Project Title: SkillMatch Connect-Smart Skill-Based Student Job Recommendation System

Project Overview:

SkillMatch Connect is a Salesforce-based intelligent job recommendation system that connects students, educational institutions, and employers through skill-focused matchmaking. The platform enables students to register their profiles, showcase their skills, and receive personalized job or internship recommendations based on real-time analysis. Employers can post openings with specific skill requirements, while the system automatically maps qualified candidates using Salesforce automation and AI-powered logic. It manages the full recruitment lifecycle—from application to selection—while offering transparency, detailed analytics, and scalable integration for institutions and organizations.

Objectives

- Enable students to create profiles and register their skills, qualifications, and career interests.
- Allow companies and institutions to post job and internship listings with skill criteria.
- Automate skill-based job matching using Salesforce CRM data models and automation tools.
- Provide real-time notifications and application status updates to students via email.
- Generate analytical dashboards and reports for administrators and educational partners.
- Ensure a secure, scalable, and easily maintainable CRM environment to support continuous student-employer engagement.

Phase 1: Problem Understanding & Industry Analysis

Requirement Gathering

- Conducted sessions with NGOs, Youth Coordinators, and Employers.
- Core requirements:
    1. Youth registration with basic details and skills.
    2. Employer job postings.
    3. Auto-matching of youth ↔ jobs based on skills.
    4. Automatic interview creation and tracking.

5. Email notifications at each step.
6. Reports and dashboards for monitoring.

Scope: Digital bridge between youth, Admins and employers.

Stakeholder Analysis

- **Primary:** Youth, Employers, NGOs/Placement Coordinators.
- **Secondary:** System Administrators, Salesforce Platform.

Business Process Mapping

Current Process (Manual): Manual resume submission → NGOs connect youth → Communication gaps → No centralized reporting.
Proposed Process (Automated via Salesforce): Youth registration → Employer job postings → Auto-matching → Interviews scheduled → Emails sent → Dashboards.

Industry-Specific Use Case Analysis

- Existing job portals are generic and not NGO-specific.
- CareerBridge centralizes youth/job data, provides skill-based matching, and automated interview lifecycle.

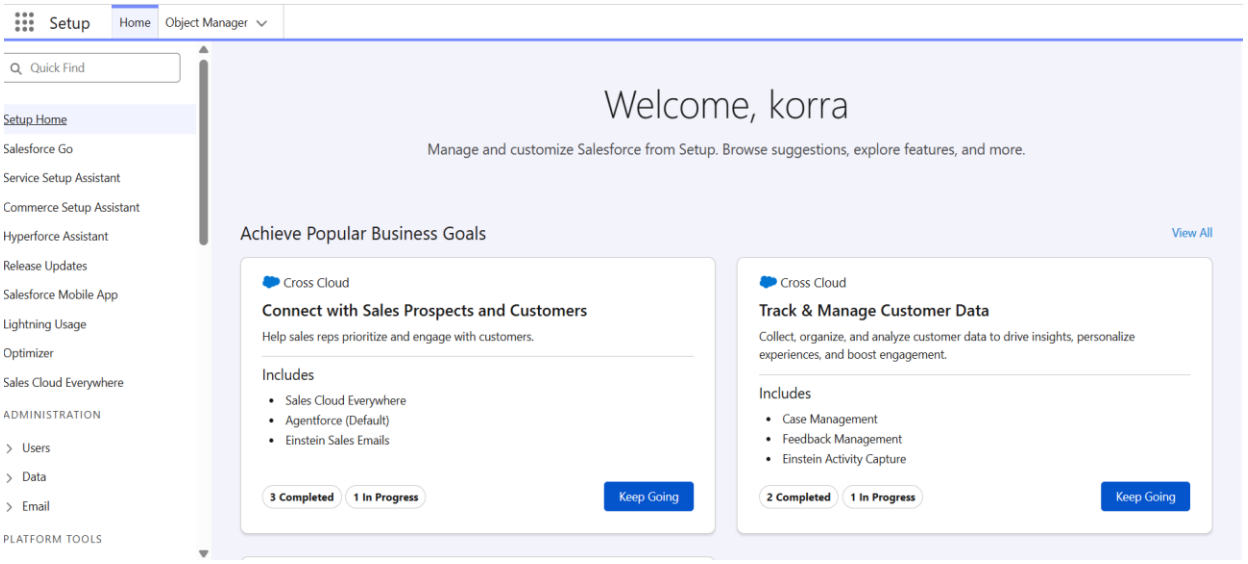
AppExchange Exploration

- Explored recruitment apps; none provide NGO-focused youth employability features.
- Decision: Build custom solution using Salesforce objects (Youth, Job, Skill, Interview), flows, and custom emails.

Phase 2: Org Setup & Configuration

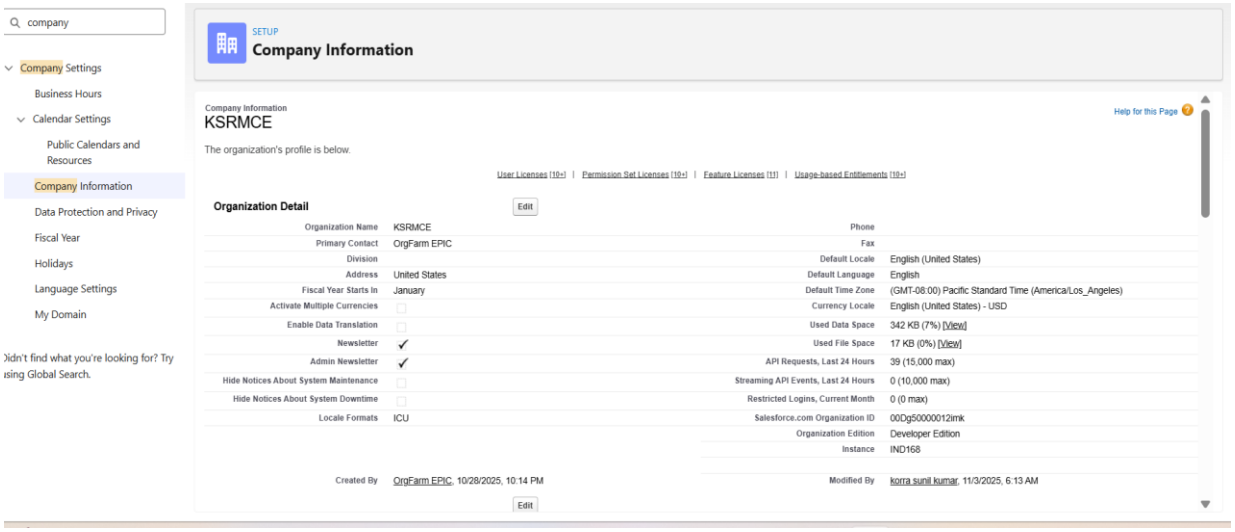
Step 1: Open Setup

1. Login to **Salesforce Lightning**.
2. Click the **Gear (☼O◦) icon** in the top-right → select **Setup**.



Step 2: Update Company Information

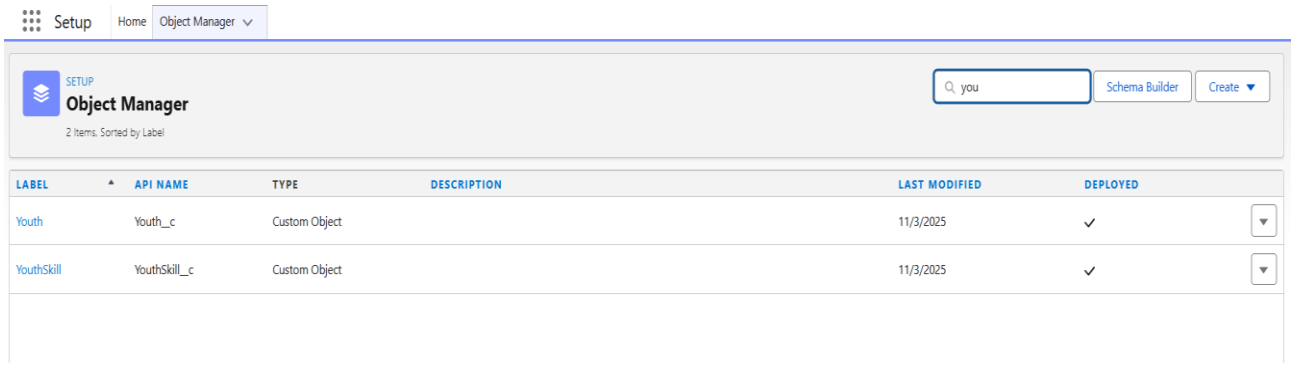
1. In Setup, use **Quick Find** → type **Company Information** → open it.
2. Click **Edit**.
3. Update:
    - **Organization Name:** KSRM College of Engineering
    - **Default Time Zone:** (09:00 – 18:00)
4. Click **Save**



Step 3: Set Business Hours

1. Quick Find → **Business Hours** → click **New**.
2. Fill in:
    - **Name:** Default Hours
    - **Hours:** 09:00 – 18:00 (or your actual business hours)
3. Click **Save**.

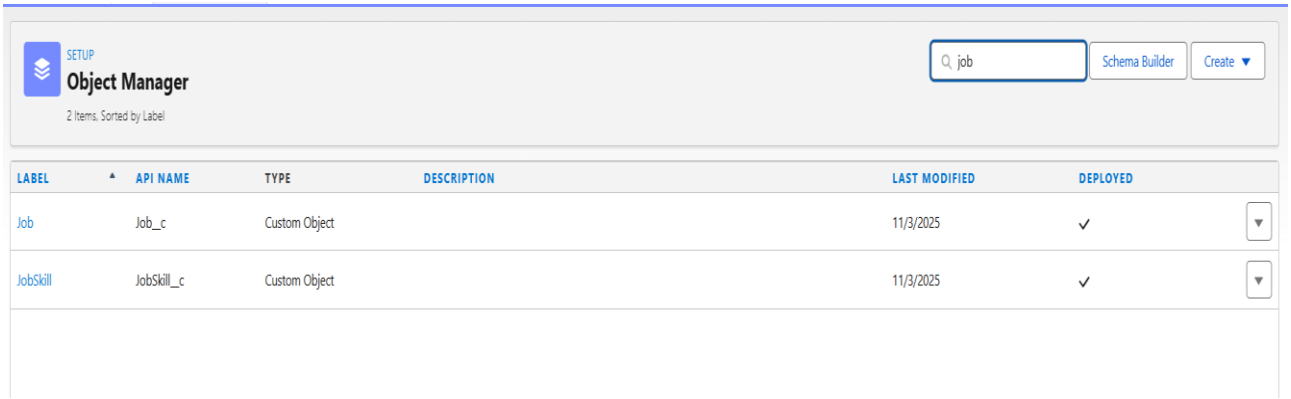
Why: Defines working hours for workflows, notifications, and approval processes.



Step 4: Add Holidays

1. Quick Find → **Holidays** → click **New**.
2. Add important dates, for example:
    - Independence Day
    - Republic Day
3. Click **Save**.

Why: Salesforce respects holidays for time-dependent automation (tasks, emails).



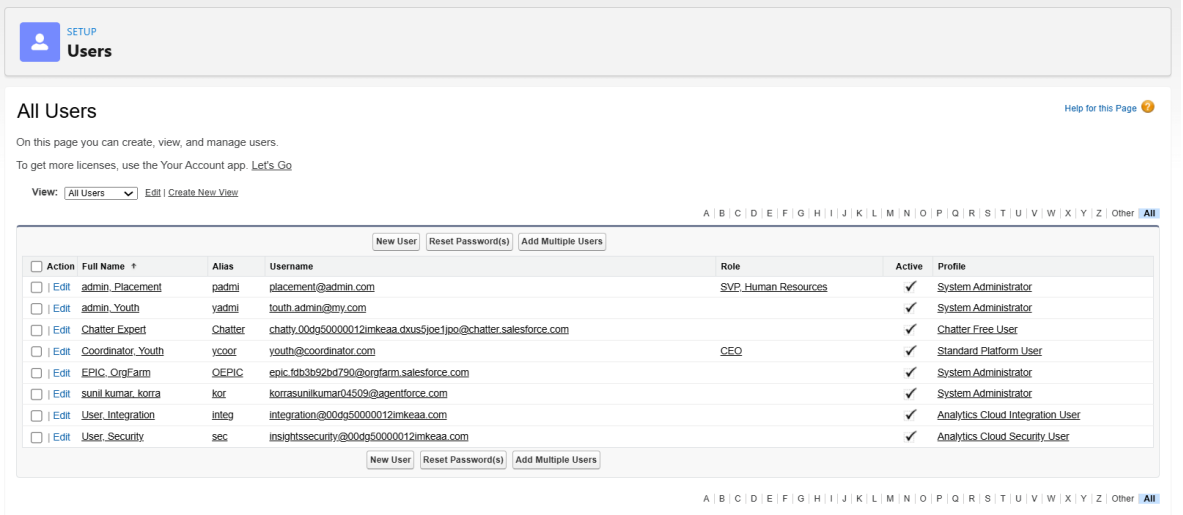
Step 5: Create Test Users

You need at least **3 users**: Youth Coordinator, Employer, NGO Admin.

1. Quick Find → **Users** → click **New User**.
2. Fill example details

Name	Username	Profile	Role
Youth Coordinator	your.email+yc@sandbox.my.salesforce.com	System Administrator	Youth Coordinator
Employer	employer@sandbox.my.salesforce.com	Standard User	Employer
NGO Admin	ngo@sandbox.my.salesforce.com	Standard User	NGO Manager

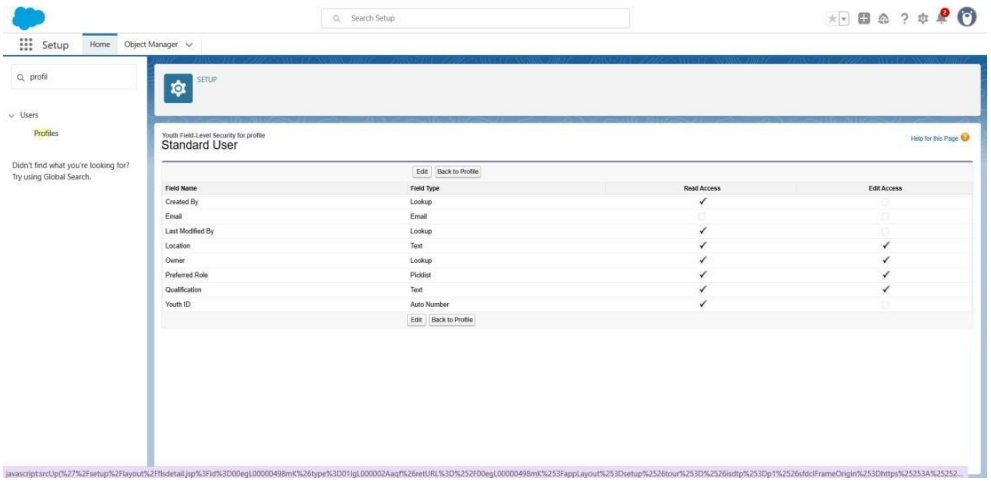
3. Click **Save** after each user.



