



Project Title: A CRM Application to Manage the Services offered by an Institution

College Name: MAILAM ENGINEERING COLLEGE

Code: 4216

Department of Computer Science and Business Systems

Team Leader:

Name: SANTHOSH KUMAR

Reg No: 421622244039

Team Members:

Name: KARTHEESHWARAN S S Reg No: 421622244019

Name: AJMAL HUSSAIN I Reg No: 421622244001

Name: SANJAI KUMAR N Reg No: 421622244037

Registered Email Id: rsksanthos08@gmail.com
TrailheadUrl: <https://www.salesforce.com/trailblazer/r/7tsvx8d5kptxa3929>



Project Abstract

The project involved developing a Salesforce CRM application for EduConsultPro Institute, aimed at improving the management of student admissions, consulting requests, and immigration cases. The system enabled prospective students to submit admission applications online, with data automatically captured in Salesforce and automated email notifications sent to applicants. The project also included managing consulting services, where students could request consultations, and consultants were notified to schedule and manage appointments within the CRM. Additionally, an immigration case management system was implemented to efficiently log, process, and track cases, with automated notifications for agents and robust document management features. This solution significantly streamlined operations and enhanced the overall experience for both students and staff at EduConsultPro.

Table of Contents

| | |
|---|----|
| Project Abstract | 2 |
| Table of Contents..... | 2 |
| Introduction | 6 |
| <i>TASK 1: Creating Developer Account.....</i> | 7 |
| <i>TASK 2: Account Activation</i> | 8 |
| <i>TASK 3: Create Objects From Spreadsheet.....</i> | 9 |
| Subtask 1: Create Course Object | 9 |
| Subtask 2: Create Relationships Among the Objects..... | 12 |
| Subtask 3: Configure the Case Object..... | 14 |
| Subtask 4: Create a Lightning App..... | 16 |
| <i>TASK 4: Create A ScreenFlow For Student Admission Application Process</i> | 19 |
| Subtask 1: Add Screen Element..... | 19 |
| Subtask 2: Create Student Record Using Create Element..... | 21 |
| Subtask 3: Add Course Selection Screen | 22 |
| Subtask 4: Add Decision Element | 23 |
| Subtask 5: Add GET Record Element | 24 |
| Subtask 6: Create Registration Record Using Create Records Element..... | 25 |



| | |
|--|-----------|
| Subtask 7: Create Email Text Template Variables | 26 |
| Subtask 8: Add Action Element | 28 |
| Subtask 9: Add Success Screen | 29 |
| Task 5: Create Users | 30 |
| Subtask 1: Create a User | 30 |
| Subtask 2: Configure the User Settings | 31 |
| Task 6: Create an Approval Process for the Property Object..... | 32 |
| Subtask 1: Create Email Templates | 32 |
| Subtask 2: Create An Approval Process | 35 |
| Task 7: Create A Record-Triggered Flow | 39 |
| Subtask 1: Configure The Start Element..... | 39 |
| Subtask 2: Add An Action Element..... | 40 |
| Task 8: Create A ScreenFlow For Existing Students To Book An Appointment 42 | |
| Subtask 1: Add Screen Element..... | 42 |
| Subtask 2: Get Record..... | 43 |
| Subtask 3: Add a Decision Element..... | 44 |
| Subtask 4: Add Screen Element..... | 45 |
| Subtask 5: Add GET Record Element | 46 |
| Subtask 6: Create Appointment Record Using Create Records Element..... | 47 |



| | |
|---|-----------|
| Subtask 7: Add Screen Element..... | 48 |
| Subtask 8: Add a Subflow Element | 49 |
| Task 9: Create A ScreenFlow To Combine All The Flows At One Place..... | 50 |
| Subtask 1: Add Welcome Screen Element..... | 50 |
| Subtask 2: Add Existing or New Student Confirmation Screen | 51 |
| Subtask 3: Add Decision Element | 52 |
| Subtask 4: Add Subflow for Existing Students | 54 |
| Subtask 5: Add Subflow for New Students..... | 54 |
| Task 10: Create A Lightning App Page..... | 56 |
| Subtask 1: Create A Lightning App Page..... | 56 |



INTRODUCTION

EduConsultPro Institute, a prominent educational institution, offers a diverse array of courses and programs to students from various backgrounds. With the institute's increasing popularity, managing the growing number of student admissions, consulting requests, and immigration cases became a significant challenge. To overcome these operational inefficiencies, EduConsultPro embarked on a project to implement a robust Customer Relationship Management (CRM) system using Salesforce.

