Internal Job Application Management System CRM

Prepared by : Gollapalli.Lakshmi Tulasi Jyothi Sri

Batch : 4

Project Overview

The proposed CRM system is designed to streamline and optimize the recruitment and job application process within an organization. It provides a centralized platform to manage candidate applications, track interview progress, and automate communication with applicants. The system integrates key features such as candidate shortlisting, automated email notifications, interview scheduling, and real-time status updates. By addressing business needs such as reducing manual effort, improving candidate engagement, and ensuring a transparent hiring process, the CRM enhances efficiency and delivers a structured recruitment workflow.

Objectives

The primary objective of this CRM is to improve the efficiency and accuracy of the recruitment process while ensuring a seamless experience for both candidates and recruiters. The system aims to automate repetitive tasks such as sending updates, managing application records, and handling interview scheduling, thereby reducing manual workload. Additionally, it enhances decision-making by providing recruiters with organized candidate data and status tracking. From a business perspective, the CRM delivers value by accelerating hiring timelines, reducing administrative overhead, and strengthening employer branding through professional and timely communication with applicants.

Phase 1: Problem Understanding & Industry Analysis

- **Requirement Gathering**

  - Identified issues with manual tracking of job applications.

  - Noted delays in communication between recruiters and candidates.

  - Highlighted lack of real-time application status updates.

  - Collected inputs from HR teams on their pain points.

- **Stakeholder Analysis**

  - Recruiters: Need faster tracking and communication tools.

  - Hiring Managers: Require better candidate evaluation insights.

  - Candidates: Expect timely updates and smooth experience.

  - HR Teams: Want reduced manual work and organized data.

- **Business Process Mapping**

  - Documented current recruitment workflow step by step.

  - Found repetitive manual tasks causing inefficiency.

  - Identified communication gaps during interview scheduling.

  - Mapped areas where automation can streamline hiring.

- **Industry-specific Use Case Analysis**

  - Researched common Applicant Tracking Systems (ATS).

  - Compared features like scheduling, status tracking, and reporting.

- ○ Identified best practices widely used across industries.

- ○ Applied insights to tailor CRM for recruitment needs.

- **AppExchange Exploration**

  - ○ Explored recruitment-related Salesforce apps.

  - ○ Noted features like candidate management and automation.

  - ○ Understood limitations of existing solutions.

  - ○ Used findings to design a more customized CRM.


Phase 2: Org Setup & Security Configuration

1.Roles

Use Case:
Roles in Salesforce define the hierarchy of data access. In this project, HR Managers need to access all job applications, while Job Trackers should only see records they own. By defining roles, we can implement secure, role-based access that mirrors organizational responsibilities.

Implementation Steps:

1. Setup → Quick Find → Roles → Set Up Roles → Add Role → HR Manager → Save.

2. Under HR Manager → Add Role → Job Tracker → Save.
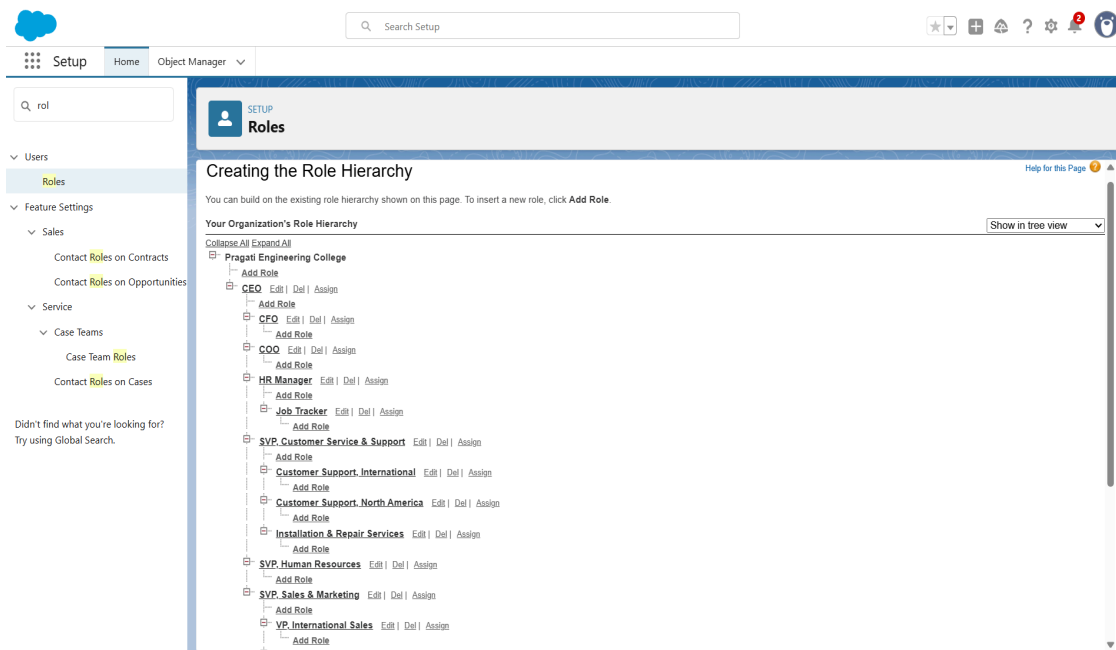
2 .Profiles

Use Case :
Profiles determine the permissions and access levels for users. HR requires full access (Create, Read, Edit, Delete) to Applications, while Job Trackers have limited access. By creating separate profiles, we enforce data security and workflow compliance.