Step 6: Configure Profiles (Object Permissions)

1. Quick Find → **Profiles** → open a profile (e.g., *Standard User*).
2. Scroll to **Object Settings** → select object Youth__c → click **Edit**.
3. Grant permissions:
    - **Read, Create, Edit** (as required)

4. Click **Save**



Step 7: Create Role Hierarchy

1. Quick Find → **Roles** → click **Set Up Roles** → **Add Role**.

2. Example hierarchy:

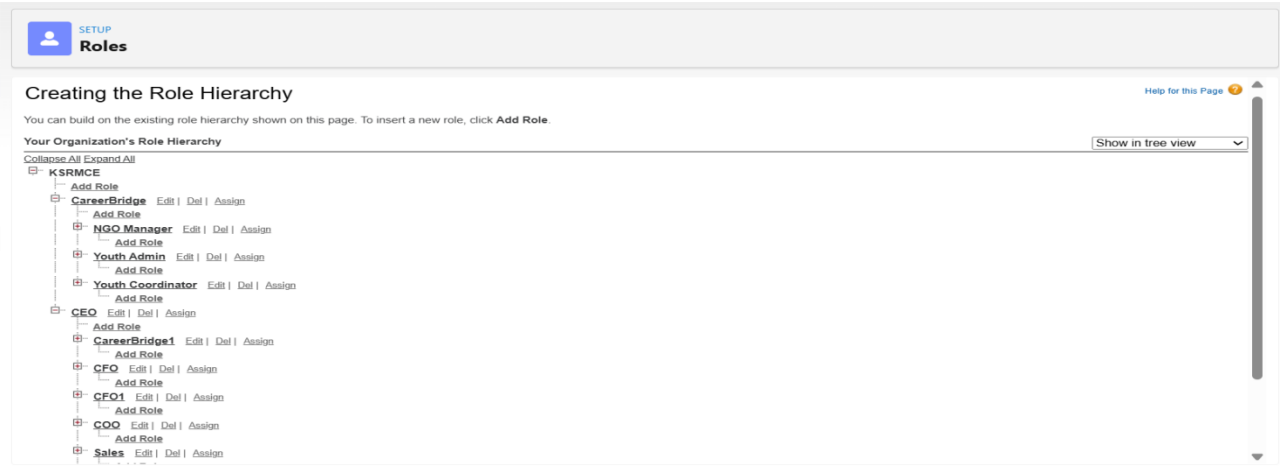
CareerBridge

├── NGO Manager

├── Youth Coordinator

└── Recruiter

3. Click **Save**.



Step 8: Permission Sets

1. Quick Find → **Permission Sets** → click **New**.
2. Label: Interview Scheduler PS → **Save**.
3. Click **Manage Assignments** → assign to users who need extra permissions.

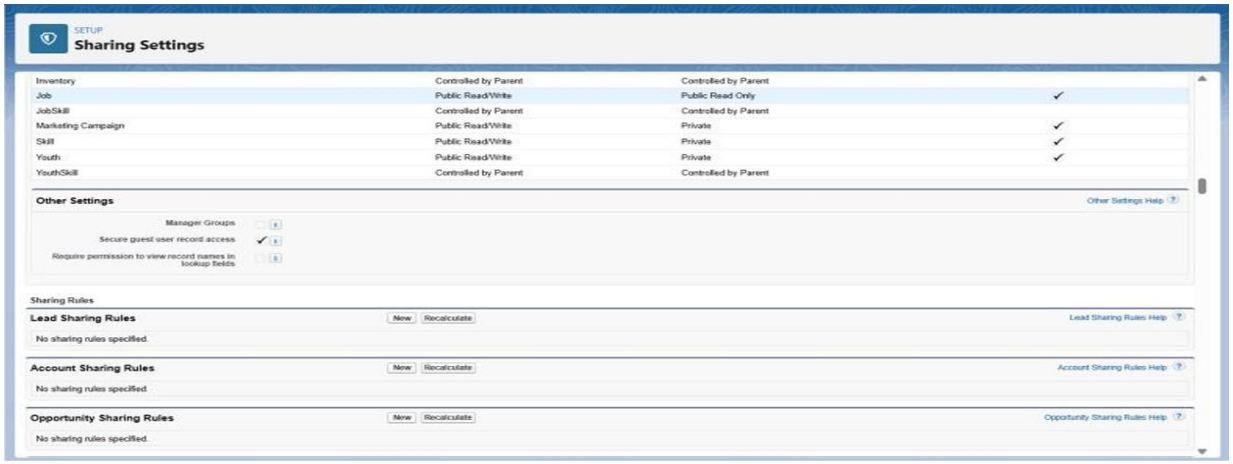
Why: Grants additional permissions without changing profiles.

Step 9: Set Org-Wide Defaults (OWD) & Sharing

1. Quick Find → **Sharing Settings** → click **Edit**.
2. Set defaults:

Object	Default Access
Youth__c	Private
Job_c	Public Read Only
Interview_c	Private

1. Create a **Sharing Rule**:
    - Scroll to **Youth__c** → click **New**.
    - Name: Share Youth to Recruiters
    - Rule Type: Owner-based
    - Shared To: Role → Recruiter
    - Click **Save.**



Step 10: Enable Admin Login Access

1. Quick Find → **Users** → select a user → click the down-arrow → **Login.**
2. If option missing: Setup → **Login Access Policies** → enable.

Why: Admins can log in as other users for testing.

Phase 3: Data Modeling & Relationships

This phase covers creating custom objects, fields, relationships, layouts, and adding sample data.

Step 1: Create Custom Objects

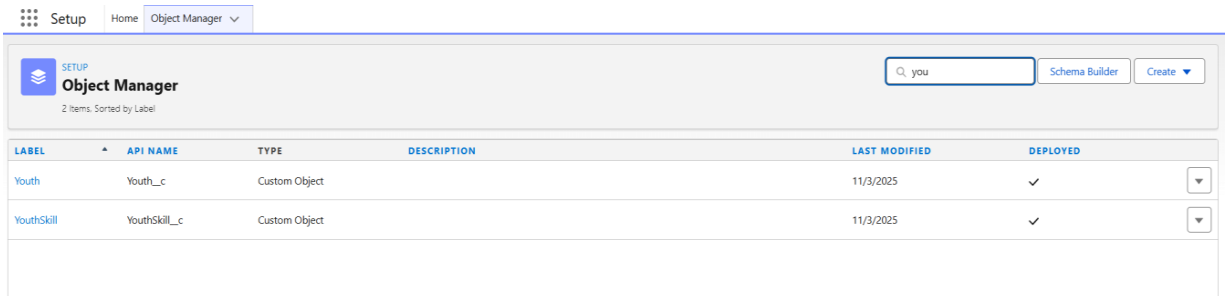
Navigation:

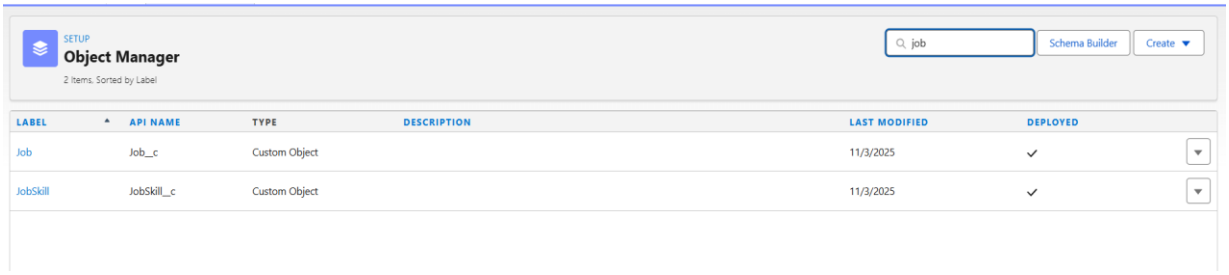
Setup → Object Manager → Create → Custom Object

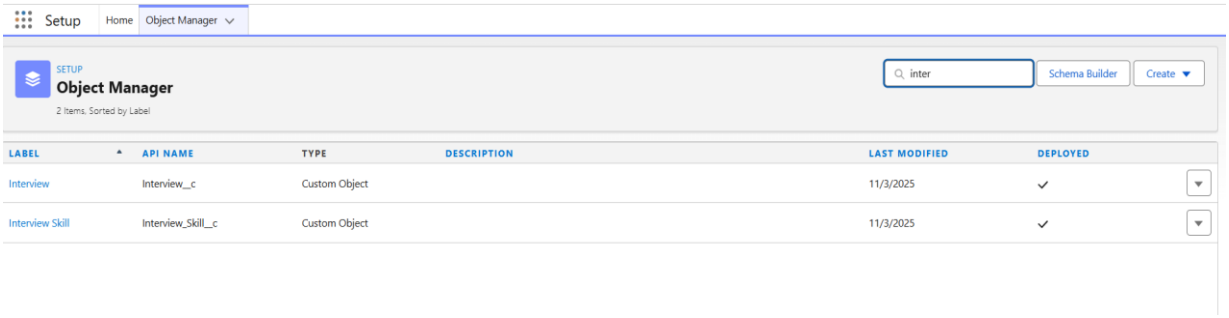
Objects to create:

Object Name	API Name	Purpose
Youth	Youth_c	Tracks youth registration info
Job	Job_c	Tracks job postings

Skill	Skill_c	Stores skill names like Java, Python
Interview	Interview_c	Stores interview details
YouthSkill	YouthSkill_c	Links Youth → Skill (Master-Detail on Youth, Lookup to Skill)
JobSkill	JobSkill_c	Links Job → Skill (Master-Detail on Job, Lookup to Skill)







Step 2: Create Fields

Each object needs fields to capture record details.

Youth__c Fields

Field Name	Type	Notes
Email_c	Email	Email address of youth
Location_c	Text(255)	City/region
Qualification__c	Text	Education level

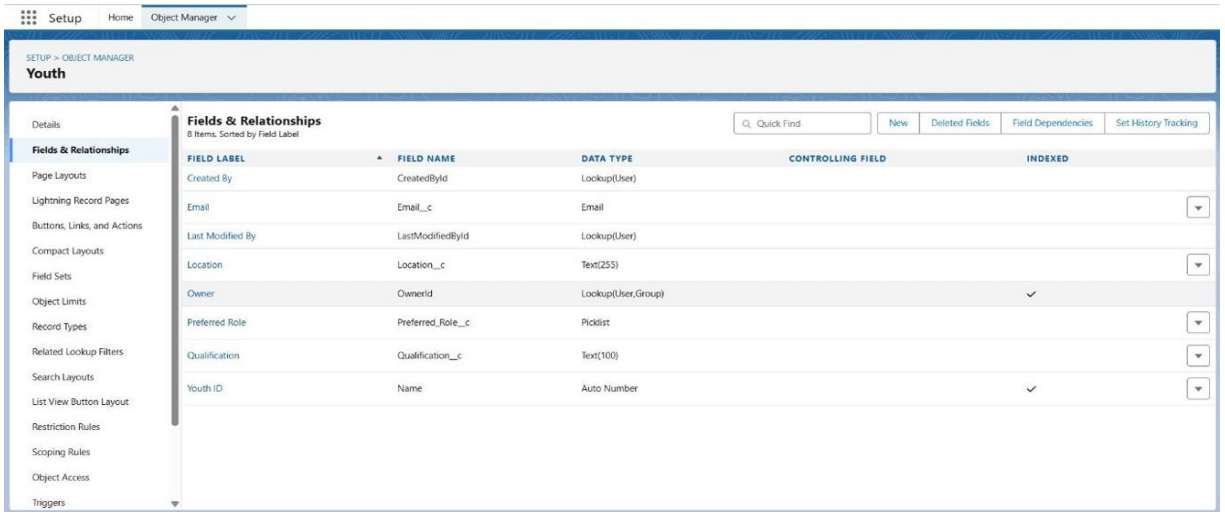
Preferred_Role__c Picklist Options: Developer, Designer, Sales

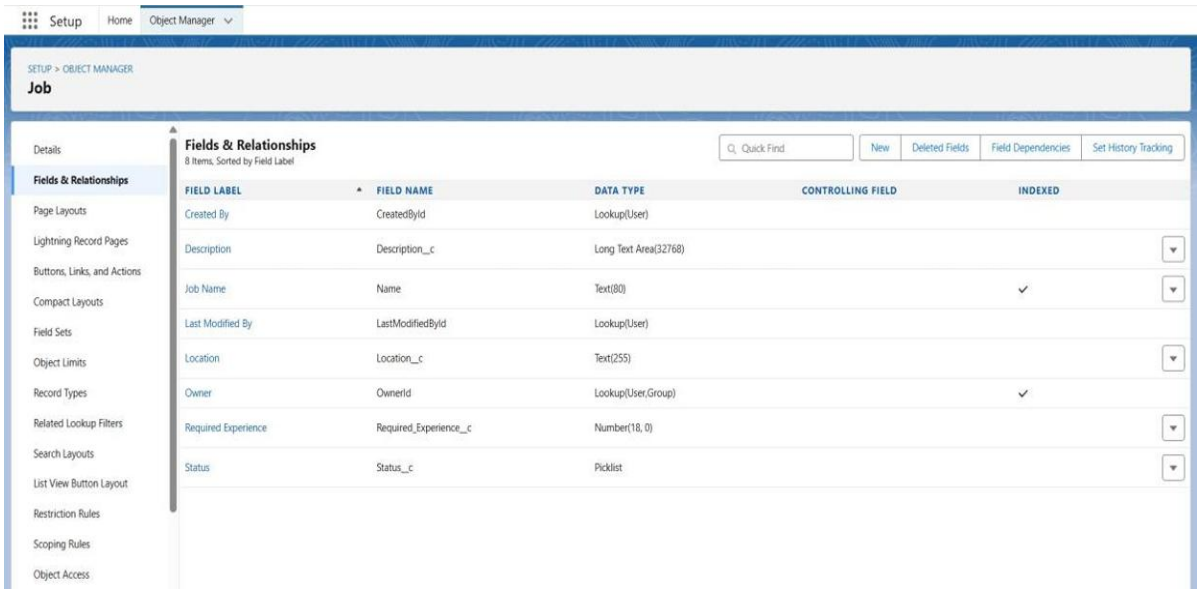
Job__c Fields

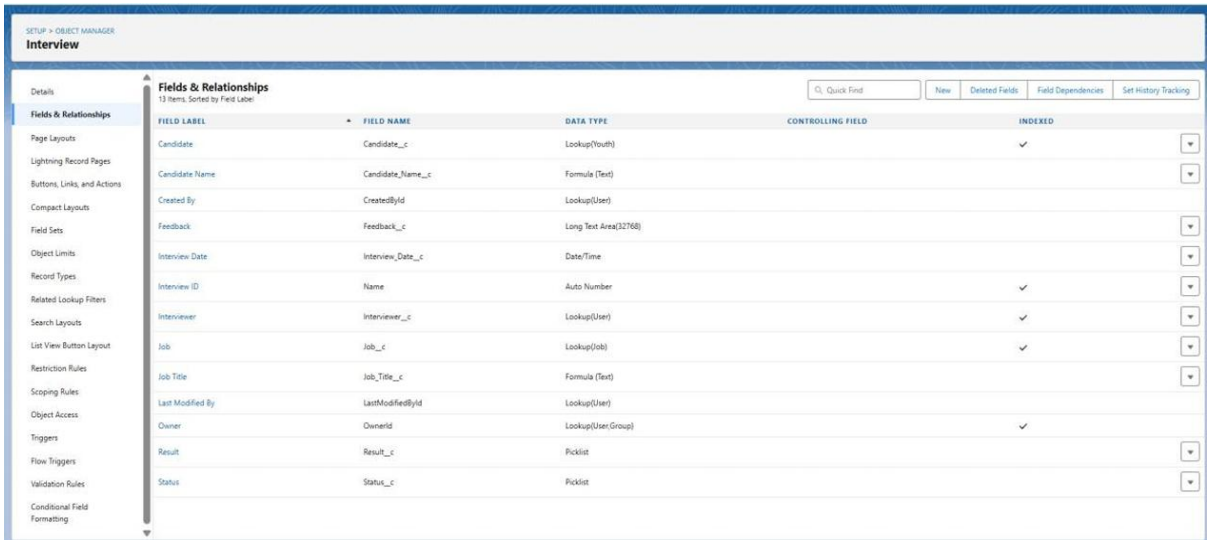
Field Name	Type	Notes
Location__c	Text(255)	City/region of job
Description__c	Long Text	Job description
Required_Experience__c	Number/Text	Experience in years
Status__c	Picklist	Options: Open, Closed