The project's primary objective was to streamline and automate the institute's core administrative processes. For admission application management, the goal was to enable prospective students to submit their applications through the institute's online portal. This information would be captured directly into the Salesforce CRM, ensuring efficient data handling. Additionally, automated email notifications were set up to acknowledge successful submissions, and tools were provided to admissions staff for generating insightful reports and dashboards, allowing for a deeper analysis of application metrics, acceptance rates, and enrollment trends.

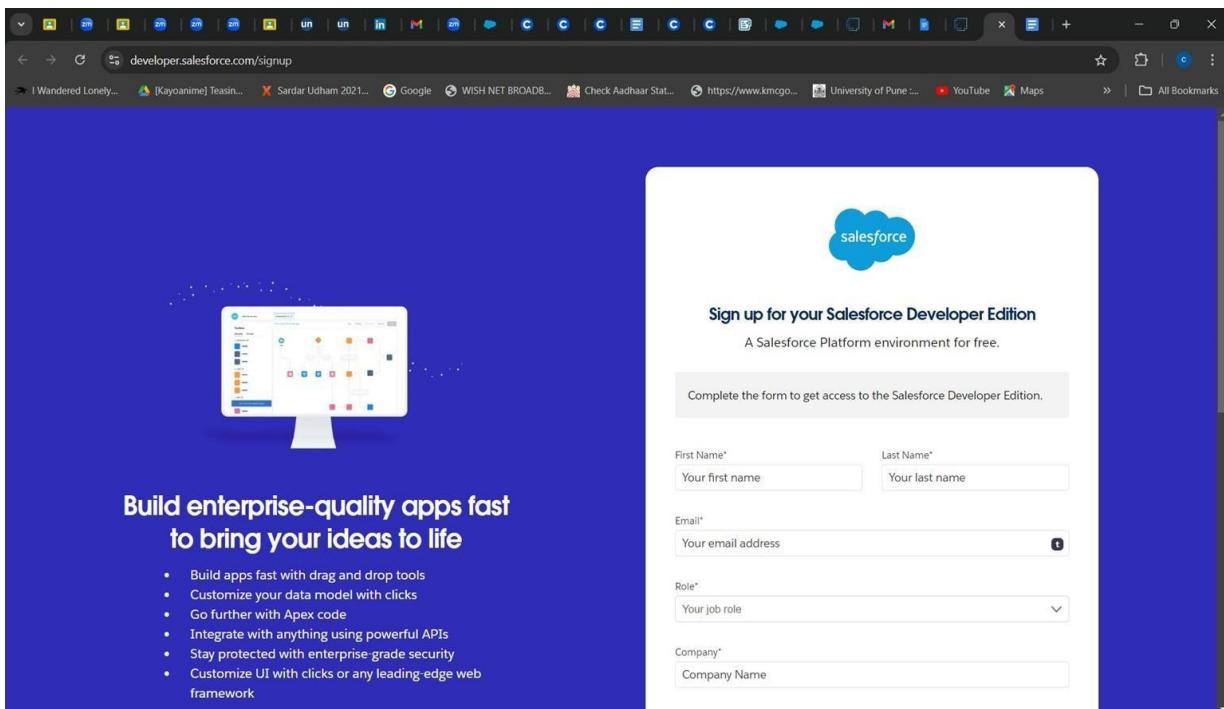
In terms of consulting services management, the project focused on creating a seamless process where students could request consulting services via the institute's website. These requests would be recorded in Salesforce, triggering automated notifications to consultants who could then schedule and manage appointments within the CRM. This ensured that the scheduling process was organized, and the status of each appointment, whether scheduled, completed, or canceled, was tracked effectively.

Another critical component was the implementation of an immigration case management system. Students could initiate immigration cases through various channels such as phone, email, or web. These cases were then logged into Salesforce, with immigration agents receiving immediate notifications to take action. The system also included tools for tracking case statuses and managing related documents, ensuring that every case was processed efficiently and transparently.