Implementation Steps:

1. Setup → Quick Find → Profiles → Clone Standard User → Name: HR Profile.

2. Configure Object Settings for Application object → Full CRUD for HR.

3. Clone again → Job Tracker Profile → Limited CRUD.
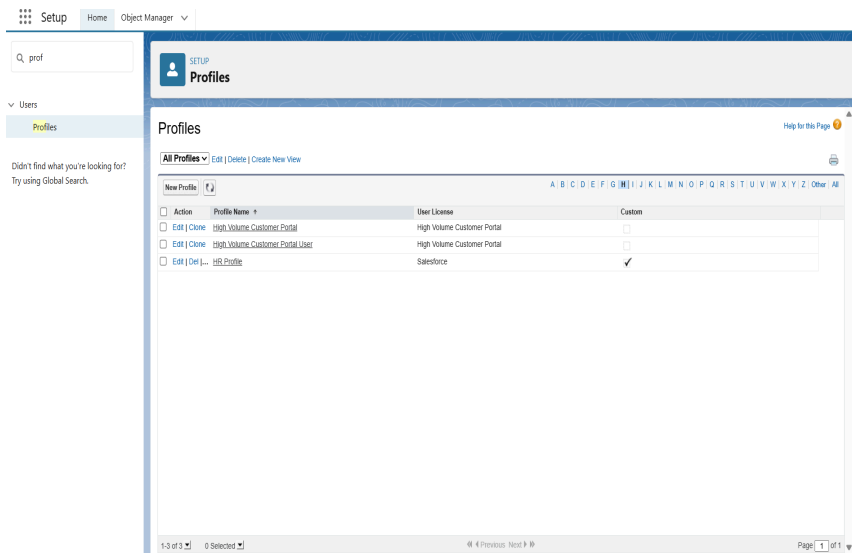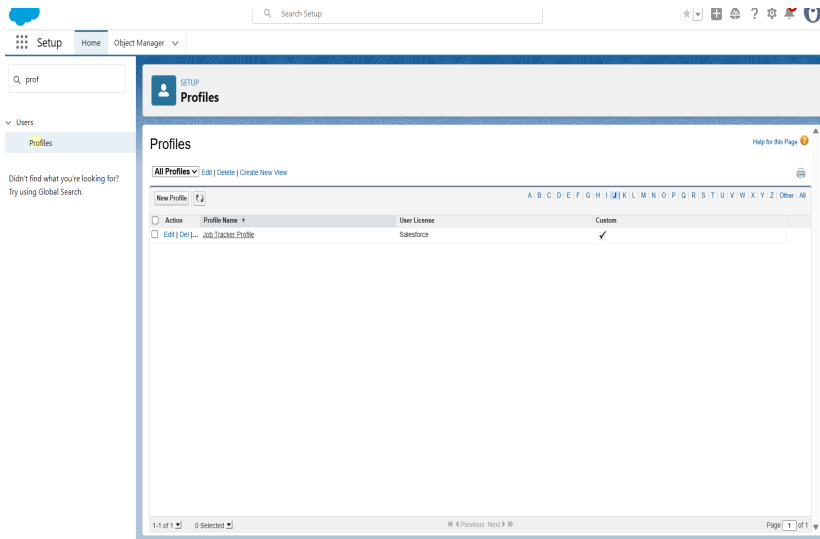
3 Sharing Rules

Use Case:
Sharing Rules allow exceptions to org-wide defaults. Since Applications are set to Private, we need to share Job Tracker records with HR Managers for visibility. This ensures HR can oversee the recruitment pipeline without violating privacy.

Implementation Steps:

1. Setup → Quick Find → Sharing Settings → Application Sharing Rules → New.

2. Rule: Share Applications with HR → Owned by Job Tracker → Share with HR Manager → Read/Write → Save.
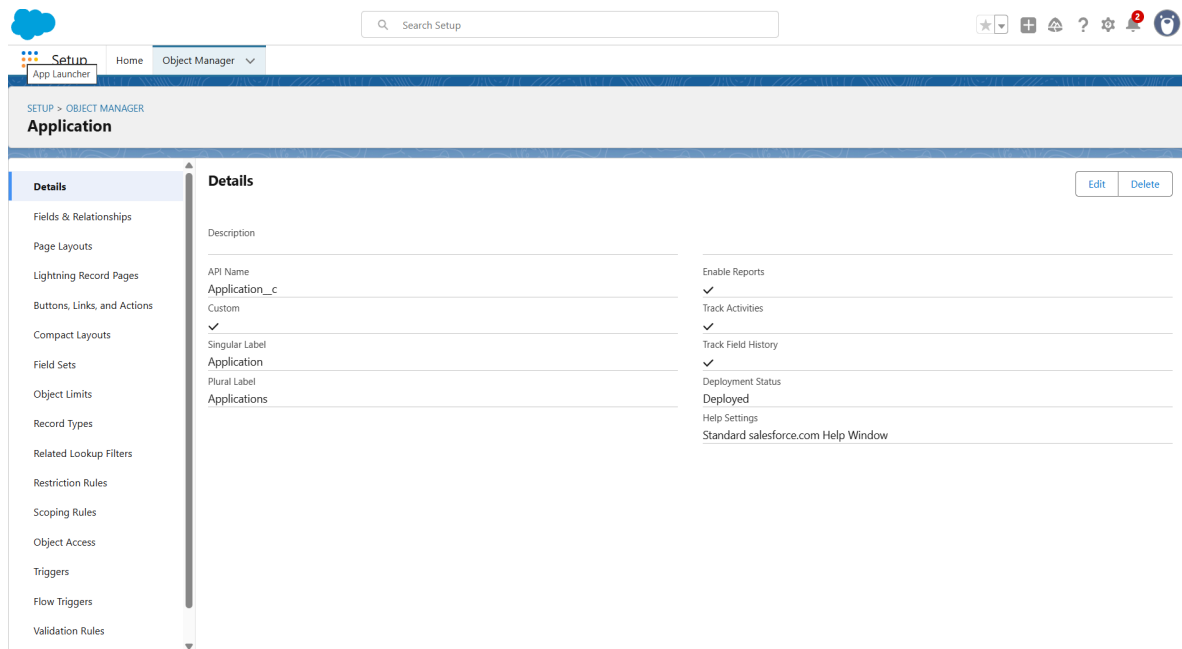