Interview__c Fields

Field Name	Type	Notes
Interview_Date__c	DateTime	Scheduled date/time
Status__c	Picklist	Scheduled, Completed, Cancelled, Feedback_Given
Candidate_c	Lookup → Youth_c	Links to the youth being interviewed
Job_c	Lookup → Job_c	Links to the job
Interviewer_c	Lookup → User	Recruiter/interviewer
Feedback_c	Long Text Area	Interview feedback







Step 3:Create Relationships

YouthSkill_c

- Master-Detail → Youth__c
- Lookup → Skill__c

JobSkill__c

- Master-Detail → Job__c

- Lookup → Skill__c

Step 4 : Configure Page

Layouts Youth_c Layout

- Fields: Email, Location, Qualification, Preferred Role
- Related Lists: YouthSkill__c, Interview__c

Job__c Layout

- Fields: Location, Description, Required Experience, Status
- Related Lists: JobSkill__c, Interview__c

Interview__c Layout

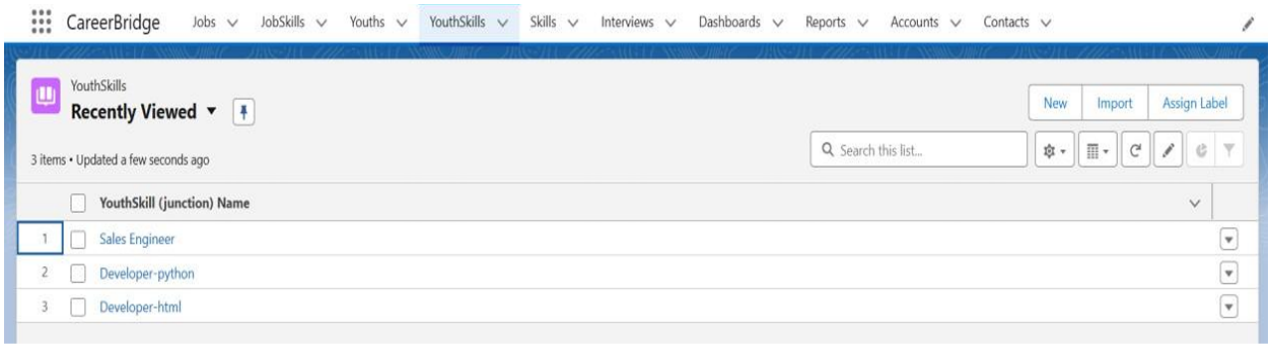
- Fields: Candidate, Job, Interview Date, Interviewer, Status, Feedback

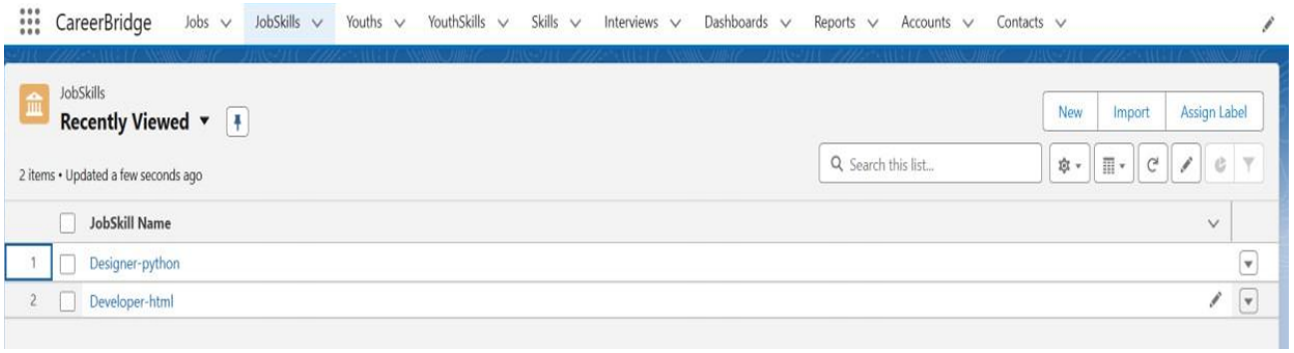
Step 5 : Add Sample Records

- **Skills:** Java, Python, HTML
- **Youth:** Test Youth → Email: test@y.com, Location: Hyderabad
- **Job:** Java Developer → Location: Hyderabad, Status: Open

Link YouthSkill & JobSkill

- Test Youth → Java (YouthSkill)
- Java Developer Job → Java (JobSkill)


CareerBridge
Jobs
JobSkills
Youths
YouthSkills
Skills
Interviews
Dashboards
Reports
Accounts
Contacts
YouthSkills
Recently Viewed
New
Import
Assign Label
3 items • Updated a few seconds ago
Search this list...
YouthSkill (junction) Name
1
Sales Engineer
2
Developer-python
3
Developer-html


CareerBridge
Jobs
JobSkills
Youths
YouthSkills
Skills
Interviews
Dashboards
Reports
Accounts
Contacts
JobSkills
Recently Viewed
New
Import
Assign Label
2 items • Updated a few seconds ago
Search this list...
JobSkill Name
1
Designer-python
2
Developer-html

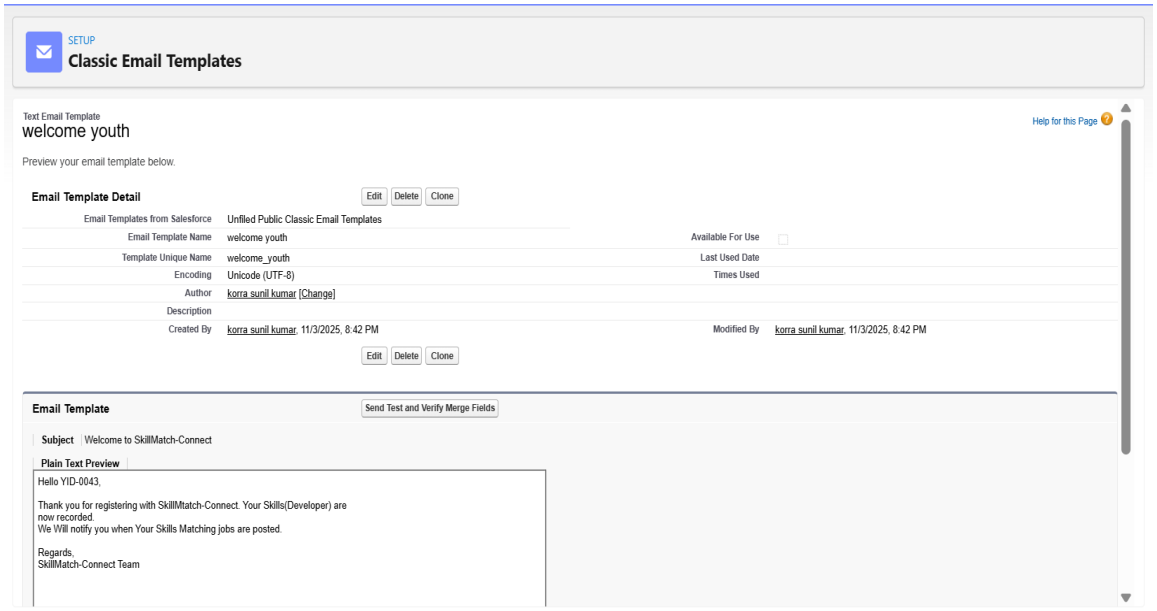
Phase 4: Process Automation (Admin)

Goal: Automate youth–job matching, interview scheduling, and email notifications, while ensuring data integrity.

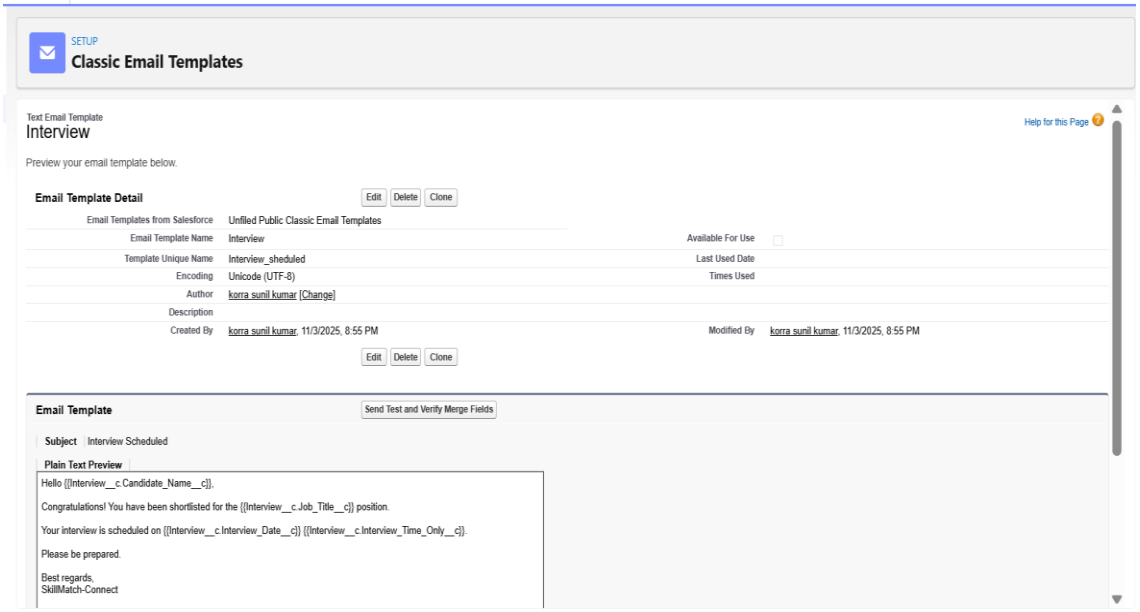
Step 1: Create Email Templates

1. Click **Setup (☼O◦)** → In Quick Find, type **Email Templates** → Select **Email Templates**.
2. Click **New Email Template** → Select **Lightning Email Template**.
3. Fill details like **Name** and **Subject**, then design the email body.
4. Save the template.

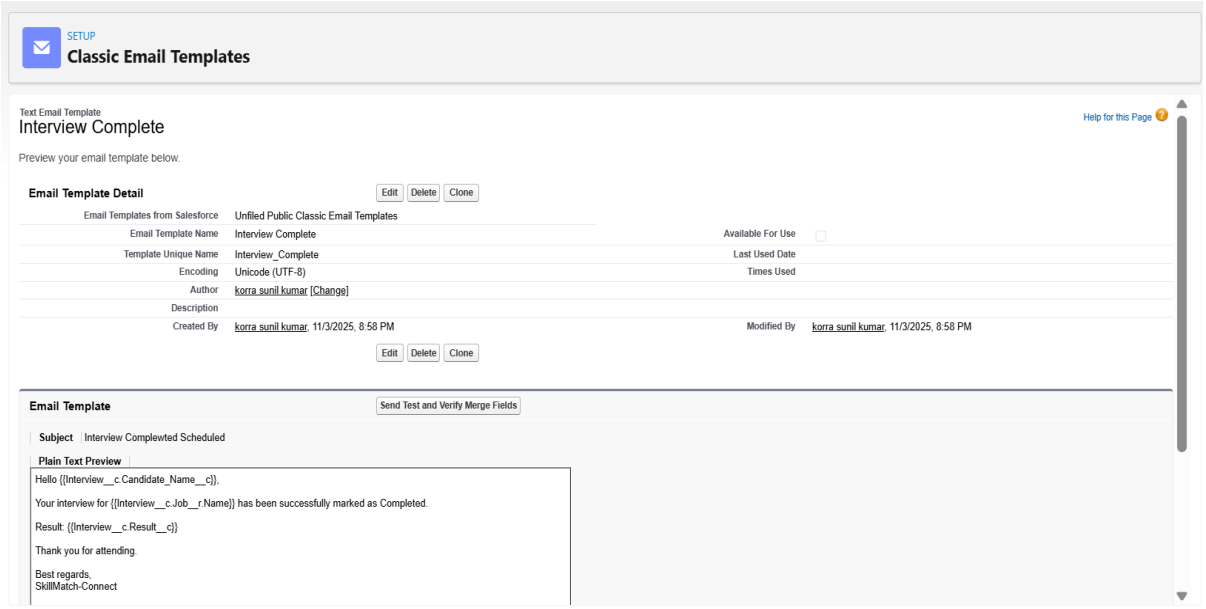
- **Welcome Email** → Sent after youth registration.



- **Interview Scheduled Email** → Sent after youth is matched with a job and interview is created.



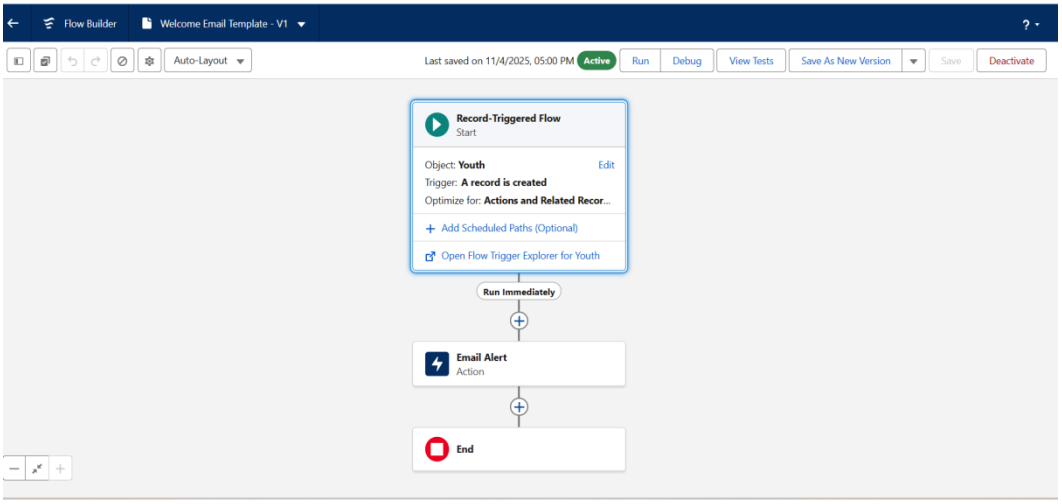
- **Interview Completed Email** → Sent after interview status is marked as Completed.