This comprehensive Salesforce solution not only addressed the immediate operational needs of EduConsultPro but also set the foundation for future scalability, enabling the institution to provide an enhanced and streamlined experience for both students and staff.

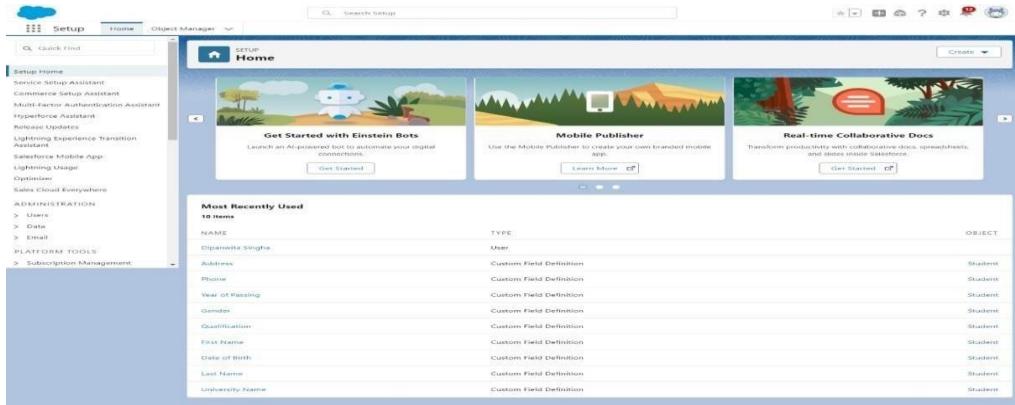
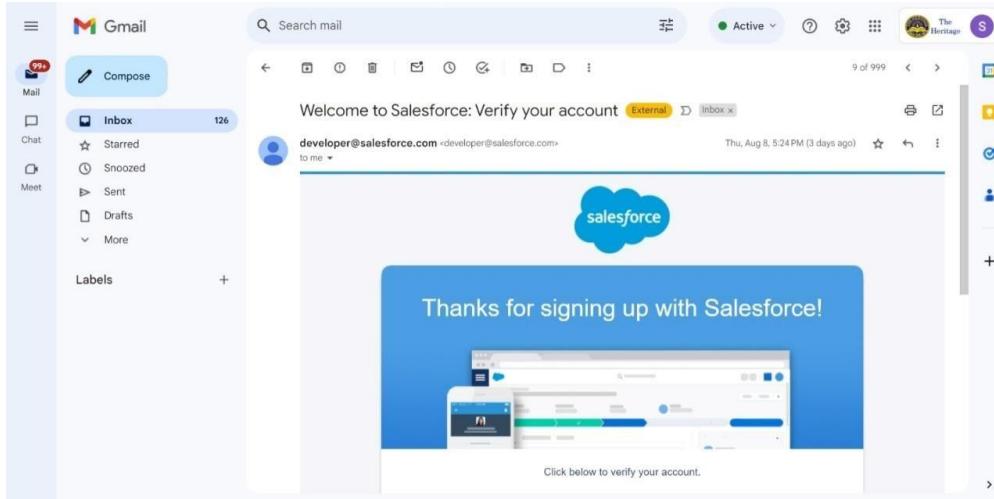
TASK 1: Creating Developer Account

1. Go to <https://developer.salesforce.com/signup>
2. Fill up form with personal information



TASK 2: Account Activation

1. Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account.
2. Give a password and answer a security question and click on change password.
3. Then you will redirect to your salesforce setup page.