Phase 3 :Data Modeling & Relationships

1. Custom Object – Application__c

Use Case:
Central object to store all candidate application data. Acts as a single source of truth.

Implementation Steps:

- Setup → Object Manager → Create Custom Object → Application__c → Auto Number Record Name → Enable Reports & Activities.

2. Fields & Page Layouts

Use Case:

Custom fields capture necessary candidate information. Page Layouts ensure ease of data entry for users.

Implementation Steps:

- Create fields: Candidate Name, Email, Phone, Job Title, Education, Experience, Notice Period, Resume Link, Status, Applied Date, Interview Details, Eligibility Score.
- Configure Field-Level Security and add to Page Layout.

Phase 4 :Process Automation

1. Email Templates & Alerts

Use Case:
Automates communication to candidates based on application status. Improves candidate experience and reduces HR workload.

Implementation Steps:

- Create templates: Under Review, Shortlisted, Selected, Rejected, Interview Details.
- Setup Email Alerts and link to flows.

3. Flows

Use Case Explanation:
Automates default values and status changes, triggers email alerts, and evaluates eligibility.

Implementation Steps:

- After-Save Flow → decision elements for Education, Experience, Notice Period → update Status → send emails.
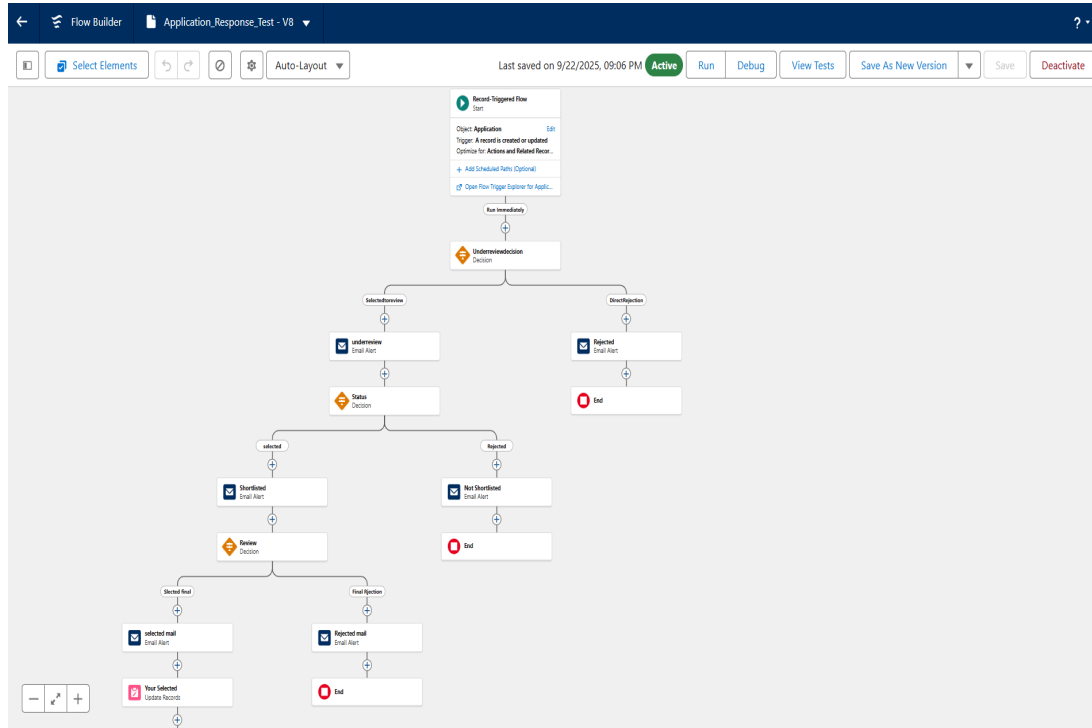
Phase 5 : Apex Programming

1.ApplicationTaskTrigger

Use Case:
Automatically creates follow-up tasks for HR to ensure timely processing of applications.