Step 2: Build Flows

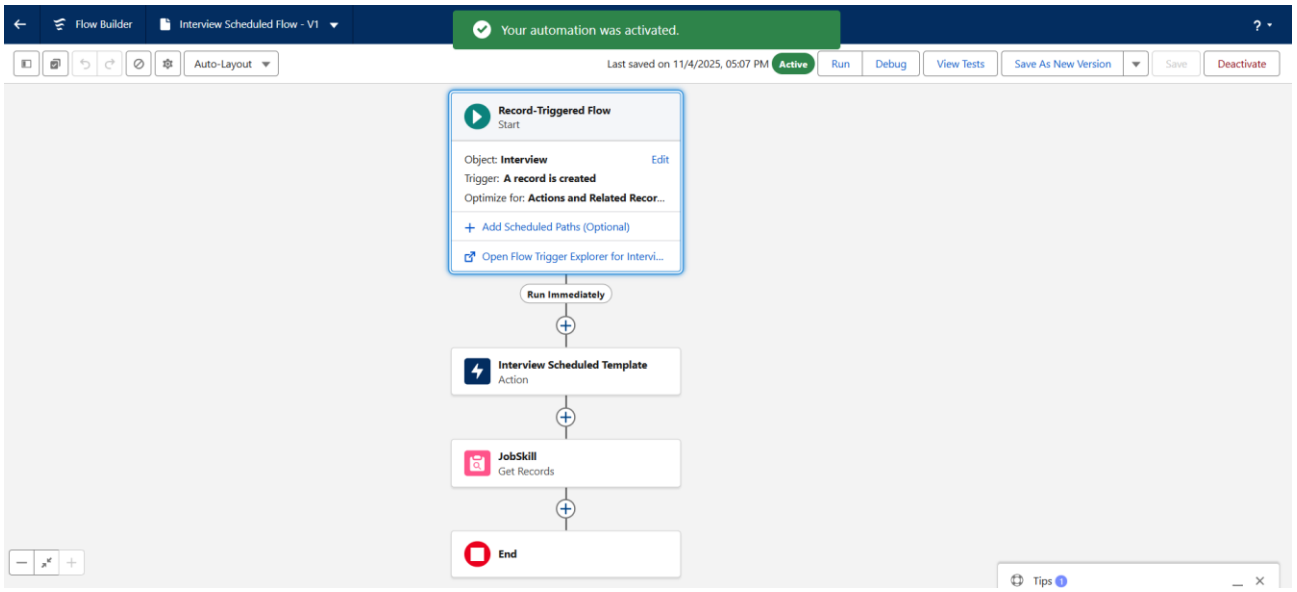
1. Welcome Email Flow (After Registration)

1. Click **Setup** ⚙ → Quick Find → **Flows** → **New Flow**.
2. Select **Record-Triggered Flow**.
3. Choose **Object: Youth__c** → Trigger: **When record is created → After Save**.
4. Add **Action → Send Email** → Select **Welcome Email Template**.
5. Save and Activate.



2. Interview Scheduled Flow (After Match & Scheduling)

1. Click **Setup → Flows → New Flow**.
2. Select **Record-Triggered Flow → Object: Job_c** → Trigger: **Created or Updated → After Save**.
3. Add **Get Records → JobSkill__c** (fetch required skills).
4. Add **Get Records → YouthSkill__c** (find youth with matching skills).
5. Add **Loop** → For each Youth, **Create Interview_c record** (Candidate, Job, Status = Scheduled, Date).
6. Add **Action → Send Email** → Select **Interview Scheduled Template**.
   Save and Activate

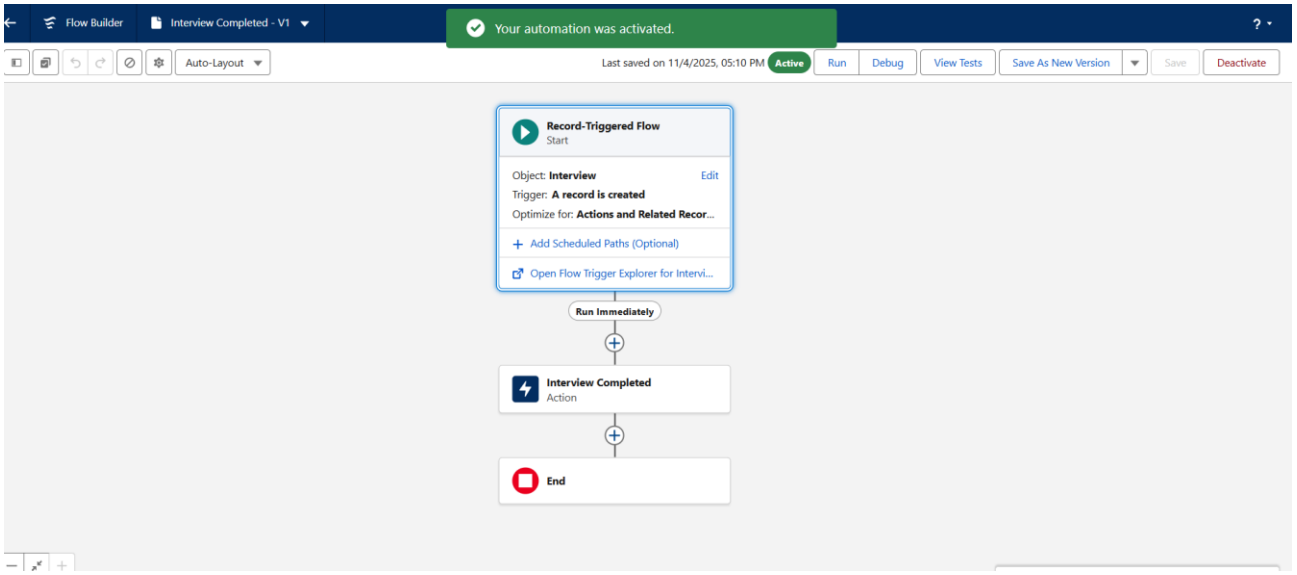


3. Interview Completed Flow (After Interview Completion)

6. Click **Setup → Flows → New Flow**.
7. Select **Record-Triggered Flow → Object: Interview__c**.
8. Trigger: **When record is updated → After Save**.
9. Add **Decision Element** → Condition: Status__c = Completed.

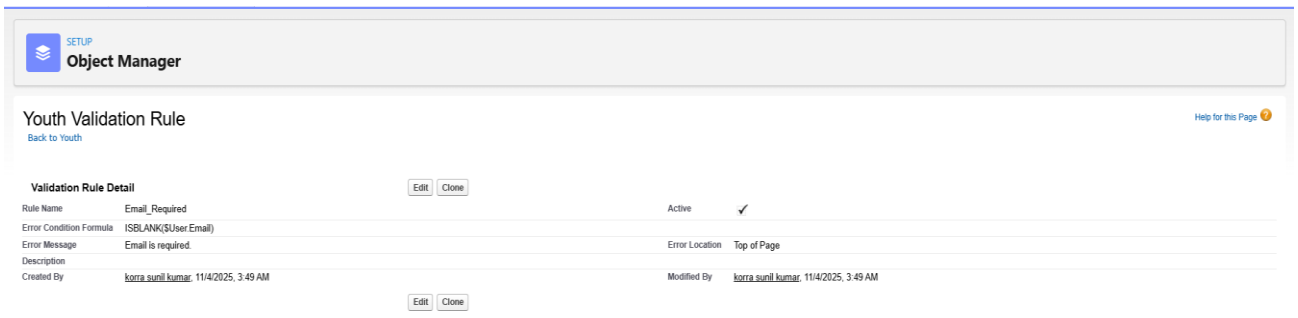
10. Add **Action → Send Email** → Select **Interview Completed Template**.

11. Save and Activate.

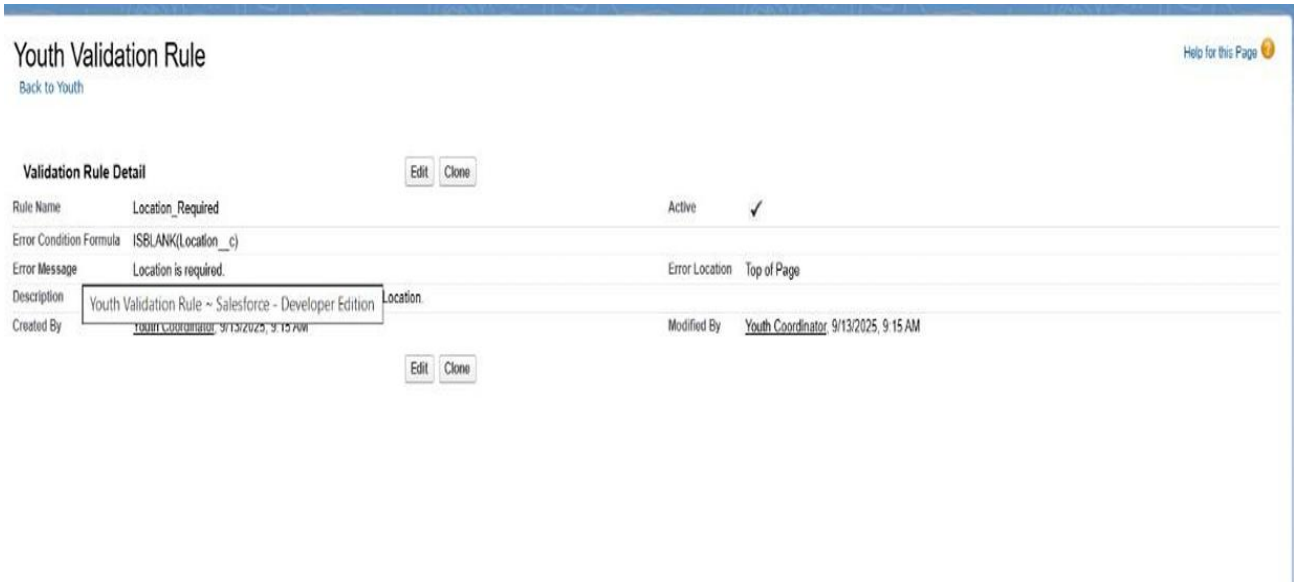


Step 3: Add Validation Rules

1. Click **Setup → Object Manager → Youth__c → Validation Rules → New**.
2. Create Rule: **Email Required**
    - Formula: ISBLANK(Email__c)
    - Error: “Email is required.”



3. Create Rule: **Location Required**
    - Formula: ISBLANK(Location__c)
    - Error: “Location is required.”



Phase 5: Apex Programming (Developer)

Goal:

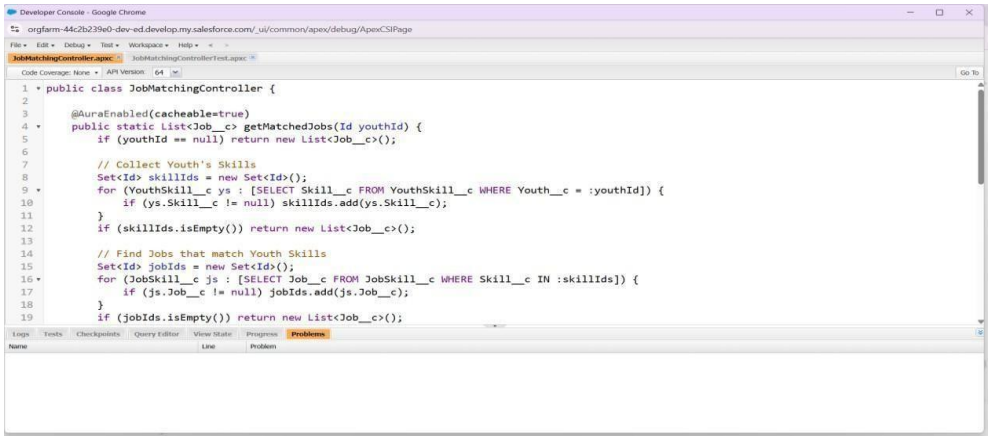
- Match youth to jobs based on skills using Apex.
- Automatically create Interview__c records.
- Handle bulk data safely and meet Salesforce deployment standards.

Step 1: Apex Class — JobMatchingController

Purpose: Perform matching and create interviews programmatically.

Steps to create Apex Class:

1. Click **Setup → Quick Find → Apex Classes → New**.
2. Copy and paste the JobMatchingController code below.
3. Click **Save**



Code Logic Overview:

- **Method 1: getMatchedJobs(youthId)**
    - Fetch youth skills.
    - Find jobs requiring those skills.
    - Return matched jobs (up to 100 records).

Method 2: createInterview(youthId, jobId, interviewDateIso, interviewerId)

- Converts ISO date string → Datetime.
- Creates Interview__c record with Status__c = Scheduled.
- Returns the created interview Id.

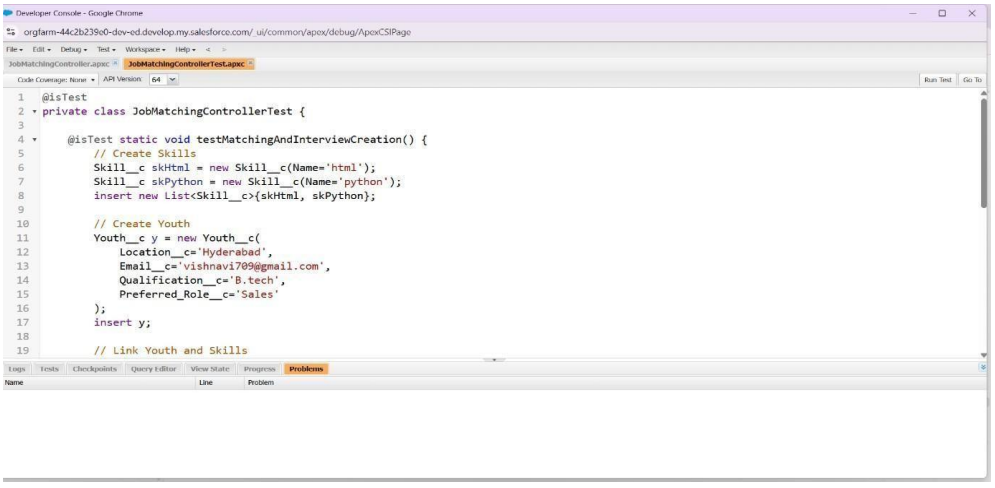
code works for these tasks. **Small improvement:** In AutoInterviewScheduler, query COUNT() inside a loop — for very large data, consider bulkifying (collect all youth + jobs, then insert interviews in one DML).

Step 2: Apex Test Class — JobMatchingControllerTest

Purpose: Ensure at least 75% coverage and test functionality.