A screenshot of the Salesforce Setup Home page. The left sidebar includes links for 'Setup Home', 'Service Setup Assistant', 'Commerce Setup Assistant', 'Multi-Factor Authentication Assistant', 'Hyperedge Assistant', 'Release Updates', 'Lightning Experience Transition Assistant', 'Salesforce Mobile App', 'Lightning Usage Optimizer', 'Sales Cloud Everywhere', 'Data', 'Email', 'Platform Tools', and 'Subscription Management'. The main content area features three cards: 'Get Started with Einstein Bots', 'Mobile Publisher', and 'Real-time Collaborative Docs'. Below these cards is a section titled 'Most Recently Used' with a table showing 10 items. The table columns are 'NAME', 'TYPE', and 'OBJECT'. The items listed are:

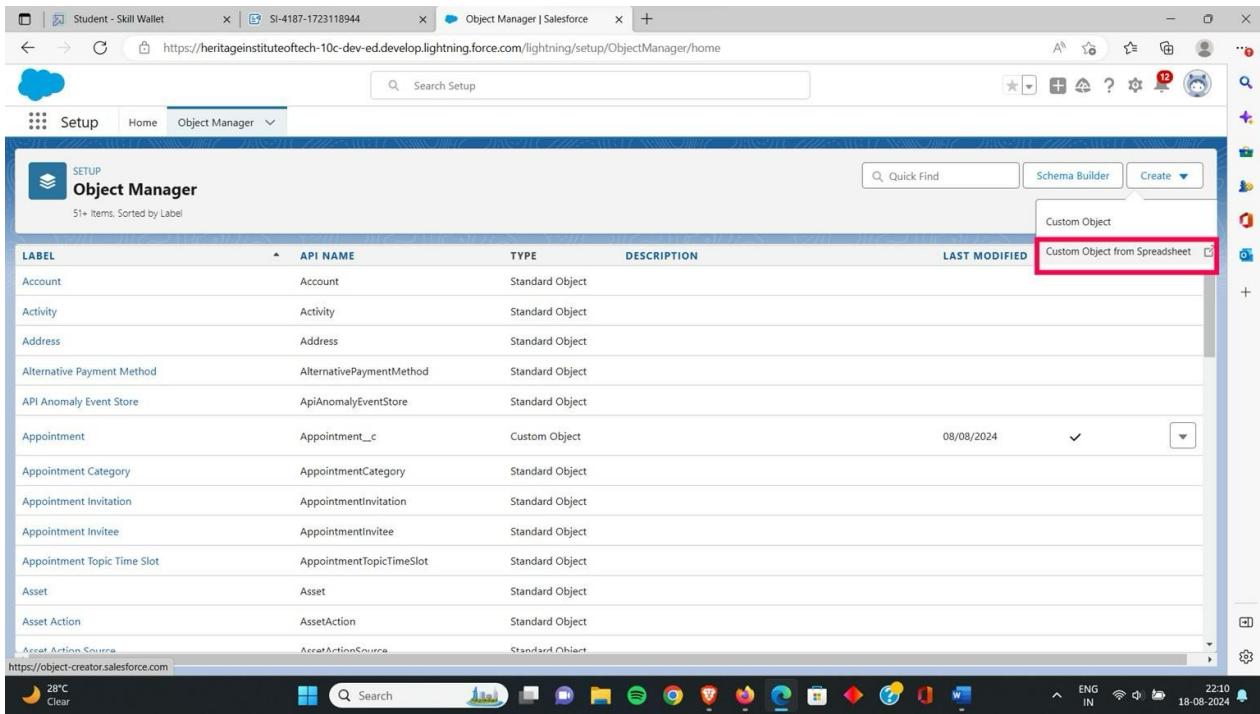
| NAME | TYPE | OBJECT |
|-----------------|-------------------------|---------|
| Dipenita Singha | User | Student |
| Address | Custom Field Definition | Student |
| Phone | Custom Field Definition | Student |
| Year of Passing | Custom Field Definition | Student |
| Gender | Custom Field Definition | Student |
| Qualification | Custom Field Definition | Student |
| First Name | Custom Field Definition | Student |
| Date of Birth | Custom Field Definition | Student |
| Last Name | Custom Field Definition | Student |
| University Name | Custom Field Definition | Student |

TASK 3: Create Objects from Spreadsheet

This task involves directly creating objects in Salesforce by importing data from spreadsheets and establishing relationships between them.

Subtask 1: Create Course Object

1. Navigate to **Object Manager**.
2. Click on **Create Object from Spreadsheet**.



The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes links for Setup, Home, and Object Manager. The main area is titled "Object Manager" and displays a list of objects with columns for Label, API Name, Type, Description, and Last Modified. A specific row for "Custom Object" is highlighted with a red box, and the "Last Modified" field shows "Custom Object from Spreadsheet". The bottom status bar shows the URL "https://object-creator.salesforce.com" and the date "18-08-2024".