Implementation Steps:

- Trigger after insert → create Task for each application → assign to HR.

```
1    @isTest
2  ▾ public class ApplicationTaskTriggerTest21 {
3  ▾     @isTest static void testTaskCreated() {
4          // Create a sample application record
5          Application__c app = new Application__c(
6              Candidate_Name__c = 'Test Candidate',
7              Candidate_Email__c = 'test@example.com',
8              Job_Tittle__c = 'Engineer - Frontend',
9              Education_Qualification__c = 'B.Tech',
10             Experience_Level__c = 'Mid'
11         );
12         insert app;
13
14         // Verify task created
15         List<Task> tasks = [SELECT Id, Subject, WhatId FROM Task WHERE WhatId = :app.Id];
16         System.assert(tasks.size() > 0, 'Task should be created for the application');
17     }
18 }
19
```

```
1    @isTest
2  ▾ public class ApplicationTaskTriggerTest21 {
3  ▾     @isTest static void testTaskCreated() {
4          // Create a sample application record
5          Application__c app = new Application__c(
6              Candidate_Name__c = 'Test Candidate',
7              Candidate_Email__c = 'test@example.com',
8              Job_Tittle__c = 'Engineer - Frontend',
9              Education_Qualification__c = 'B.Tech',
10             Experience_Level__c = 'Mid'
11         );
12         insert app;
13
14         // Verify task created
15         List<Task> tasks = [SELECT Id, Subject, WhatId FROM Task WHERE WhatId = :app.Id];
16         System.assert(tasks.size() > 0, 'Task should be created for the application');
17     }
18 }
19
```
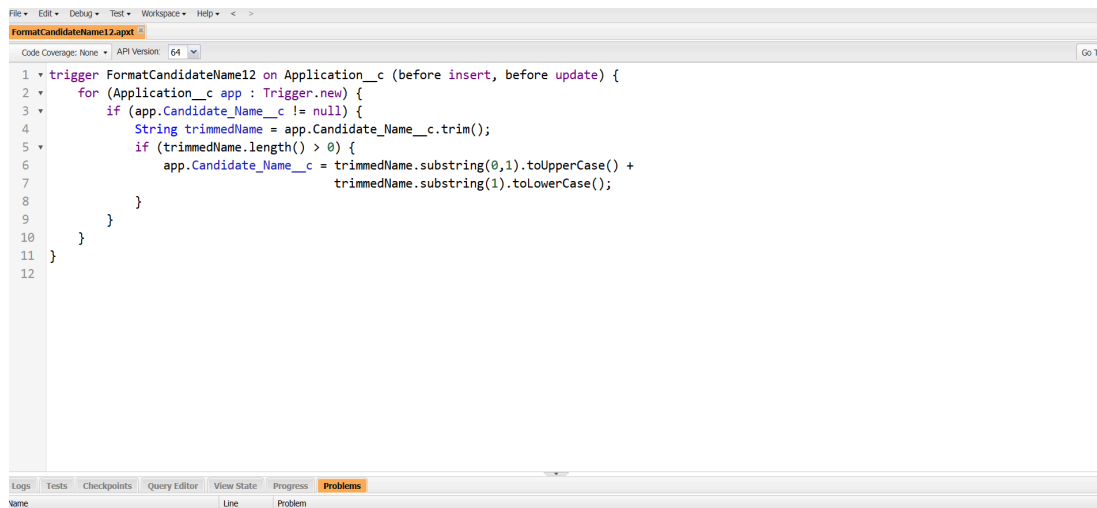
2. FormatCandidateName Trigger

Use Case:

Normalizes candidate names to proper capitalization for data consistency.

Implementation Steps:

- Trigger before insert/update → trim & capitalize first letter.

```
File ▼  Edit ▼  Debug ▼  Test ▼  Workspace ▼  Help ▼  <  >
FormatCandidateName12.apxt
Code Coverage: None ▼  API Version:  64 ▼                                              Go To
 1 ▼ trigger FormatCandidateName12 on Application__c (before insert, before update) {
 2 ▼     for (Application__c app : Trigger.new) {
 3 ▼         if (app.Candidate_Name__c != null) {
 4              String trimmedName = app.Candidate_Name__c.trim();
 5 ▼             if (trimmedName.length() > 0) {
 6                  app.Candidate_Name__c = trimmedName.substring(0,1).toUpperCase() +
 7                                          trimmedName.substring(1).toLowerCase();
 8              }
 9          }
10      }
11 }
12
Logs  Tests  Checkpoints  Query Editor  View State  Progress  Problems
Name                        Line        Problem
```

4 .Test Classes

Use Case:

Ensure code coverage ≥75% for deployable org and validate trigger/class functionality.

Implementation Steps:

- ApplicationTaskTriggerTest, EligibilityCalculatorTest → insert test data → assert expected results.

Phase 6 :Lightning App & UI Development

1.Lightning App

Use Case:

Provides an intuitive interface for HR and Job Tracker to manage applications efficiently.

Implementation Steps:

- App Manager → New Lightning App → Add Navigation Items (Applications, Reports, Dashboards, Tasks) → Assign Profiles.