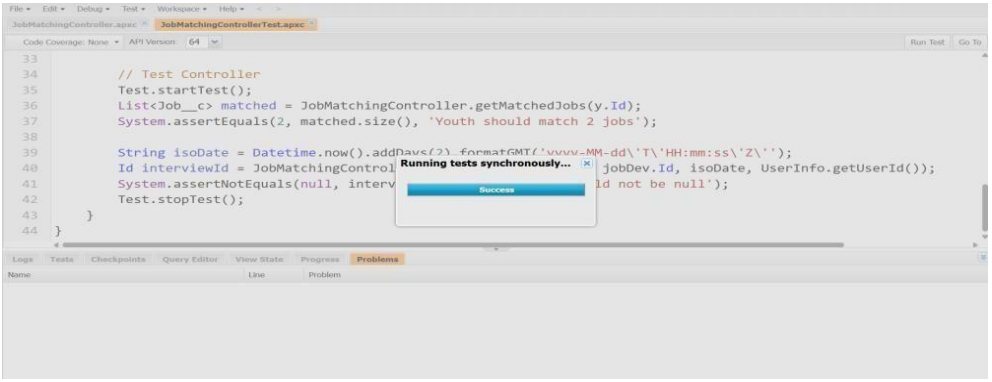
Steps to create Test Class:

1. Click **Setup → Apex Classes → New**.
2. Copy and paste the JobMatchingControllerTest code below.
3. Click **Save**.



Test Logic Overview:

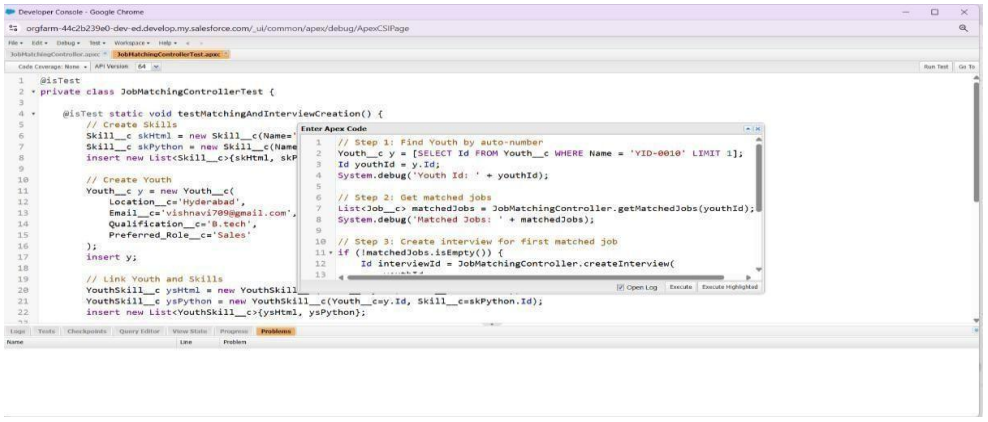
- Creates test Skill__c, Youth__c, YouthSkill__c, Job__c, JobSkill__c.
- Tests getMatchedJobs() → asserts correct number of jobs.
- Tests createInterview() → asserts interview Id is not null



Step 3: Execute Apex for Individual Youth

Steps:

1. Open **Developer Console → Execute Anonymous Window**.
2. Run this snippet (replace YID-0010 with actual youth auto-number):



This works — it will create an interview for the first matched job

Step 4: Scheduler — AutoInterviewScheduler

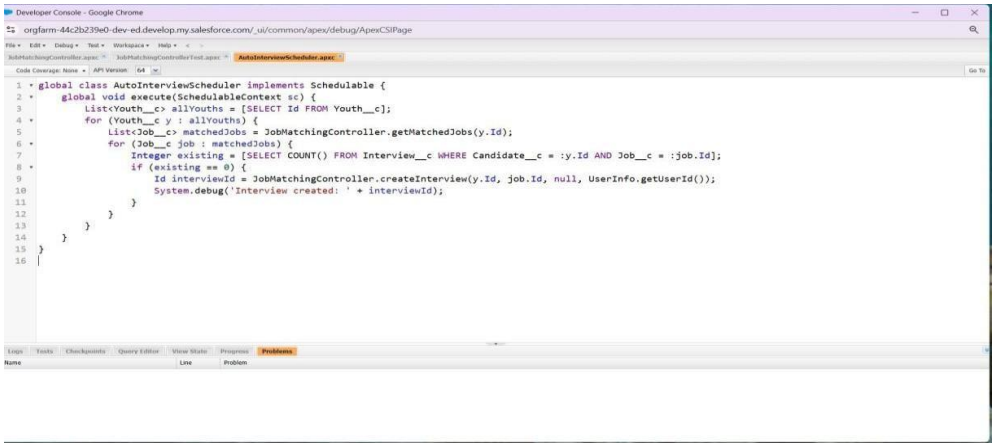
Purpose: Automatically create interviews for all youth. **Steps to schedule:**

1. **Click Setup → Apex Classes → Schedule Apex → New.**

2. Choose **AutoInterviewScheduler** class.

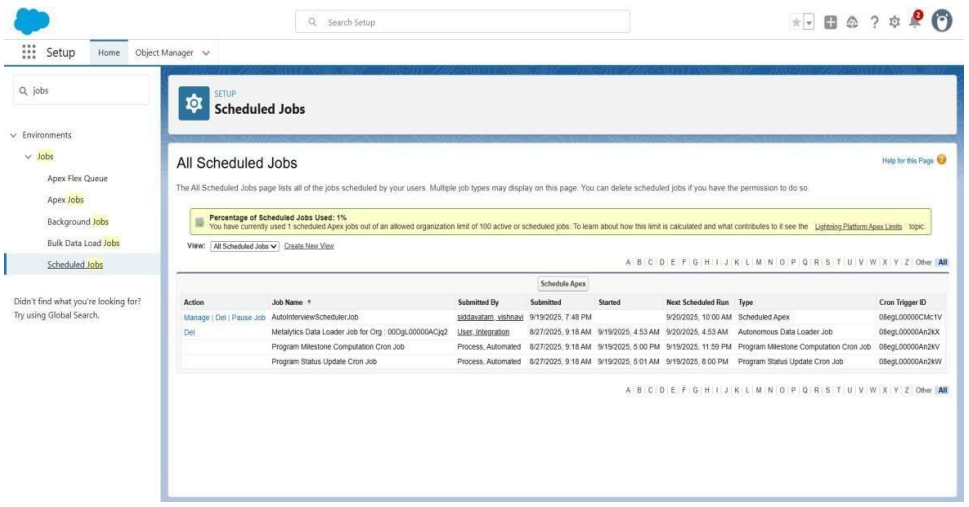
3. Set frequency (daily/hourly) and start/end dates.

4. Click **Save** to activate scheduler.



Logic:

- Loops through all youth → gets matched jobs → creates interview if none exists.
- System.debug() logs each creation.

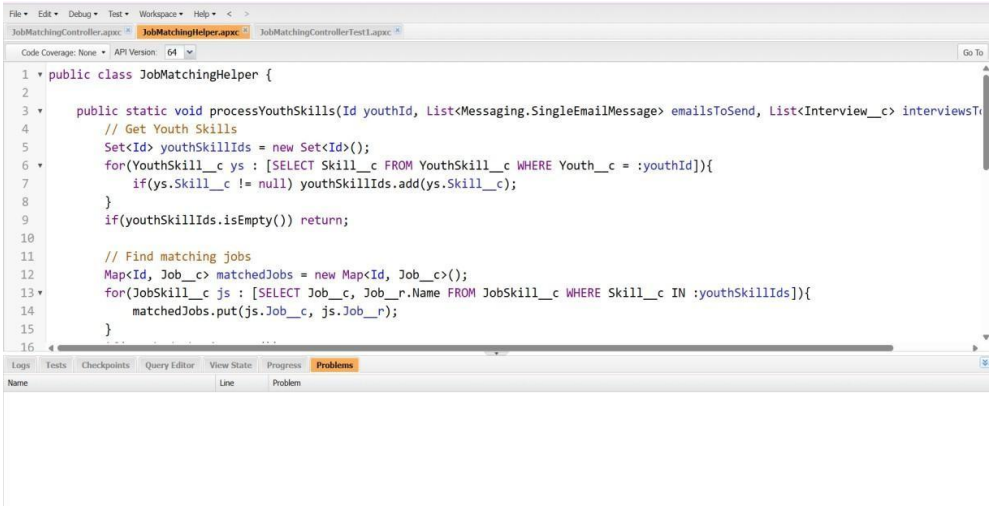


Step 5: Apex Helper Class — JobMatchingHelper

Purpose Handle automatic interview creation and email sending when youth skills match jobs.

Steps to create Helper Class:

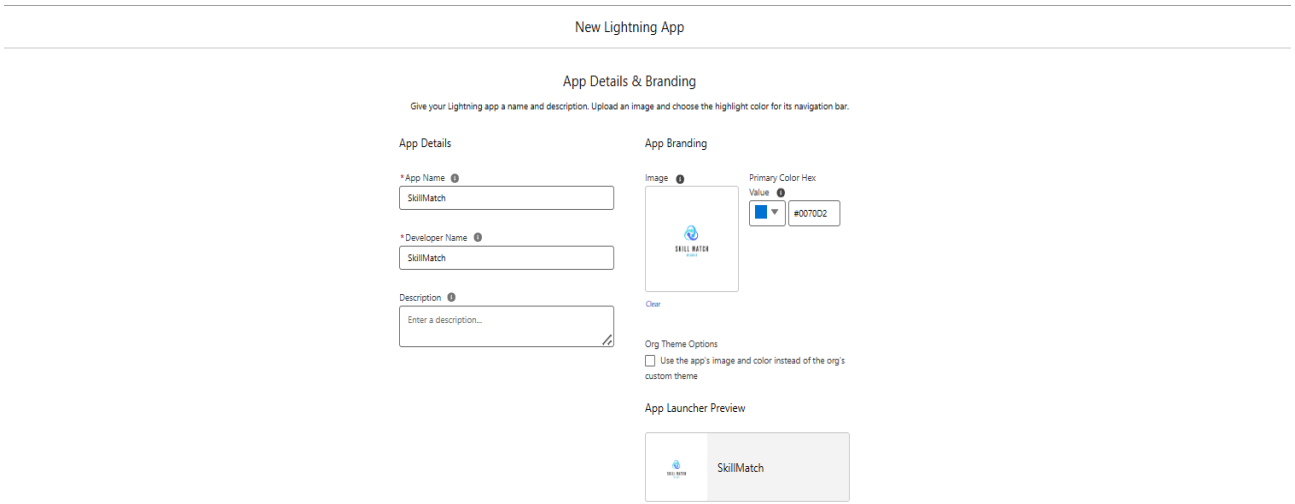
1. Click **Setup → Apex Classes → New**.
2. Copy and paste the **JobMatchingHelper** Code below.
3. Click **Save**.

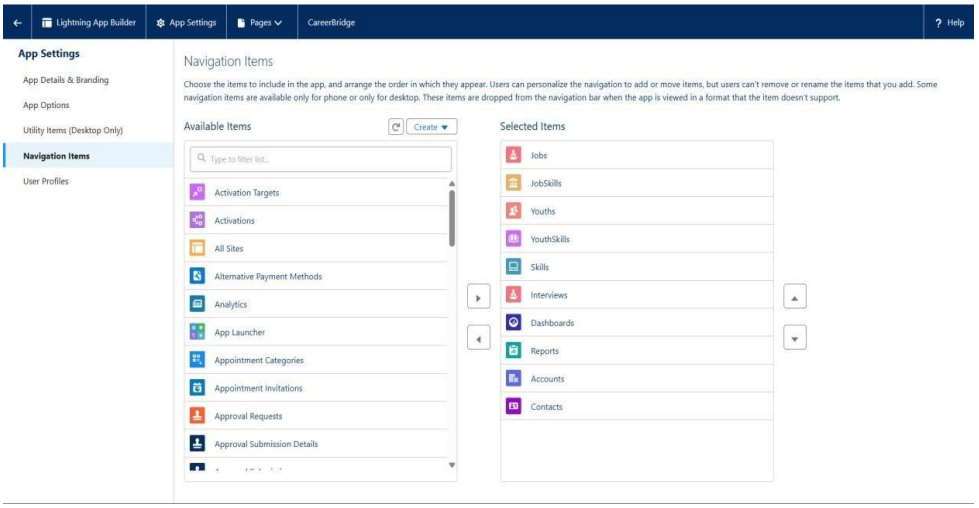


Phase 6: User Interface Development

Step1:Lightning App Builder

- Go to setup -> App Manager -> New Lighting App.
- Name the app CareerBridge.
- Add navigation items: Home, Youth, Job, Interviews, Reports, Dashboards
- Assign to profiles and Finish.

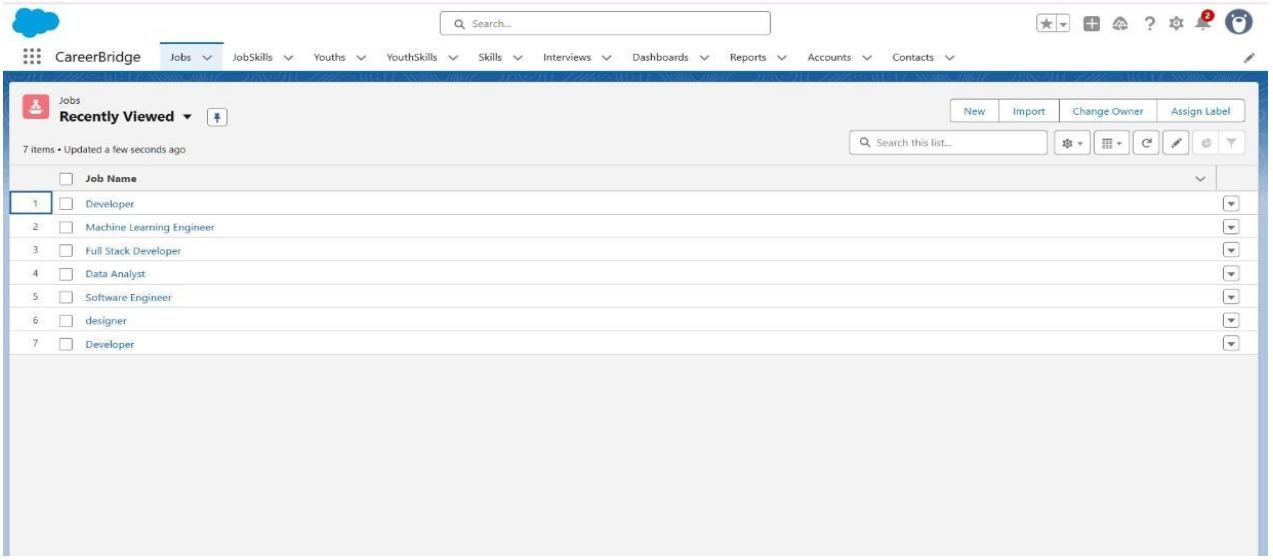




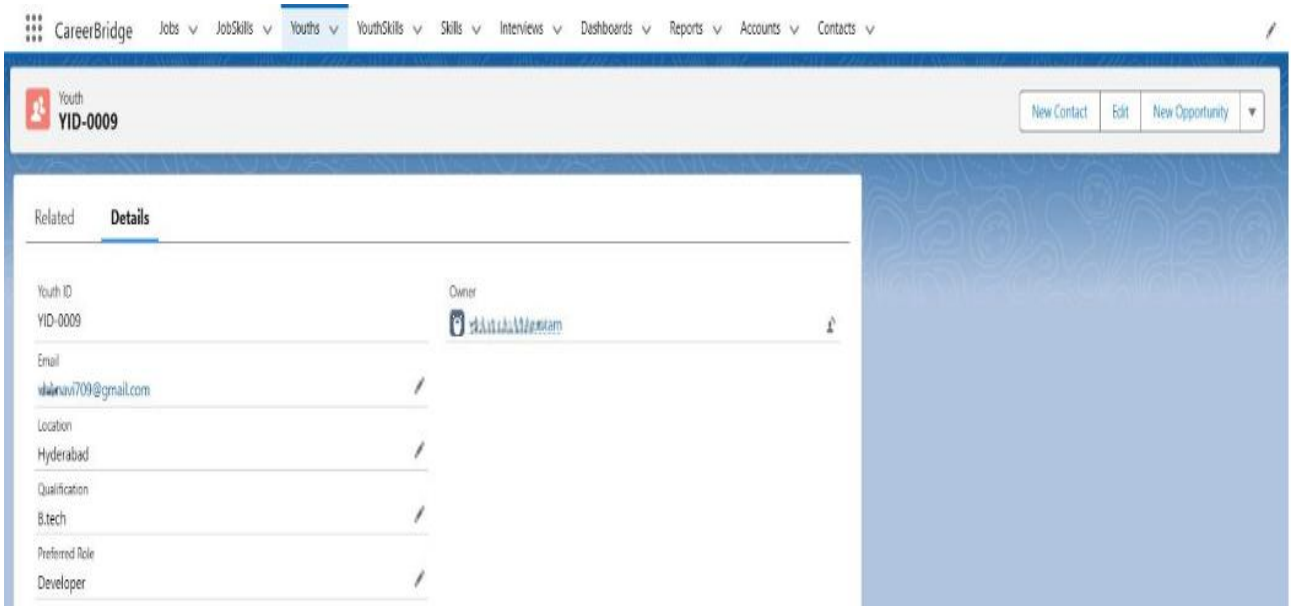
Step 2:Record Pages

- Custom layouts for specific object records **(Youth, Job, Interview, Skill).**
- You can decide what **fields, related lists, and components appear.**

Youth Record Page: Show youth details, skills, and recommended jobs.