3. Upload the spreadsheet to Salesforce.



4. Map the fields from the spreadsheet to the object fields.

Create a custom object from a spreadsheet

Define object and fields

Choose the data source, map fields and their types, and import field data.

Worksheet Details

Field Label Source * Field Labels Row
 Enter manually Detect from row 1

Import 2 rows of Data? No, skip import Yes, import data

Record Name Field Let Salesforce Create a Default

Fields 12 of 12 to import Hide mapped fields

| IMPORT FILE FIELD NAME | SALESFORCE FIELD NAME | SALESFORCE FIELD TYPE | ADD TO LAYOUTS | FIELD PREVIEW |
|------------------------|-----------------------|-----------------------|-------------------------------------|---------------|
| ✓ Address | Address | Text | <input checked="" type="checkbox"/> | Hyderabad |
| ✓ City | City | Text | <input checked="" type="checkbox"/> | Hyderabad |
| ✓ Phone | Phone | Phone | <input checked="" type="checkbox"/> | 1234567890 |
| ✓ Qualification | Qualification | Text | <input checked="" type="checkbox"/> | M.Sc |
| ✓ University Name | University Name | Text | <input type="checkbox"/> | BHIT |
| ✓ Year of Passing | Year of Passing | Date | <input type="checkbox"/> | 2020 |

Back Next

5. Define the Object properties

Create a custom object from a spreadsheet

Object properties

Almost finished! Time to define your object's attributes.

