryMapsService {
2     // Returns raw JSON string or you can parse as needed
3     public static String getLocationData(String branchAddress) {
4         HttpRequest req = new HttpRequest();
5         req.setEndpoint(
6             'callout:GoogleMapsAPI/maps/api/geocode/json?address=' +
7             EncodingUtil.urlEncode(branchAddress, 'UTF-8') +
8             '&key=YOUR_KEY'
9         );
10        req.setMethod('GET');
11
12        HttpResponse res = new Http().send(req);
13
14        // Parse JSON if needed
15        Map<String, Object> result =
16            (Map<String, Object>) JSON.deserializeUntyped(res.getBody());
17
18        // For now just return raw response
19        return res.getBody();
20    }

```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name: `apex` Search Setup

Setup Home Object Manager

Apex Classes

Apex Class Detail

Name: `LibraryMapsService`

Namespace Prefix: `ManjeetUrmaliya`

Created By: `Manjeet Urmaliya`, 9/25/2025, 10:05 PM

Status: `0% (0/1)`

Last Modified By: `Manjeet Urmaliya`, 9/25/2025, 10:06 PM

Class Body

```

1 public with sharing class LibraryMapsService {
2     // Returns raw JSON string or you can parse as needed
3     public static String getLocationData(String branchAddress) {
4         HttpRequest req = new HttpRequest();
5         req.setEndpoint(
6             'callout:GoogleMapsAPI/maps/api/geocode/json?address=' +
7             EncodingUtil.urlEncode(branchAddress, 'UTF-8') +
8             '&key=YOUR_KEY'
9         );
10        req.setMethod('GET');
11
12        HttpResponse res = new Http().send(req);
13
14        // Parse JSON if needed
15        Map<String, Object> result =
16            (Map<String, Object>) JSON.deserializeUntyped(res.getBody());
17
18        // For now just return raw response
19        return res.getBody();
20    }

```

An LWC is always 3 separate files inside force-app/main/default/lwc/libraryMap/:

- `libraryMap.html`
- `libraryMap.js`
- `libraryMap.js-meta.xml`

Here's the **correct code** for your libraryMap LWC — all in one answer:



`<template>`

```
<lightning-card title="Nearest Library Branch">  
  <div class="slds-p-around_medium">  
    <iframe  
      src={mapUrl}  
      width="100%"  
      height="300"  
      style="border:0;"  
      allowfullscreen  
      loading="lazy">  
    </iframe>  
  </div>  
</lightning-card>  
</template>
```

libraryMap.js

```
import { LightningElement, api } from 'lwc';  
  
export default class LibraryMap extends LightningElement {  
  @api branchAddress;  
  mapUrl;  
  
  connectedCallback() {  
    // Directly build Google Maps iframe URL  
    this.mapUrl =  
      'https://www.google.com/maps/embed/v1/place?key=YOUR_KEY&q=' +  
      this.branchAddress;
```

```
    }  
}  
  
}
```

libraryMap.js-meta.xml

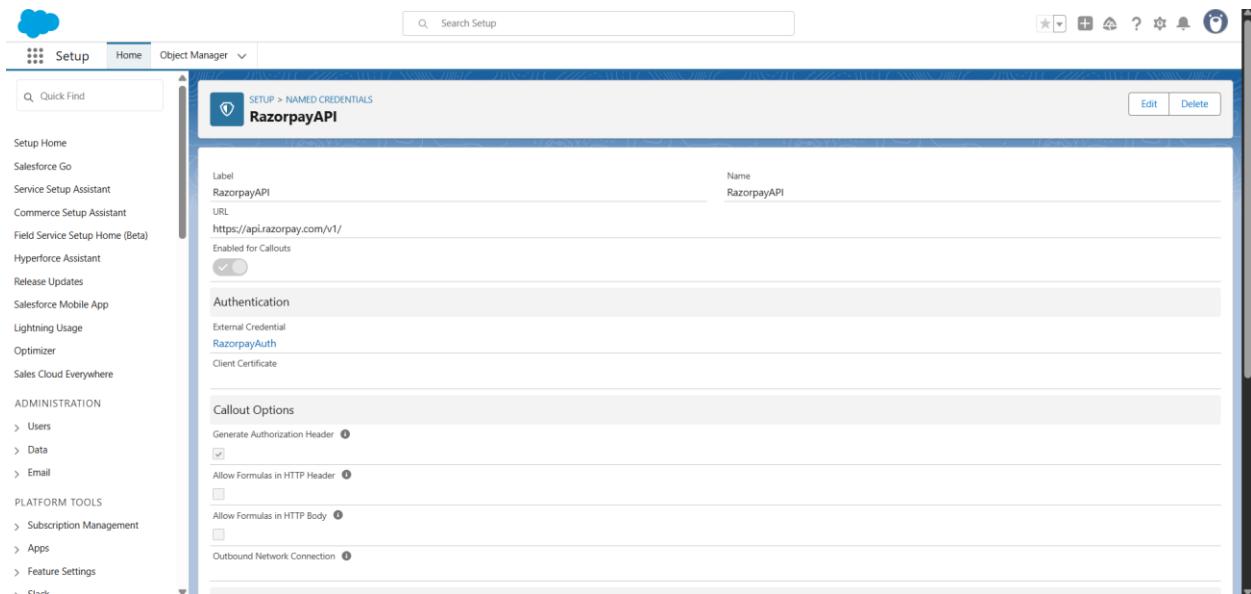
```
<?xml version="1.0" encoding="UTF-8"?>  
  