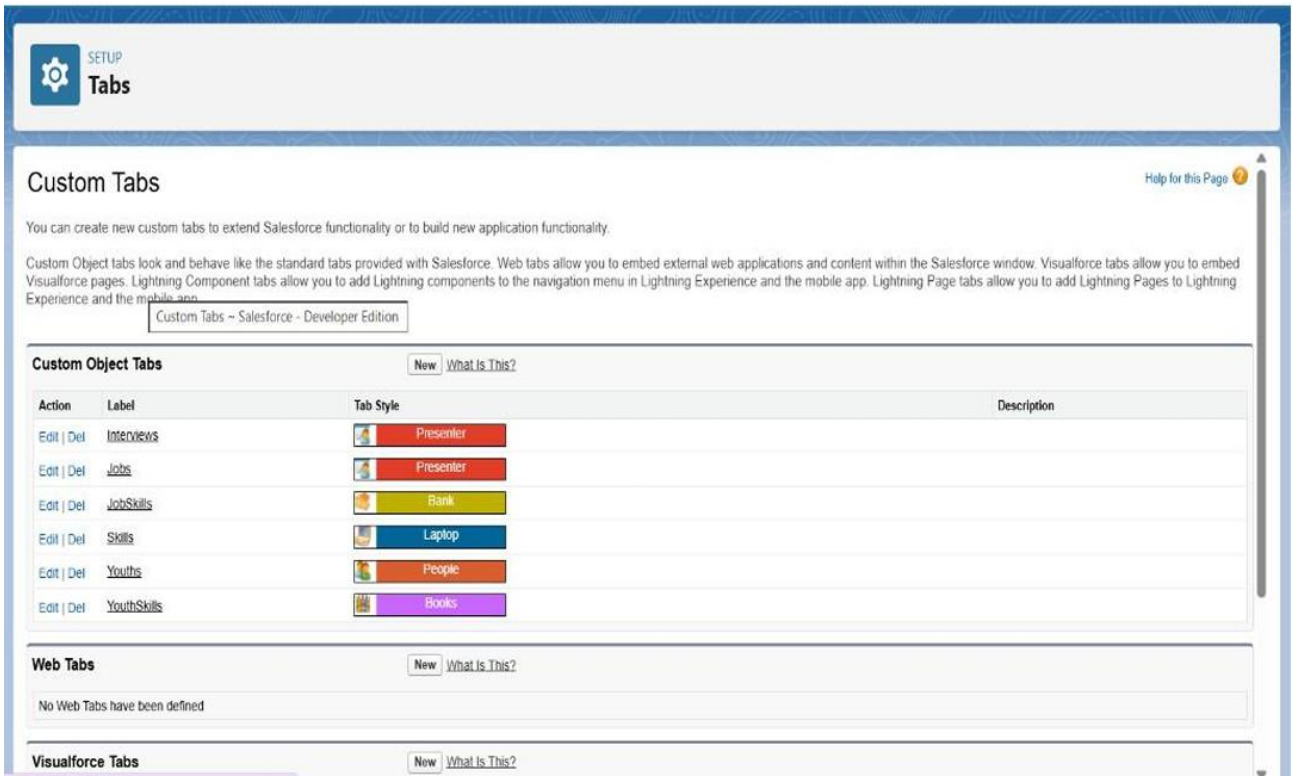
Job Record Page: Show job details, required skills, and linked inter views



- **Interview Record Page:** Show candidate, job, date, and status.

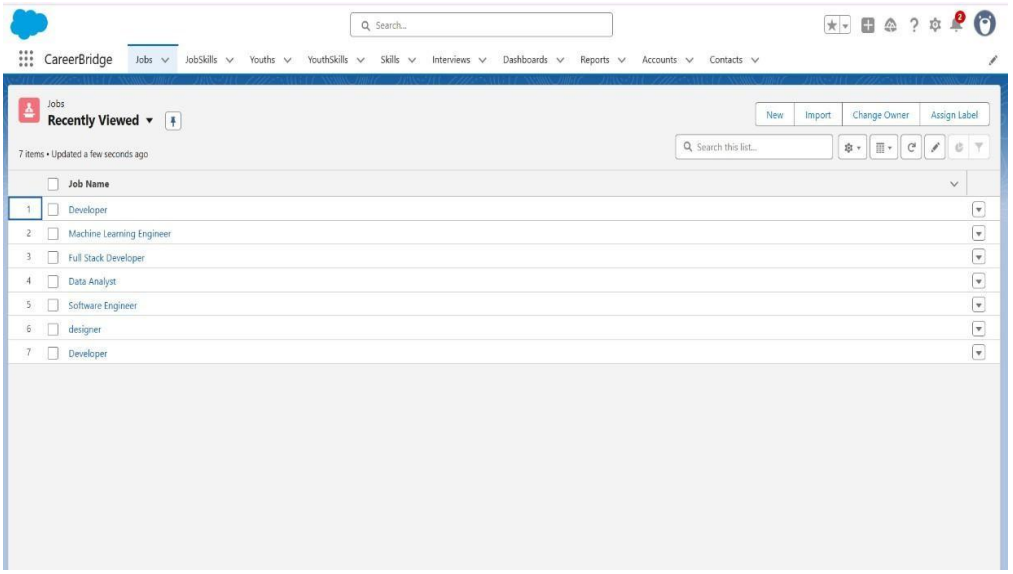
Step 3:Tabs

- Setup → **App Manager → Edit CareerBridge → Navigation Items**.
- Add or reorder tabs (**YouthSkill, Youth, Jobs, Interviews, JobsSkill, Skill**).
- **Save.**



Step 4:Home Page Layouts

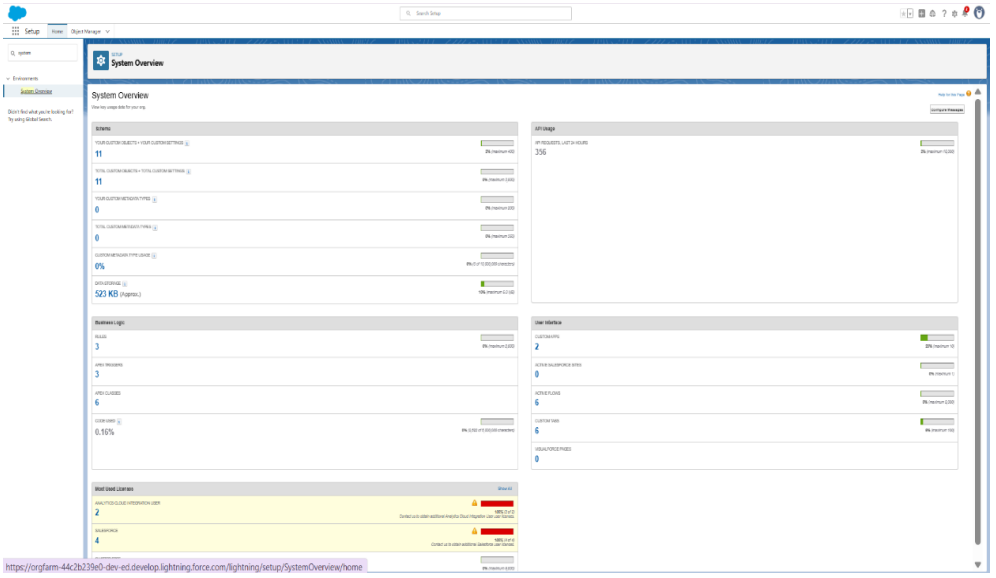
- The home page can be customized for different profiles (e.g., Youth, Admin).
- Provides dashboards, tasks, and quick actions.
    - **CareerBridge Example:**
        - For Youth: Show registered jobs, upcoming interviews, and assigned skills.
        - For Admin: Display KPIs like total youth registered, jobs posted, and interviews scheduled.



Phase 7: Integration & External Access

1. API Limits

- Salesforce tracks API calls made by external apps or integrations.
- In CareerBridge, all actions (Youth registration, Job posting, Flows, Emails) happen **inside Salesforce.**
- **No external API calls are used**, so API Limits **do not affect your project**.
- You can check usage in **Setup → System Overview**, but nothing needs to be configured.



Phase 8: Data Management & Deployment

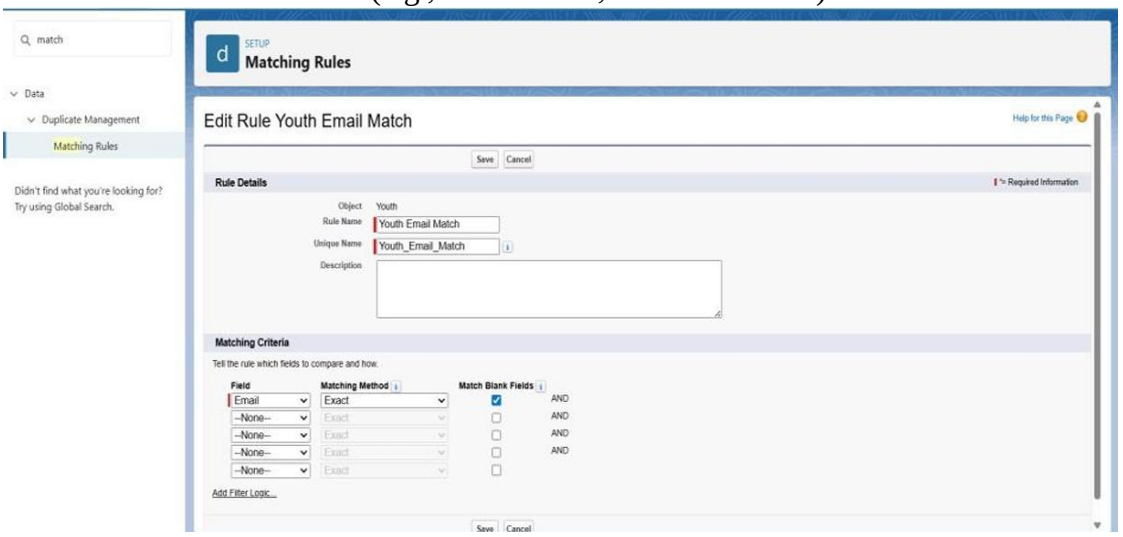
Step 1: Duplicate Management

Preventing duplicate records is critical, especially for **Youth** and **Job** objects.

A. Matching Rules

- Define the criteria to identify duplicates.
- **Steps:**

1. Go to **Setup → Matching Rules → New**.
2. Example: Match Youth_c.Email_c.
3. Define criteria (e.g., exact match, case-insensitive) → Save → Activate.



B. Duplicate Rules

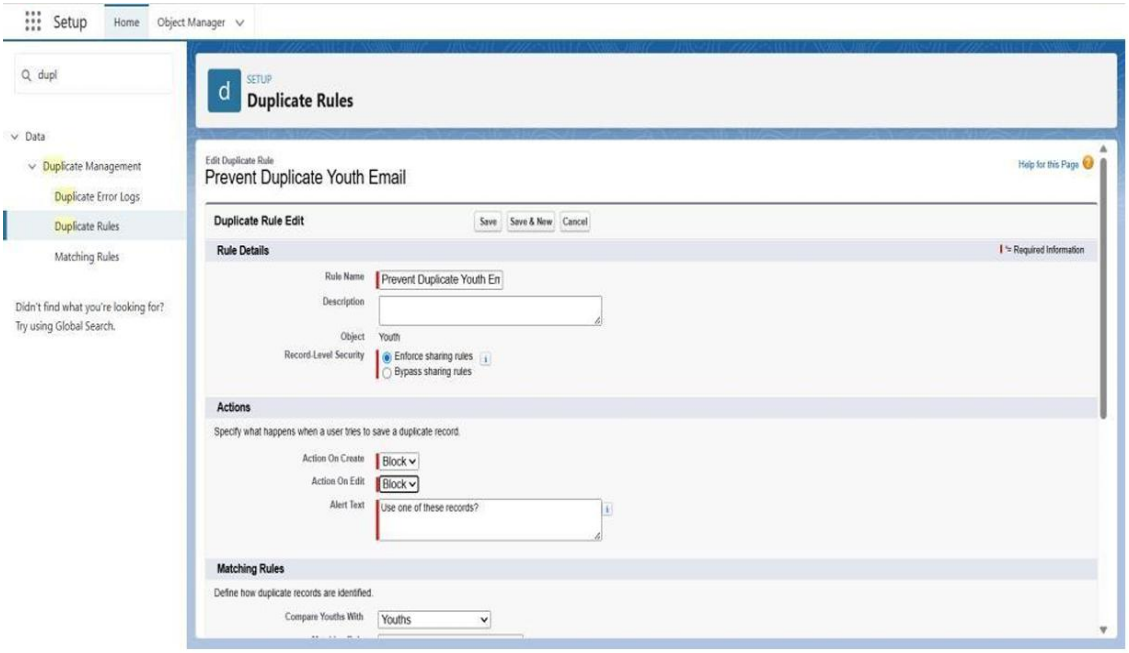
- Decide what happens when duplicates are detected.