* Label

* Plural Label

* API Name

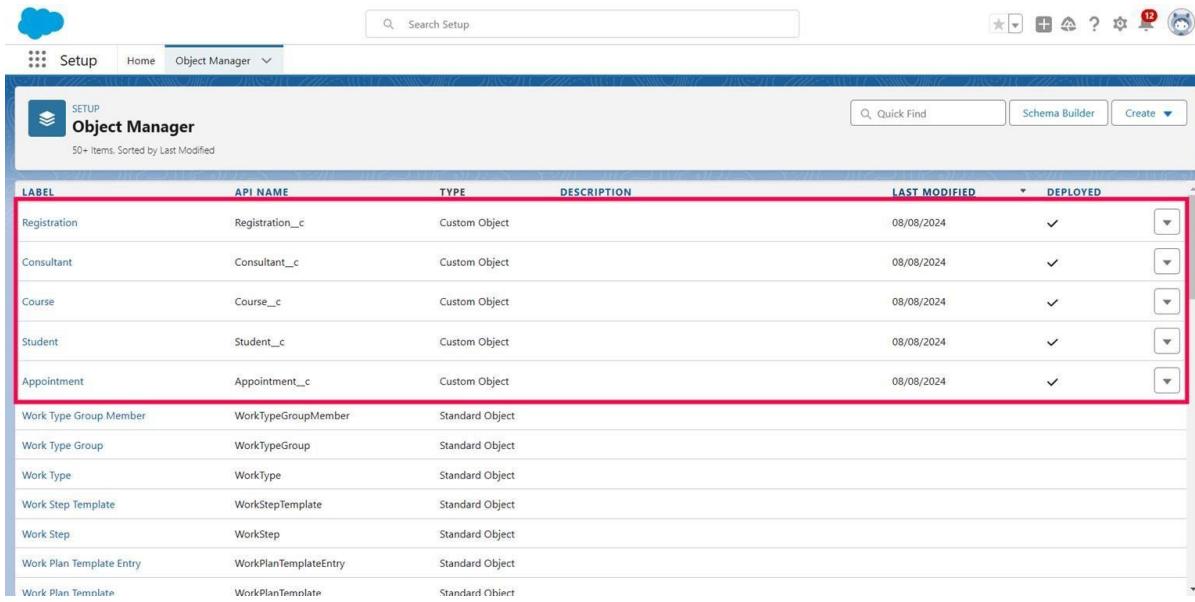
Object Description

> Advanced Settings

Back Finish

6. Following the same steps create the following objects:

- a. Consultant**
- b. Student**
- c. Appointment**
- d. Registration**



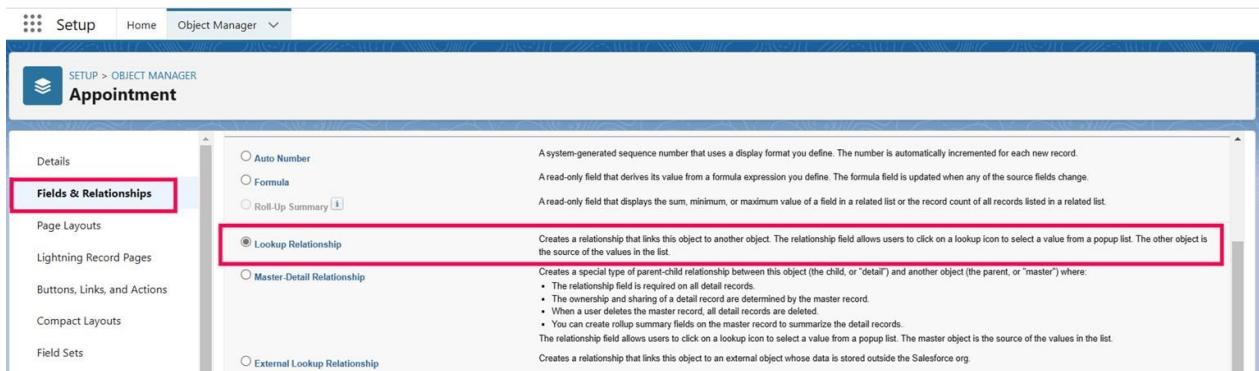
The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'SETUP', 'Home', and 'Object Manager'. The main area is titled 'Object Manager' with the sub-header '50+ items. Sorted by Last Modified'. A search bar at the top right says 'Search Setup'. Below is a table with columns: LABEL, API NAME, TYPE, DESCRIPTION, LAST MODIFIED, and DEPLOYED. The rows for 'Registration', 'Consultant', 'Course', 'Student', and 'Appointment' are highlighted with a red border.

| LABEL | API NAME | TYPE | DESCRIPTION | LAST MODIFIED | DEPLOYED |
|--------------------------|-----------------------|-----------------|-------------|---------------|----------|
| Registration | Registration__c | Custom Object | | 08/08/2024 | ✓ |
| Consultant | Consultant__c | Custom Object | | 08/08/2024 | ✓ |
| Course | Course__c | Custom Object | | 08/08/2024 | ✓ |
| Student | Student__c | Custom Object | | 08/08/2024 | ✓ |
| Appointment | Appointment__c | Custom Object | | 08/08/2024 | ✓ |
| Work Type Group Member | WorkTypeGroupMember | Standard Object | | | |
| Work Type Group | WorkTypeGroup | Standard Object | | | |
| Work Type | WorkType | Standard Object | | | |
| Work Step Template | WorkStepTemplate | Standard Object | | | |
| Work Step | WorkStep | Standard Object | | | |
| Work Plan Template Entry | WorkPlanTemplateEntry | Standard Object | | | |
| Work Plan Template | WorkPlanTemplate | Standard Object | | | |

Subtask 2: Create Relationships Among the Objects

1. Create a Lookup relationship between Appointment and Student Object

- a. Go to the Appointment object and create a new custom field from Fields & Relationships and choose the field type **Lookup relationship****