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">  
    <apiVersion>59.0</apiVersion>  
    <isExposed>true</isExposed>  
    <targets>  
        <target>lightning__RecordPage</target>  
        <target>lightning__AppPage</target>  
        <target>lightning__HomePage</target>  
    </targets>  
    <targetConfigs>  
        <targetConfig targets="lightning__RecordPage,lightning__AppPage,lightning__HomePage">  
            <property name="branchAddress" type="String" label="Branch Address"  
            description="Address to show on map"/>  
        </targetConfig>  
    </targetConfigs>  
</LightningComponentBundle>
```

Step 3:

Create a Named Credential

1. In Setup, search **Named Credentials** → **New**.
2. Fill:
 - **Label:** RazorpayAPI
 - **Name:** RazorpayAPI
 - **External Credential:** **RazorpayAuth** (the one you just made)
 - **URL:** <https://api.razorpay.com/v1/>
 - Leave “Generate Authorization Header” checked.

3. Save.



```
public with sharing class RazorpayPaymentService {
    public static String createPaymentLink() {
        HttpRequest req = new HttpRequest();
        req.setEndpoint('callout:RazorpayAPI/payment_links'); // note plural endpoint
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/json');
        req.setBody('{'
            + '"amount": 1000,' +
            + '"currency": "INR",' +
        });
    }
}
```

```

    "customer": {"email": "test@example.com"}' +
}

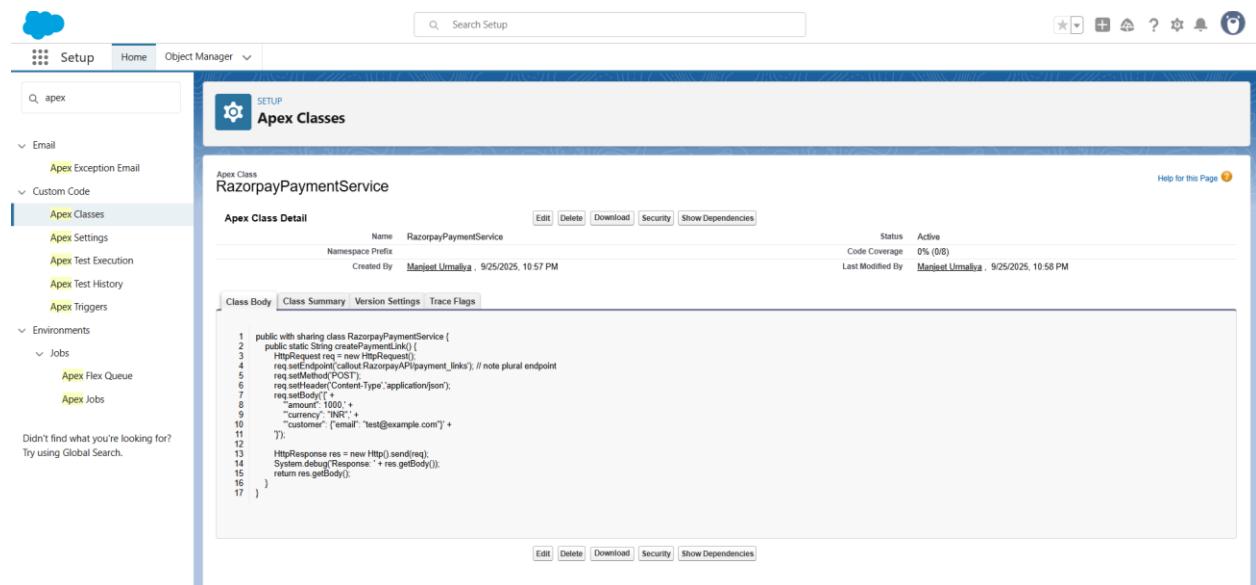
HttpServletResponse res = new HttpResponse().send(req);