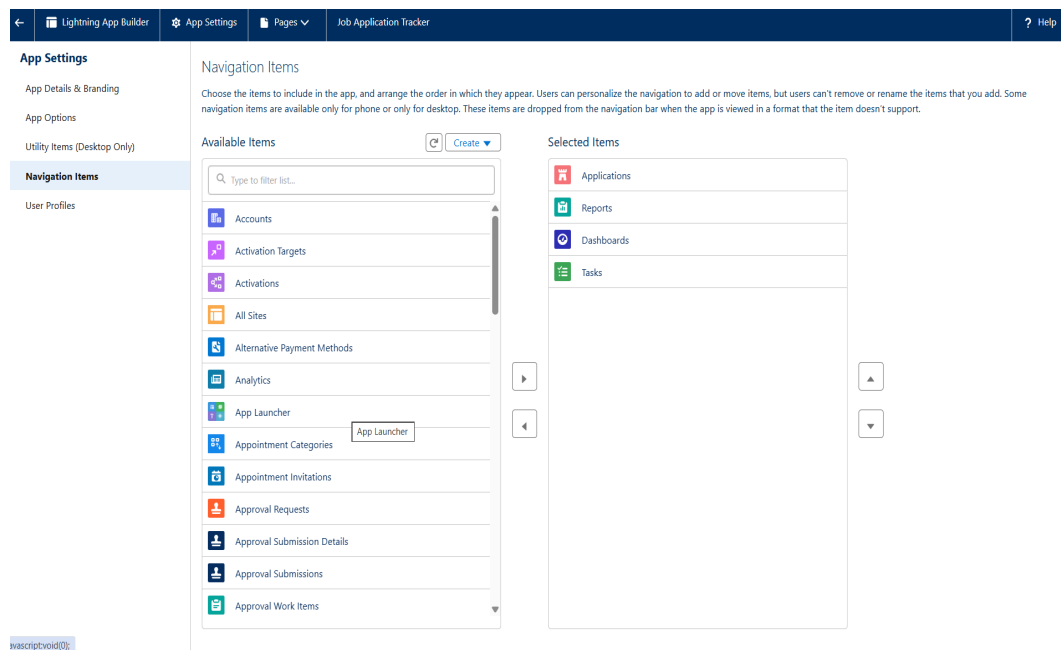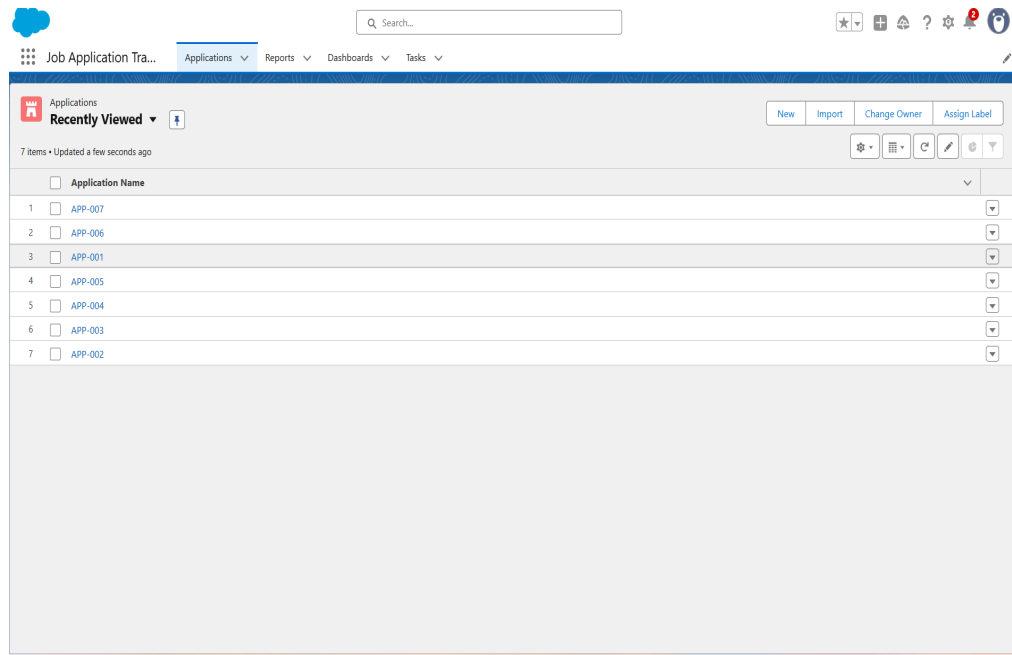## 2. Lightning Record Pages

Use Case:
Customizes layout to show only relevant fields and related lists. Improves usability.

Implementation Steps:

- Lightning App Builder → Edit Record Page → drag relevant components → assign to profiles.

Phase 7: Integration & External Access

1. Email Alerts in Flow

Use Case:

- Simulated Web-to-Application form + Email notifications.
- To mimic candidate external access (apply via form)
- Candidates apply online, HR receives automated notifications.
- Web-to-Application & Email simulation

Implementation Steps:

- Decision → HasInterviewDetails → Send Interview Details Email Alert.

2.Simulation of Integration

- Instead of heavy API/web integration, you demonstrated how Salesforce CRM can send external communication through automated emails (just like Email-to-Case or Web-to-Lead

---

Phase 8 : Data Management & Deployment

1. Create Matching Rule (defines duplicate condition)

Use Case:

● Import records, Duplicate & Matching Rules.