- **Steps:**

1. Go to **Setup → Duplicate Rules → New**.
2. Apply your **Matching Rule** to the object (e.g., Youth__c).
3. Choose **Action**:
    - **Block** → Prevents user from creating duplicates.

- **Alert** → Warns the user, but allows creation.

4. Activate the rule.

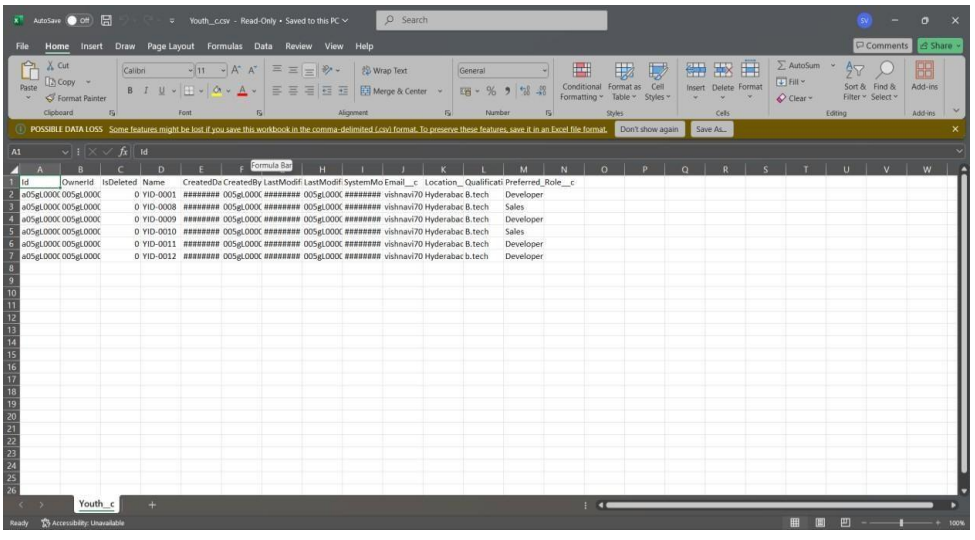


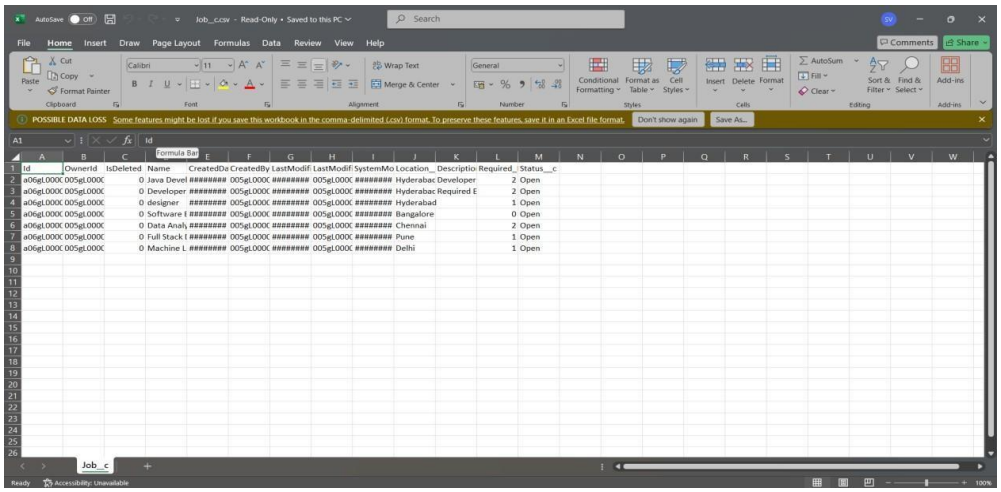
Step 2: Data Backup

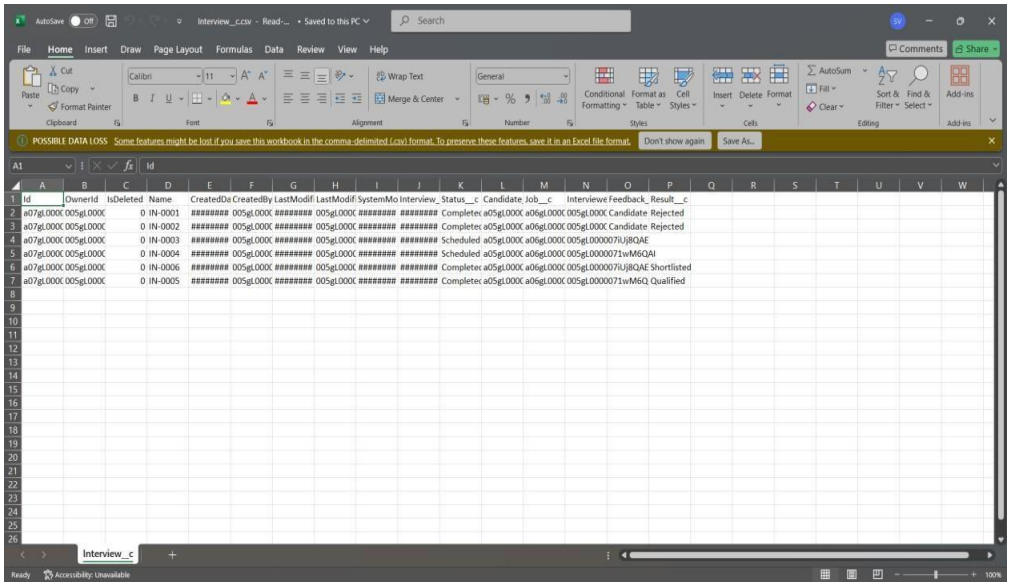
Always back up before importing or deploying, in case something goes wrong.

- **Steps:**

1. Go to **Setup → Data Export → Export Now**.
2. Choose objects to backup (e.g., Youth__c, Job__c, Interview__c).
3. Download the zip files.







Phase 9: Reporting, Dashboards & Security Review

Goal:

- Track progress through reports and dashboards
- Protect sensitive data from unauthorized access
- Ensure stakeholders see the right information in real time

Step 1: Reporting

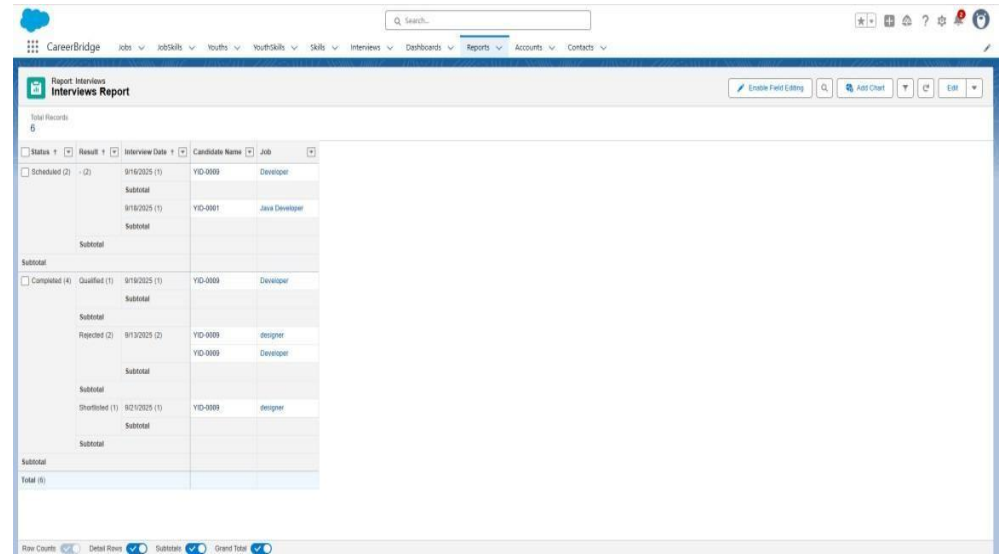
Reports help track Youth, Jobs, and Interviews effectively.

A. Create Reports

1. Go to **App Launcher → Reports → New Report**
2. Select the report type: e.g., **Interview** or **Custom Report**
3. Add fields:
    - Candidate Name (Youth__c Name)
    - Job Title (Job__c Name)
    - Interview Date (Interview_Date__c)
    - Status (Status__c)
4. Save & Run the report

B. Useful Reports

- **Interviews by Status** → Group by Status_c (Shows Scheduled vs Completed interviews)
- **Interviews by Result** → Group by Result_c (Shows Qualified, Shortlisted, Rejected outcomes)
- **Interviews Over Time** → Group by Last Interview Date (Shows trends or frequency of interviews)

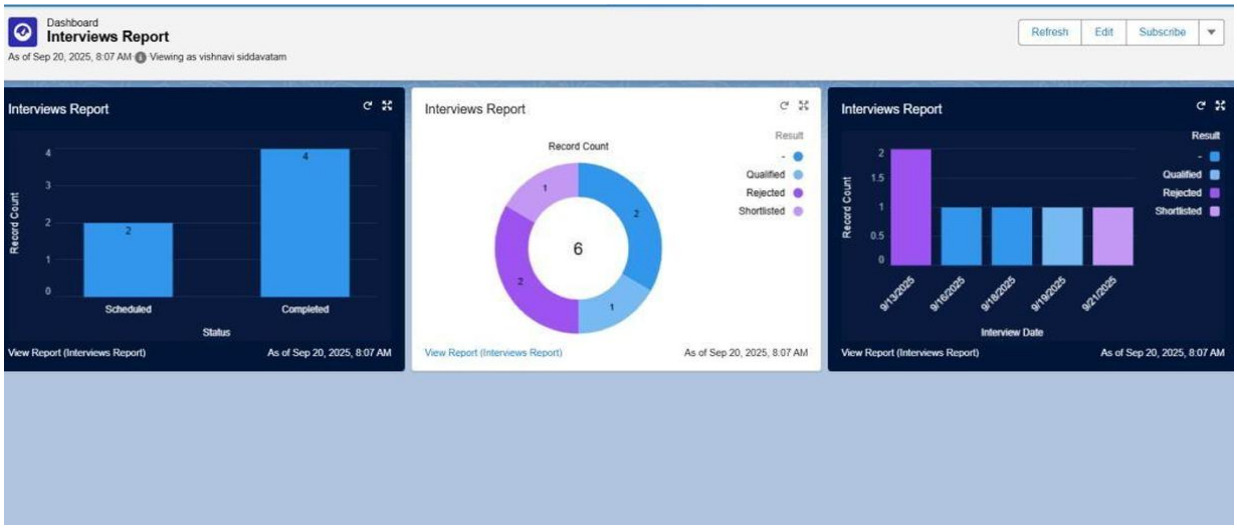


Step 2: Dashboards

Dashboards visually display metrics from reports.

A. Create Dashboard

1. Go to **App Launcher → Dashboards → New**
2. Add components:
    - **Metric:** Count of Completed Interviews
    - **Bar Chart:** Interviews by Status
    - **Line Chart:** Interviews per Week
    - **Table:** Recent Interviews
3. Enable Dynamic Dashboards → Set "View Dashboard As = Dynamic"

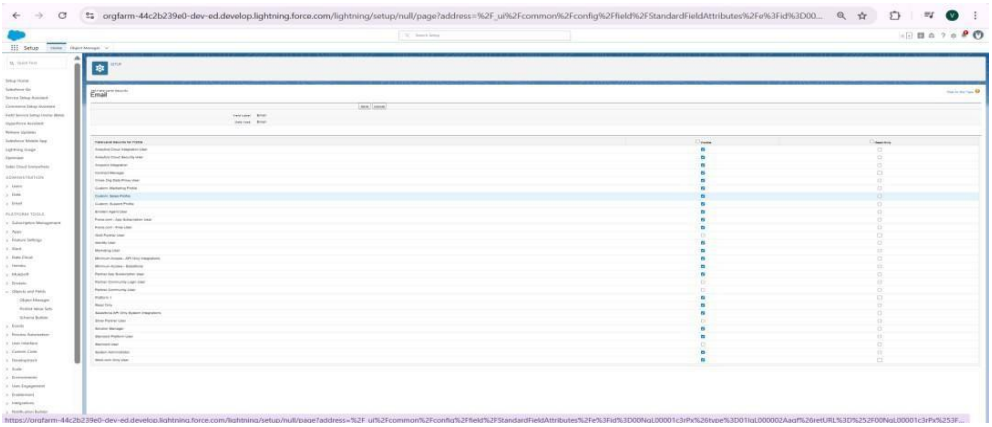


Step 3: Security

Protect sensitive data (like Youth emails, interview feedback) and control who can see what.

A. Field-Level Security (FLS)

1. Setup → Object Manager → Youth_c → Fields → Email_c → Set Field-Level Security



2. Hide fields from standard users or users without permission

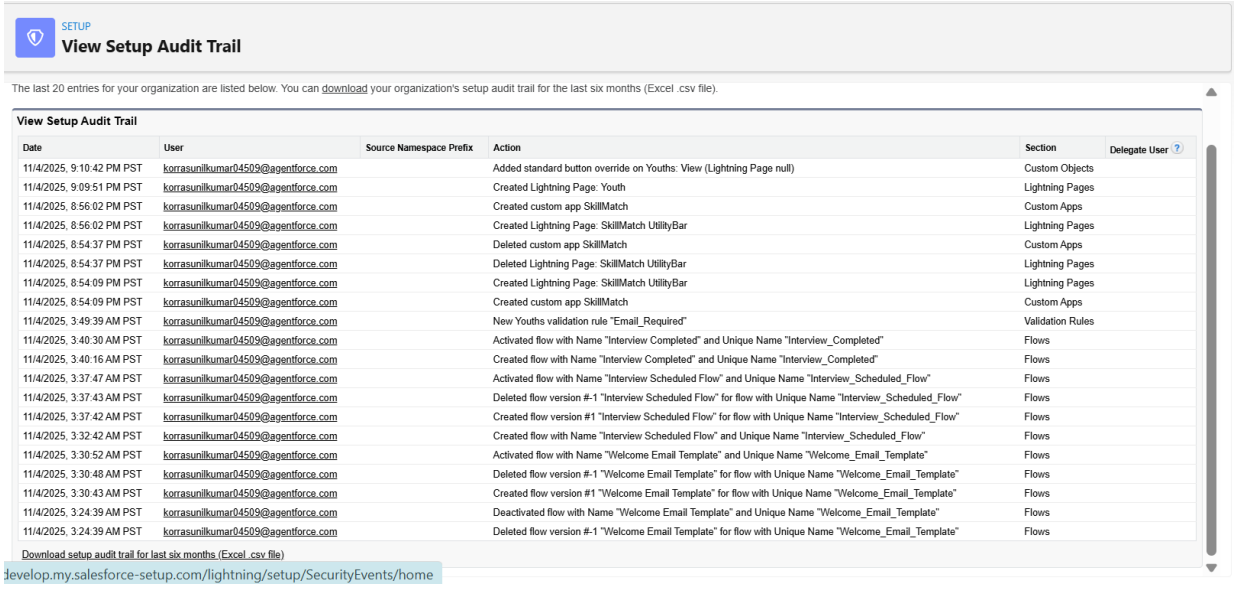
B. Sharing & Org-Wide Defaults (OWD)

1. Setup → Sharing Settings

2. Ensure Youth__c = Private

C. Audit Trail

1. Setup → View Setup Audit Trail
2. Track unauthorized changes or admin activity



SETUP
View Setup Audit Trail

The last 20 entries for your organization are listed below. You can download your organization's setup audit trail for the last six months (Excel .csv file).