System.debug('Response: ' + res.getBody());

return res.getBody();

}
}

```

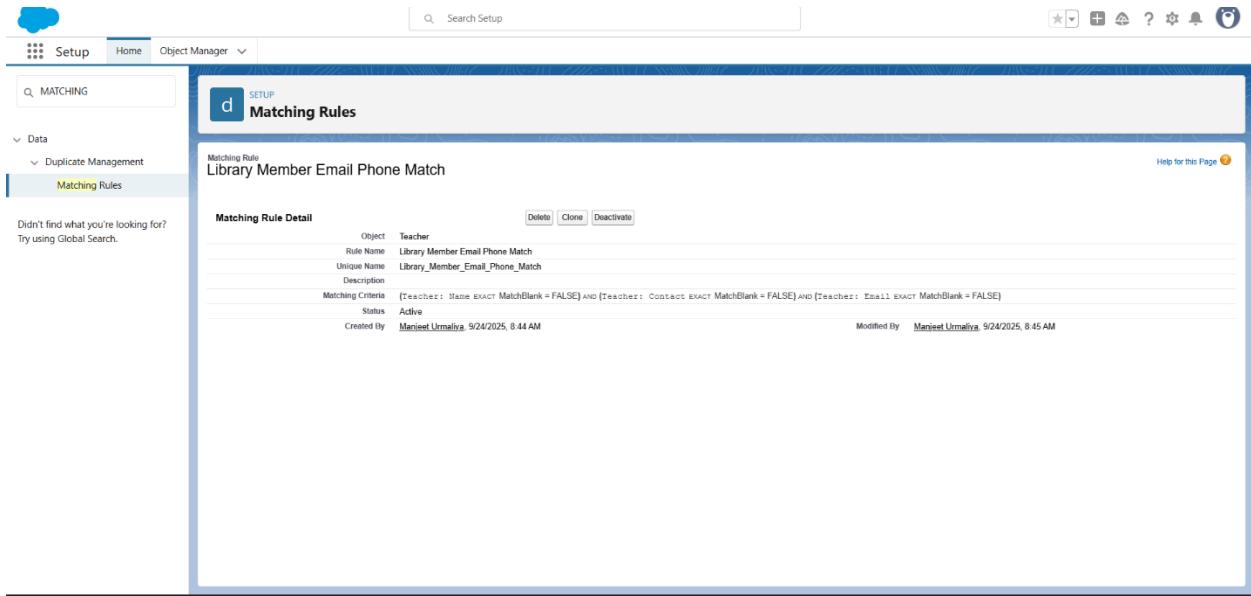


Phase 8: Data Management & Deployment

1. Create the Matching Rule

- Setup → Quick Find → Matching Rules → New Rule
- Object: Teacher
- Rule Name: Library Member Email Phone Match
- Add Matching Criteria:
 - Field = Email → Matching Method = Exact
 - Click Add Row → Field = Phone → Matching Method = Exact

- Click Save
- Click Activate



2. Create the Duplicate Rule

- Setup → Quick Find → Duplicate Rules → New Rule
- Object: Teacher
- Rule Label: LibraryMember_DuplicateRule
- Under Matching Rules: click Add and select the LibraryMember_Email_Phone_Match matching rule you activated in the previous step.
- Action on Create: choose Alert (recommended while testing so users are warned but not blocked).

- Action on Edit: choose Alert.

Step 2: Data Backup – Library Management

Goal: Regularly back up all key Library Management data (Teachers, Students, Books, Borrow Requests, etc.) for safety and compliance.

Steps

- Go to Setup.
 - In Quick Find, type Data Export → click Data Export.
 - Choose one:
 - Export Now → run a one-time backup.
 - Schedule Export → set weekly or monthly backups.
 - Select the objects you want:
 - Library Members (Teachers/Students)
 - Books/Resources
 - Borrow Requests / Returns
 - Classes / Courses
 - Librarians (Users)
- Any other standard objects you use (e.g., Users, Attachments).
- Click Start Export (for immediate) or Save (for scheduled).
 - Wait — Salesforce will email you when the backup is ready.

- Download the .zip file from the export page → extract CSV files.
- Store the backup securely (encrypted drive, secure server, or trusted cloud storage).

The screenshot shows the Salesforce Data Export Service page. The top navigation bar includes links for Home, Chatter, Campaigns, Leads, Accounts, Contacts, Opportunities, Forecasts, Contracts, Orders, Cases, Solutions, Products, Reports, Dashboards, Students, Teachers, Libraries, and Sales. A sidebar on the left titled "Administrator" lists various management options under "Data Management" and "Data Export". The main content area is titled "Monthly Export Service" and displays a message about preparing a copy of all data. It shows a scheduled export entry for "Manjeet Urmaliya" on "9/24/2025" with an ISO-8859-1 encoding. A table lists the export details: Action (download), File Name (WE_000gL000007XhYPUAO_1.ZIP), and File Size (1.4K). At the bottom, there are navigation icons and a status message indicating 1 of 1,342 records.