Implementation Steps:

● Setup → Quick Find → Matching Rules → New Rule

Matching Rule
Application_Matching_Rule

Matching Rule Detail   Delete   Clone   Deactivate

Object	Application
Rule Name	Application_Matching_Rule
Unique Name	Application_Matching_Rule
Description	
Matching Criteria	(Application: Candidate_Email EXACT MatchBlank = FALSE) AND (Application: Job_Tittle EXACT MatchBlank = FALSE)
Status	Active
Created By	Gollapalli Lakshmi Tulasi Jyothi Sri, 9/22/2025, 2:35 AM
Modified By	Gollapalli Lakshmi Tulasi Jyothi Sri, 9/22/2025, 2:36 AM

2. Duplicate Prevention

Use Case:
Prevents multiple applications from the same candidate for the same role, ensuring data integrity.

Implementation Steps:

- Matching Rule on Candidate_Email__c + Job_Title__c → Duplicate Rule → Block duplicates.

Application Duplicate Rule
Application_Duplicate_Rule

Help for this Page

Duplicate Rule Detail   Edit Delete Clone Deactivate

Rule Name	Application_Duplicate_Rule	Order	1 of 1 [Reorder]
Description			
Object	Application		
Record-Level Security	Enforce sharing rules		
Action On Create	Block	Operations On Create	☐ Alert ☐ Report
Action On Edit	Allow	Operations On Edit	✓ Alert ✓ Report
Alert Text	Use one of these records?		
Active	✓		
Matching Rule	✅ Application_Matching_Rule ✅ Mapped	Matching Criteria	(Application: Candidate_Email EXACT MatchBlank = FALSE) AND (Application: Job_Tittle EXACT MatchBlank = FALSE)
Conditions			
Created By	Gollapalli Lakshmi Tulasi Jyothi Sri, 9/22/2025, 2:40 AM	Modified By	Gollapalli Lakshmi Tulasi Jyothi Sri, 9/22/2025, 2:40 AM

Edit Delete Clone Deactivate

Application Duplicate Rule : Application_Duplicate_Rule ~ Salesforce - Developer Edition
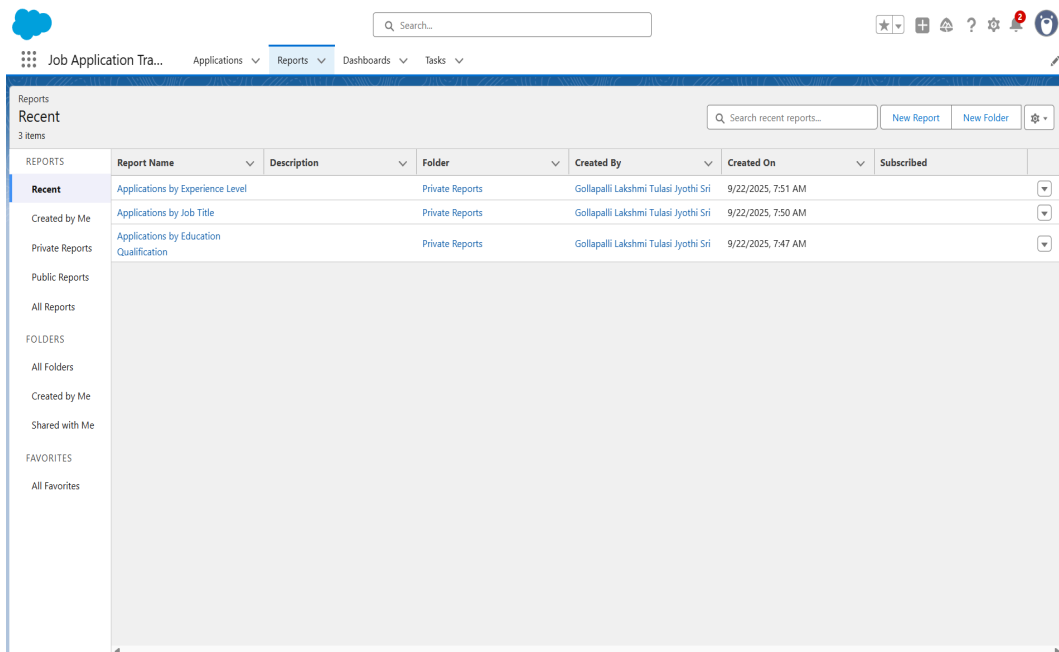

Phase 9 :Reporting & Dashboards

1. Reports

Use Case:
Provides insights on applications pipeline for HR and Hiring Managers.

Implementation Steps:

- Applications by Education → Pie Chart

- Applications by Job Title → Vertical Bar

- Applications by Experience Level → Donut Chart

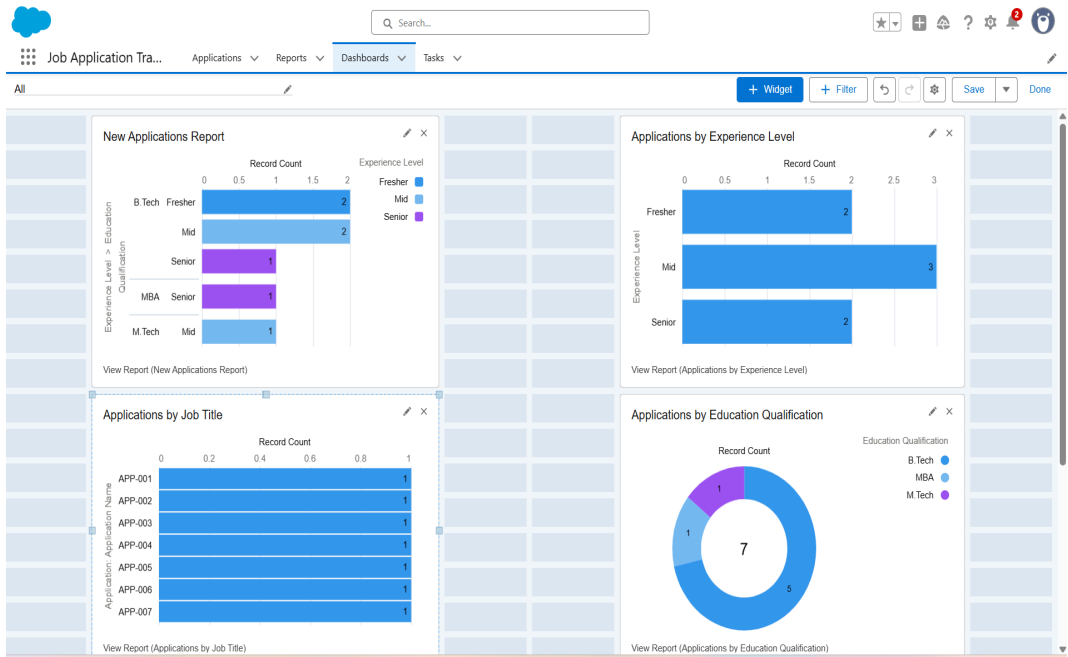# 2. Dashboard

Use Case :
Aggregates reports into a single visual dashboard for quick decision-making.

Implementation Steps:

- Dashboard → Add Components (Pie, Bar, Donut) → Resize & Save → Assign to folder.

Phase 10 : Quality Assurance Testing

1.Testing Approach:

- Functional testing performed for all Salesforce features implemented, including:

  - Record creation in Application object.

  - Flows.

  - Email Templates & Email Alerts.

  - Duplicate Prevention (Matching & Duplicate Rules).

  - Apex Triggers (ApplicationTaskTrigger,ApplicationTaskTriggerTest, FormatCandidateName).

  - Reports & Dashboards.

- Each test case includes input data, expected output, and actual output with screenshots.

- Testing was done in Developer Org using multiple users (HR Manager & Job Tracker) to validate role-based access and security.

- Regression testing ensured that new changes did not break existing functionalities.

2.Test Cases

1.Use Case / Scenario

- Application Record Creation
- Duplicate Prevention
- Email Alerts
- Flow

- Reports
- Dashboard

2.Test Steps (Input)

- Create new Application record with CandidateName, Email, Job Title, Education, Experience, Notice Period
- Attempt to create a second Application with same Candidate Email
- Record created with Education = B.Tech,Experience = MidNotice Period = Immediate
- Trigger Status Selected for a candidate
- Open Job Application Dashboard
- Run "Applications by fields" report
- Tasks should be created
- Application without Resume

3.Expected Result

- Application record created successfully.
- Record creation blocked; duplicate alert displayed
- Status updated to Shortlisted → Email alert sent to candidate
- Candidate receives Selected email with correct details
- Report displays accurate count of Applications
- Components (Pie, Bar, Donut) display correct data visually
- Tasks should be created
- Trigger when resume is blank

4.Actual Output
- Components (Pie, Bar, Donut) display correct data visually
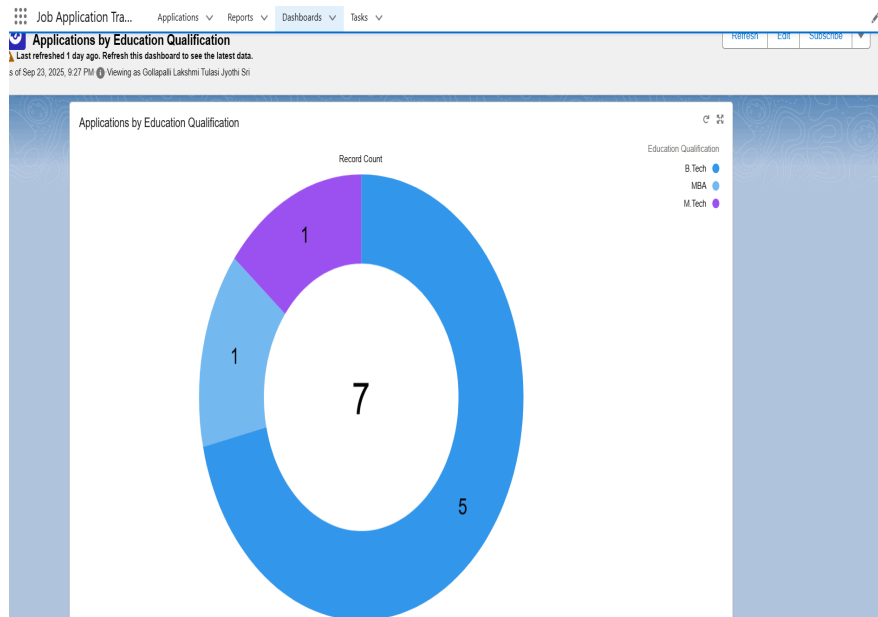
- Report displays accurate count of Applications

Report: Applications
New Applications Report

Total Records
8

Education Qualification ↑ ▼	Experience Level ↑ ▼	Notice Period ↑ ▼	Record Count
B.Tech	Fresher	Immediate	1
		5 Days	2
		Subtotal	3
	Mid	Immediate	1
		30 Days	1
		Subtotal	2
	Senior	Immediate	1
		Subtotal	1
	Subtotal		6
MBA	Senior	Immediate	1
		Subtotal	1
	Subtotal		1
M.Tech	Mid	5 Days	1
		Subtotal	1
	Subtotal		1
Total			8

● Application record created successfully.