View Setup Audit Trail

Date	User	Source Namespace Prefix	Action	Section	Delegate User
11/4/2025, 9:10:42 PM PST	korrasunilkumar04509@agentforce.com		Added standard button override on Youths: View (Lightning Page null)	Custom Objects	
11/4/2025, 9:09:51 PM PST	korrasunilkumar04509@agentforce.com		Created Lightning Page: Youth	Lightning Pages	
11/4/2025, 8:56:02 PM PST	korrasunilkumar04509@agentforce.com		Created custom app SkillMatch	Custom Apps	
11/4/2025, 8:56:02 PM PST	korrasunilkumar04509@agentforce.com		Created Lightning Page: SkillMatch UtilityBar	Lightning Pages	
11/4/2025, 8:54:37 PM PST	korrasunilkumar04509@agentforce.com		Deleted custom app SkillMatch	Custom Apps	
11/4/2025, 8:54:37 PM PST	korrasunilkumar04509@agentforce.com		Deleted Lightning Page: SkillMatch UtilityBar	Lightning Pages	
11/4/2025, 8:54:09 PM PST	korrasunilkumar04509@agentforce.com		Created Lightning Page: SkillMatch UtilityBar	Lightning Pages	
11/4/2025, 8:54:09 PM PST	korrasunilkumar04509@agentforce.com		Created custom app SkillMatch	Custom Apps	
11/4/2025, 3:49:39 AM PST	korrasunilkumar04509@agentforce.com		New Youths validation rule "Email_Required"	Validation Rules	
11/4/2025, 3:40:30 AM PST	korrasunilkumar04509@agentforce.com		Activated flow with Name "Interview Completed" and Unique Name "Interview_Completed"	Flows	
11/4/2025, 3:40:16 AM PST	korrasunilkumar04509@agentforce.com		Created flow with Name "Interview Completed" and Unique Name "Interview_Completed"	Flows	
11/4/2025, 3:37:47 AM PST	korrasunilkumar04509@agentforce.com		Activated flow with Name "Interview Scheduled Flow" and Unique Name "Interview_Scheduled_Flow"	Flows	
11/4/2025, 3:37:43 AM PST	korrasunilkumar04509@agentforce.com		Deleted flow version #-1 "Interview Scheduled Flow" for flow with Unique Name "Interview_Scheduled_Flow"	Flows	
11/4/2025, 3:37:42 AM PST	korrasunilkumar04509@agentforce.com		Created flow version #1 "Interview Scheduled Flow" for flow with Unique Name "Interview_Scheduled_Flow"	Flows	
11/4/2025, 3:32:42 AM PST	korrasunilkumar04509@agentforce.com		Created flow with Name "Interview Scheduled Flow" and Unique Name "Interview_Scheduled_Flow"	Flows	
11/4/2025, 3:30:52 AM PST	korrasunilkumar04509@agentforce.com		Activated flow with Name "Welcome Email Template" and Unique Name "Welcome_Email_Template"	Flows	
11/4/2025, 3:30:48 AM PST	korrasunilkumar04509@agentforce.com		Deleted flow version #-1 "Welcome Email Template" for flow with Unique Name "Welcome_Email_Template"	Flows	
11/4/2025, 3:30:43 AM PST	korrasunilkumar04509@agentforce.com		Created flow version #1 "Welcome Email Template" for flow with Unique Name "Welcome_Email_Template"	Flows	
11/4/2025, 3:24:39 AM PST	korrasunilkumar04509@agentforce.com		Deactivated flow with Name "Welcome Email Template" and Unique Name "Welcome_Email_Template"	Flows	
11/4/2025, 3:24:39 AM PST	korrasunilkumar04509@agentforce.com		Deleted flow version #-1 "Welcome Email Template" for flow with Unique Name "Welcome_Email_Template"	Flows	

Download setup audit trail for last six months (Excel .csv file)

develop.my.salesforce-setup.com/lightning/setup/SecurityEvents/home

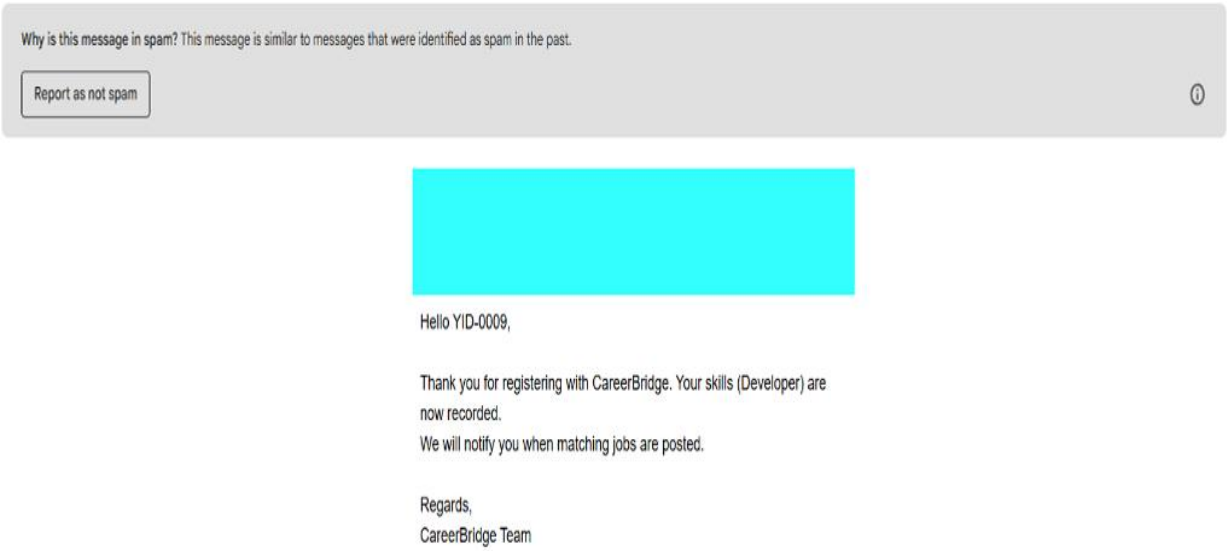
Phase 10: Quality Assurance Testing

Test Case 1 – Youth Registration → Welcome Email

- **Use Case / Scenario:** Youth registers on the platform.
- **Test Steps (Input):**

- Fill Youth form → Name = Vishnavi, Email = vishnavi@gmail.com, Skills = Java, Python.

- **Expected Result:**
  - Youth record saved in **Youth__c** object.
  - **Welcome Email** triggered:
    "Hello , thank you for registering. Your skills (Python) are now recorded."

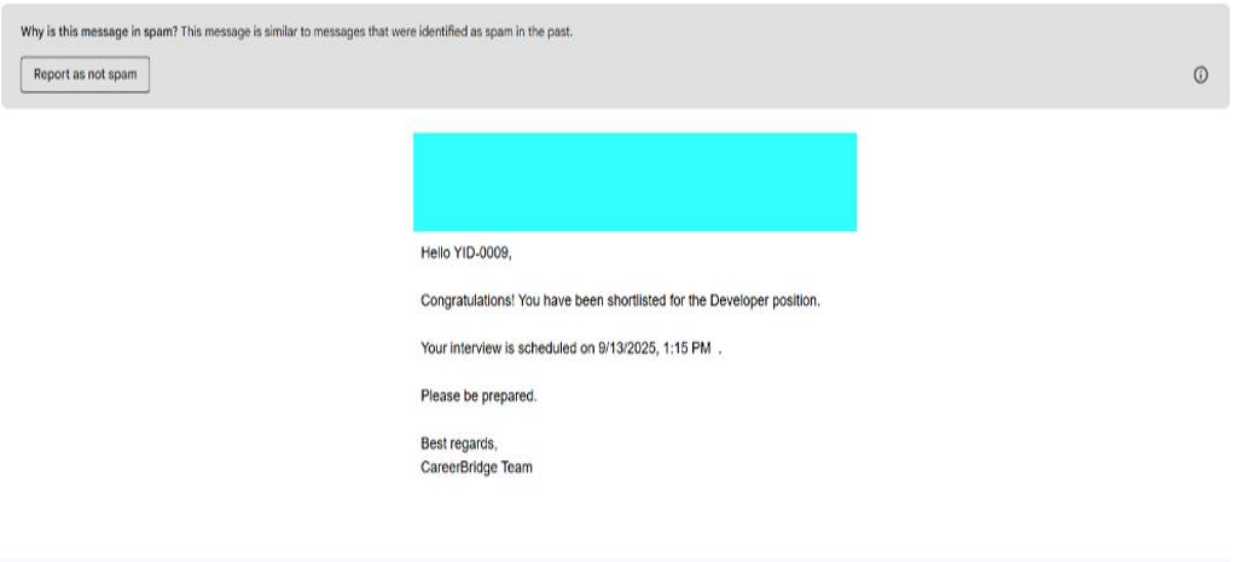


Test Case 2 – Employer Posts Job → Skills Match → Interview Scheduled Email

- **Use Case / Scenario:** Employer posts job requiring skills.
- **Test Steps (Input):**

- Job Title = Java Developer, Location = Bangalore, Required Skill = Java.

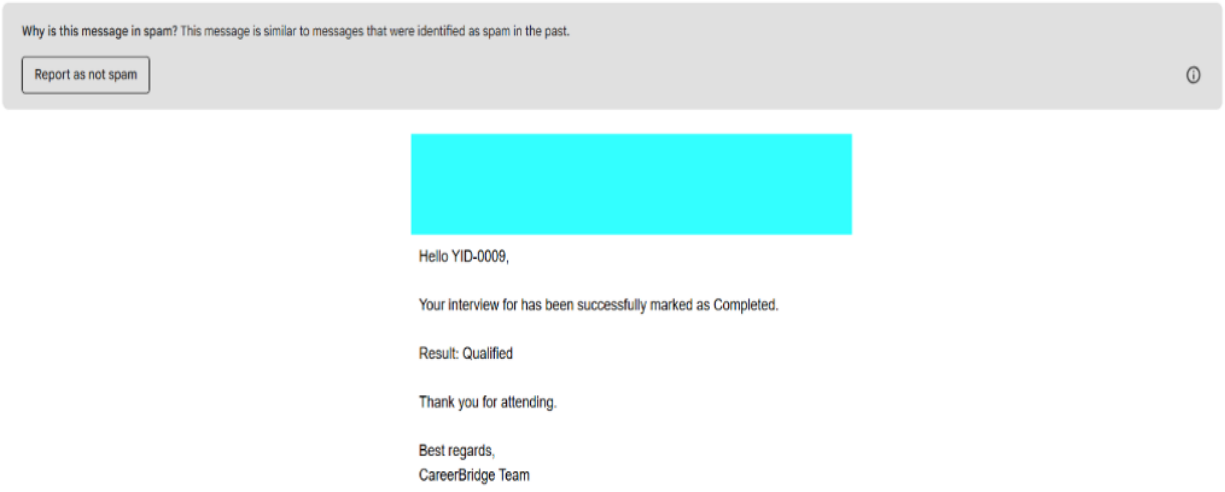
- **Expected Result:**
  - Job record created in **Job__c** object.
  - System checks youth skills → Match found with Vishnavi.
  - **Interview__c record** auto-created with Status = Scheduled.
  - **Interview Scheduled Email** triggered:
    "Hello Vishnavi, congratulations! You are shortlisted for Java Developer. Interview is scheduled on [Date & Time]."



Test Case 3 – Youth Attends Interview → Interview Completed Email

- **Use Case / Scenario:** Interview is completed.

- **Test Steps (Input):**
    - Coordinator updates Interview record → Change Status__c = Completed.
- **Expected Result:**
    - Status changes to Completed.
    - **Interview Completed Email** triggered:
      "Hello Vishnavi, your interview for Java Developer has been marked as Completed. Thank you for attending."

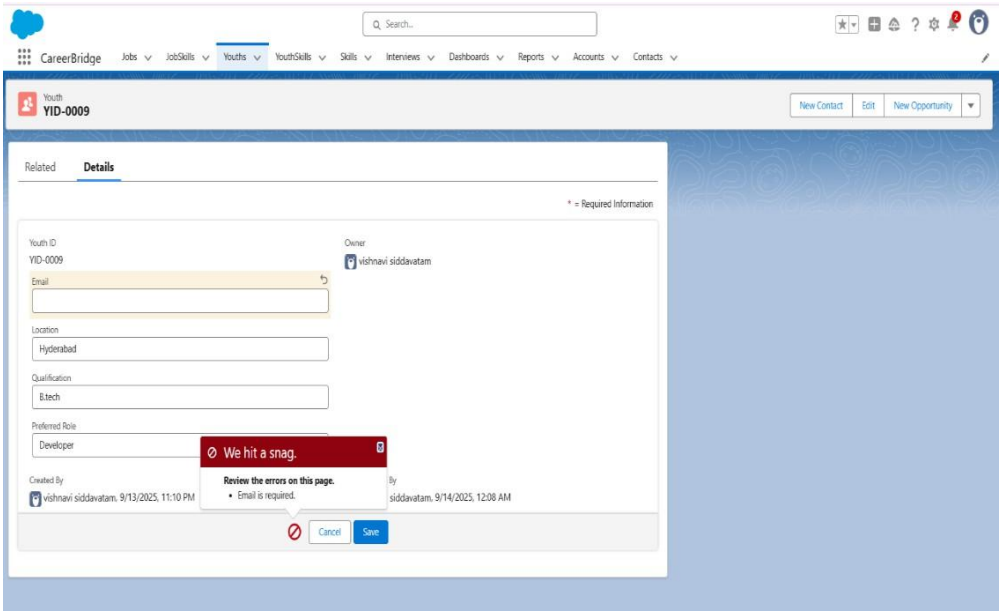


Test Case 4 – Validation Rules (Error Handling)

- **Use Case / Scenario:** Youth tries to register without Email or Skills.
- **Test Steps (Input):**

- Leave Email blank → Save.
- Leave Location blank → Save.

- **Expected Result:**
  - System displays validation error messages:
    - ISBLANK(Email__c) → “Email is required.”
    - ISBLANK(Location__c) → “Location must be entered.”



Conclusion

The CareerBridge CRM project was successfully implemented and tested across all phases. In Phase 10, Quality Assurance Testing validated that every Salesforce feature—record creation, flows, triggers, automatic emails, and validation rules—worked as expected.

The end-to-end flow ensures:

- Youth registration captures complete details and sends a **Welcome Email**.
- Employers post jobs, and the system automatically matches required skills with youth profiles.
- Matching results trigger interview scheduling along with an **Interview Scheduled Email**.
- Once interviews are completed, coordinators update the status, and the system sends an **Interview Completed Email**.
- Validation rules prevent saving incomplete or incorrect data, ensuring accuracy.

This testing confirms that the platform is:

- **Reliable** – All workflows run without errors.
- **Automated** – Reduces manual work for NGOs and coordinators.
- **User-friendly** – Provides transparency to youth, NGOs, and employers.

Final Outcome: CareerBridge CRM successfully connects youth skills to jobs, supports NGOs in managing opportunities, and helps employers find the right candidates efficiently