The screenshot shows an email notification from "Your Organization Data Export has completed - Gyan Ganga College Of Technology". The email is from "Do not reply <noreply@salesforce.com>" and was sent "10:07 (17 minutes ago)". It contains a link to the completed export: <https://orgfarm-b173f695e9-dev-ed.my.salesforce.com/ui/setup/export/DataExportPage/d>. The message thanks the user for their organization's data export completion and provides a link to view it. Below the message are "Reply" and "Forward" buttons.

A	B	C	D	E	F	G	H	I	J
1	Id	OwnerId	IsDeleted	Name	CreatedDate	CreatedById	LastModifiedDate	LastModifiedById	SystemModstamp
2	a02g1.000C.005gl.000005DWjxQ		0	Librarian Admin	9/24/2025 6:10:00	005gl.000005DWjxQAf	9/24/2025 6:10:00	005gl.000005DWjxQAG	9/24/2025 6:10:00
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									

Phase 9: Reporting, Dashboards & Security Review

Step 1: Reporting

A Report is a list or summary of records in Salesforce that helps you analyze library data.

- Reports let you filter, sort, group, and summarize your Library Management data.
- Example: You can create a report to see all books borrowed this month or the top books borrowed by Teachers vs. Students.

Steps

- Go to the App Launcher → search “Reports” → click Reports.
- Click “New Report” - Monthly Borrowed Books.
- Click “New Report”.
- Select the Report Type - Teacher
- Click Continue.

- Add Filters
- Add Columns
- Group Data (optional)
 - Example: Group Borrow Requests by Book or by Member (Teacher/Student).
- Summarize (optional)
 - Add totals, averages, or counts (like total borrowed books per Teacher).
- Click Save & Run → Give the report a name → Choose a folder to save it in.

The screenshot shows the Microsoft Power BI interface. The top navigation bar includes links for Home, Calendar, Contacts, Dashboards, Students, Mentors, Accounts, Change Requests, Custom Libraries, Teachers, Libraries, and Reports. The Reports section is currently selected. On the left, a sidebar lists categories: Reports (Recent, Created by Me, Private Reports, Public Reports, All Reports), Folders (All Folders, Created by Me, Shared with Me), and Favorites (All Favorites). The main content area displays a table of recent reports with columns: Report Name, Description, Folder, Created By, Created On, and Subscribed. The table contains the following data:

Report Name	Description	Folder	Created By	Created On	Subscribed