- Status updated to Shortlisted → Email alert sent to candidate
- Record creation blocked; duplicate alert displayed

⊘ **We hit a snag.**

Review the errors on this page.
- duplicate value found: Email__c duplicates value on record with id: a01fj00000I85JWAAZ

⊘ Cancel   Save & New   Save

-
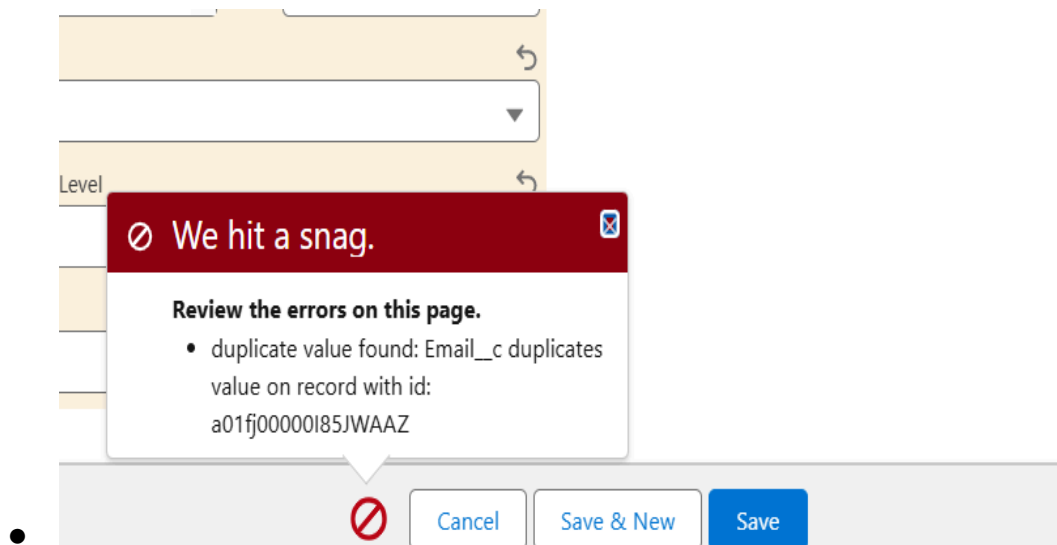
- Triggered when resume is blank

Candidate Name ↩

ashu

*Candidate Email ↩

arigelakowsalya912@gmail.com

Phone

Education Qualification ↩

B.Tech ▼

Job Title ↩

Backend ▼

Skills

Experience Level ↩

Fresher ▼

Eligibility Score

Gender

--None--

Applied Date

📅

Resume Link

Notice Period ↩

Immediate ▼

⊘ Cancel Save & New Save

3.Validation & Automation Summary

- Validation Rules: Ensured required fields (Candidate Email, Job Title) are not blank.

- Flows: Automated Status updates, Applied Date defaults, and email notifications.

- Apex Triggers: Automated task creation & candidate name formatting.

- Duplicate Rules: Block duplicate applications.

- Reports & Dashboards: Enabled HR and Hiring Manager to analyze pipeline and make decisions.

4.Conclusion

The Job Application Tracker CRM project successfully showcases how Salesforce can be customized and extended to manage the entire recruitment lifecycle in an efficient and professional manner. By combining Admin features (objects, flows, email alerts, dashboards) with Developer enhancements (Apex triggers, test classes, eligibility logic), the project delivers a complete solution that is both practical and scalable.

Advantages

1. Automation of Manual Work → Candidate status updates, email notifications, and task creation happen automatically, reducing HR's workload.

2. Data Accuracy → Duplicate prevention, validation rules, and standardized candidate name formatting ensure clean data.

3. Security & Compliance → Role hierarchy, profiles, OWD, and sharing rules protect sensitive candidate data and ensure correct access.

4. Real-Time Insights → Reports and dashboards provide instant analytics for decision-making.

5. Scalability → Built on Salesforce CRM, the solution can easily expand to include interviews, job postings, or integrations with external systems.

Highlights

- A custom Application object that centralizes all candidate information.

- Flows + Email Alerts that automate status updates (Under Review, Shortlisted, Selected, Rejected, Interview).

- Apex Triggers that auto-create HR tasks, format candidate names, and calculate eligibility scores.

- Duplicate & Matching Rules that block repeated job applications.

- Dashboards (by Status, Job Title, Experience Level) that give HR managers a 360° view of hiring.

- A secure custom app for HR and Recruiters with role-based access.

DEMO VIDEO :

https://drive.google.com/drive/folders/1z15rmmFrDJdCOIt_kqrIlMwZJZsxorGl