Monthly Borrowed Books		Teacher_Reports	Manjeet Urmaliya	9/25/2025, 4:28 AM	
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	7/18/2025, 8:45 PM	
Joined report - Acc cont lead		Tcs Report folder	Manjeet Urmaliya	8/14/2025, 7:43 AM	
Exercise Completion by Days to Complete	Analyze an exercise's completion status based on average days to complete.	Tcs Report folder	Automated Process	7/18/2025, 8:45 PM	
New Accounts Summary report Report		Private Reports	Manjeet Urmaliya	8/14/2025, 7:32 AM	
New Accounts Tabular Report		Private Reports	Manjeet Urmaliya	8/14/2025, 6:52 AM	
New Accounts Matrix Report Report		Private Reports	Manjeet Urmaliya	8/14/2025, 7:02 AM	
Tabular Custom_report_test Report		Private Reports	Manjeet Urmaliya	8/14/2025, 6:39 AM	

Step 2: Dashboards

What it is:

A Dashboard is a visual board that displays charts, numbers, and tables built from your Reports. It helps Teachers, Librarians and Management instantly see performance trends (for example: most borrowed books, top students, pending returns).

Dashboard Components You Can Add

Charts – Bar, Pie, Line charts to show trends.

Gauges – Show progress toward a target (e.g. % of books returned on time).

Metrics – One key number (e.g. total borrowings this month).

Tables – Raw report data in a grid.

Steps (Click by Click)

- Go to the App Launcher → type “Dashboards” → click Dashboards.
- Click “New Dashboard”.
- Enter Dashboard Name – e.g. Library Performance Dashboard or Teacher Engagement Dashboard.
- Choose a Folder (select the folder you created earlier for Reports/Dashboards).
- Click “Create”.
- Click “+ Component” (to add your first chart, metric or table).
- Select a Report you already built (for example: “Books Borrowed This Month” or “Top Students”).
- Choose Visualization Type:
 - Bar Chart
 - Pie Chart
 - Gauge
 - Metric
 - Table
- Configure Data Display:
 - Choose which fields, groupings, and filters you want for this chart or metric.
 - Click “Add”.
 - Repeat “+ Component” for more charts/metrics if needed.
 - Click “Save” → then “Done”.
 - Click “Refresh” anytime to pull in the latest data.
- Result:
 - Your dashboard now shows live numbers and visuals from your Salesforce reports for Teachers, Students and Librarians.

Monthly Borrowed Books

Teacher ID	Send Confirmation Email	Contact	Subject Specializati...	Departm...
a01gl.00000Opjlg	<input checked="" type="checkbox"/>	taul singh	Mathematics	-
a01gl.00000Ox8Y	<input checked="" type="checkbox"/>	good Ajay	English	Science

New Accounts Summary report

Sum of Employees

Rating

-
- Hot
- Warm
- Cold

242k

146k

64k

30k

13k

Step 3: Field-Level Security (Library Management)

What it is:

Field-Level Security controls who can see or edit individual fields on a record.

It protects sensitive data like Teacher Contact, Student DOB, Library Fines, Book Costs, etc., so only the right roles can view or update them.

Steps (Click by Click)

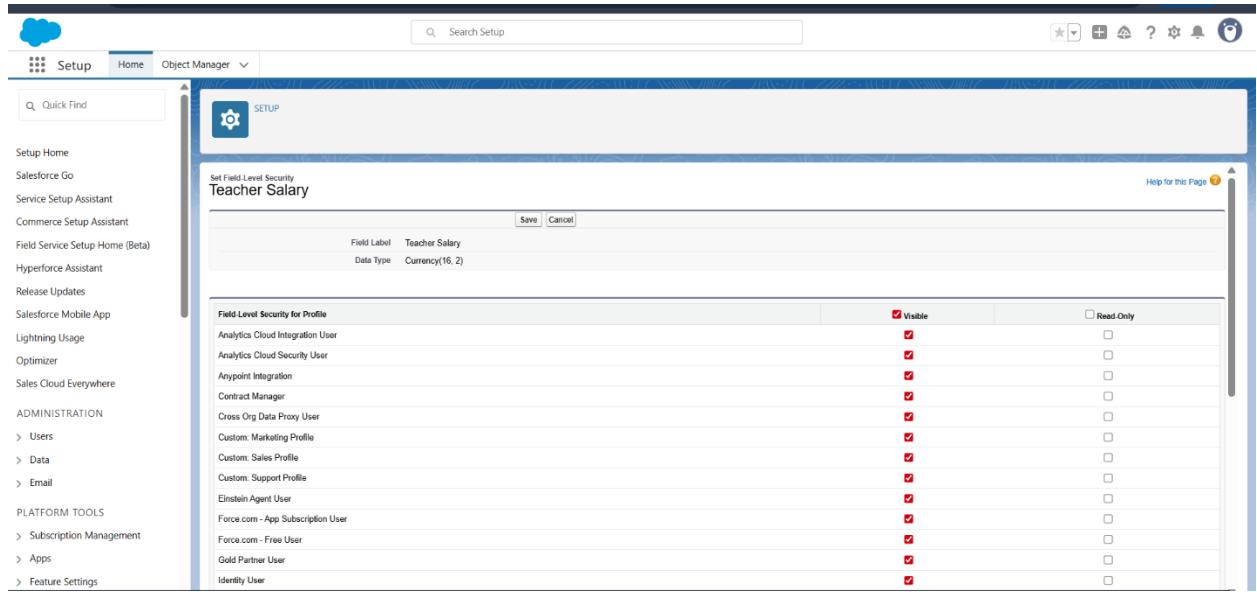
- Go to Setup → Object Manager.
- Select your Object (example: Student or Teacher or Book).
- Click “Fields & Relationships”.
- Click the field you want to protect (example: Student DOB).
- Click “Set Field-Level Security”.
- For each Profile:
 - Visible → can see the field.
 - Read-Only → can see but cannot edit.
 - Hidden → cannot see at all.

- (Example: For Student DOB → Visible to Teacher & Librarian profiles, Hidden for Student profile.)
- Click “Save”.
- Repeat steps 4–7 for other sensitive fields like Teacher Salary, Book Purchase Price, Library Fine Amount, etc.

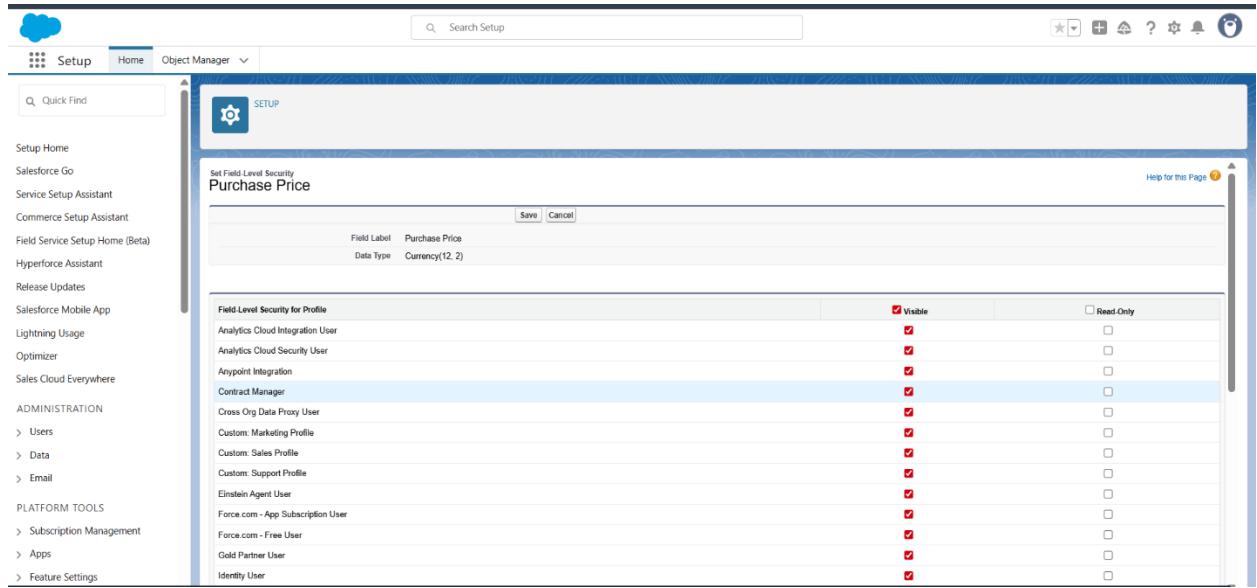
Result:

Only authorized roles (Teacher, Librarian, Admin) can view or edit those sensitive fields. Students or other profiles see nothing or read-only according to your settings.

- Setup → Object Manager → Teacher → Fields & Relationships → New.
- Choose Field Type = Currency → Next.
- Field details:
 - Field Label: Teacher Salary
 - Field Name: Teacher_Salary__c
 - Length / Decimal Places: choose e.g. 16,2 (or your org standard)
 - Help Text: Annual salary in local currency. Visible to Admin only.
 - Description: Stores salary for payroll/audit
 - Set Field-Level Security (during creation)
 - On the Set Field-Level Security screen (shown while creating), pick visibility per profile:
 - visible for all
 - Save.



- Setup → Object Manager → Student → Fields & Relationships → New.
- Field Type = Currency → Next.
- Field Label: Purchase Price
- Field Name: Purchase_Price__c
- Length/Decimals: e.g., 12,2
- Help Text: Cost price of the book — visible to Librarians & Admins. Read-only for Teachers.
- Set Field-Level Security (during creation)
- Choose visibility: all
- Save



- Setup → Object Manager → Mentor → Fields & Relationships → New.
- Field Type = Currency → Next.
- Field Label: Fine Amount
- Field Name: Fine_Amount__c
- Help Text: Fine charged for overdue returns — visible to Librarians only.
- Set Field-Level Security
- Visibility choices: for all → Save.