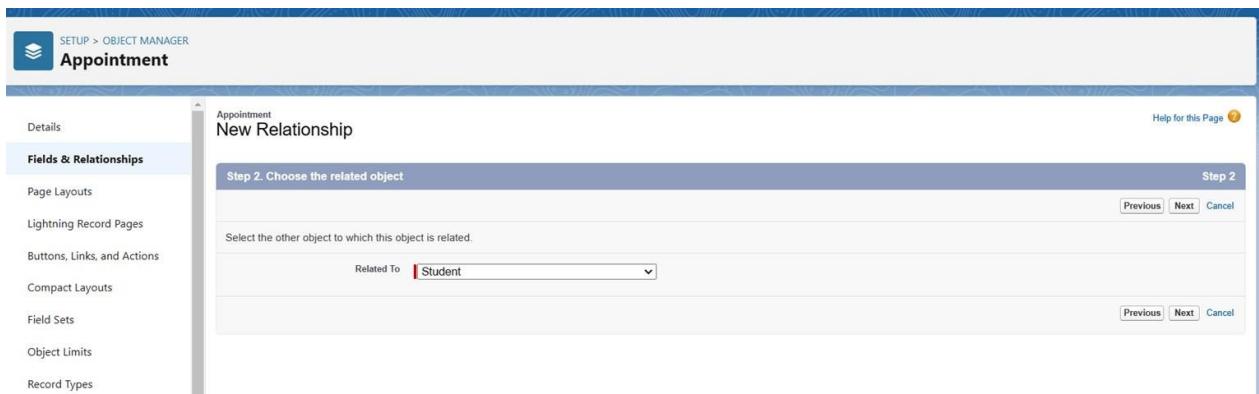
The screenshot shows the Salesforce Object Manager interface for the 'Appointment' object. The left sidebar lists various setup options like Details, Page Layouts, Lightning Record Pages, etc. The 'Fields & Relationships' tab is currently selected. On the right, there's a list of field types: Auto Number, Formula, Roll-Up Summary, and Lookup Relationship. The 'Lookup Relationship' option is selected and highlighted with a red box. A detailed description of what it does follows:

Creates a relationship that links this object to another object. The relationship field allows users to click on a lookup icon to select a value from a popup list. The other object is the source of the values in the list.

- This field type is available on all detail records.
- The ownership and sharing of a detail record are determined by the master record.
- When a user deletes the master record, all detail records are deleted.
- You can create rollup summary fields on the master record to summarize the detail records.

The relationship field allows users to click on a lookup icon to select a value from a popup list. The master object is the source of the values in the list.

- b. Choose the related object for the relationship as **Student****



The screenshot shows the 'New Relationship' wizard for the 'Appointment' object. It's on 'Step 2: Choose the related object'. The left sidebar shows standard setup options like Details, Page Layouts, etc. The main area says 'Select the other object to which this object is related.' Below it is a dropdown labeled 'Related To' with 'Student' selected. At the bottom right are 'Previous', 'Next', and 'Cancel' buttons.

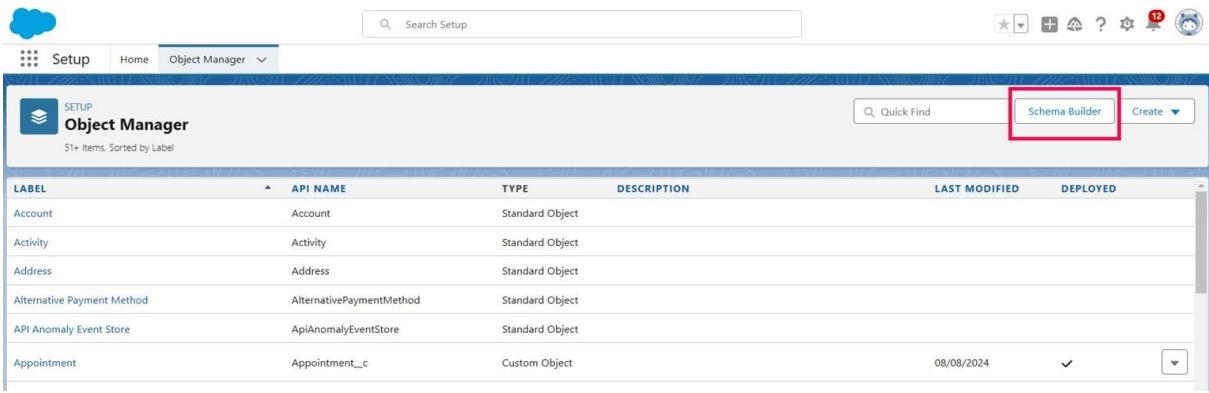


2. In the same way create **Lookup relationship** between the following:

- **Appointment and Student Object**
- **Student and Case Object**

The Lookup relationship **Student** and **Case Object** is to manage student queries related to immigration or visa applications.

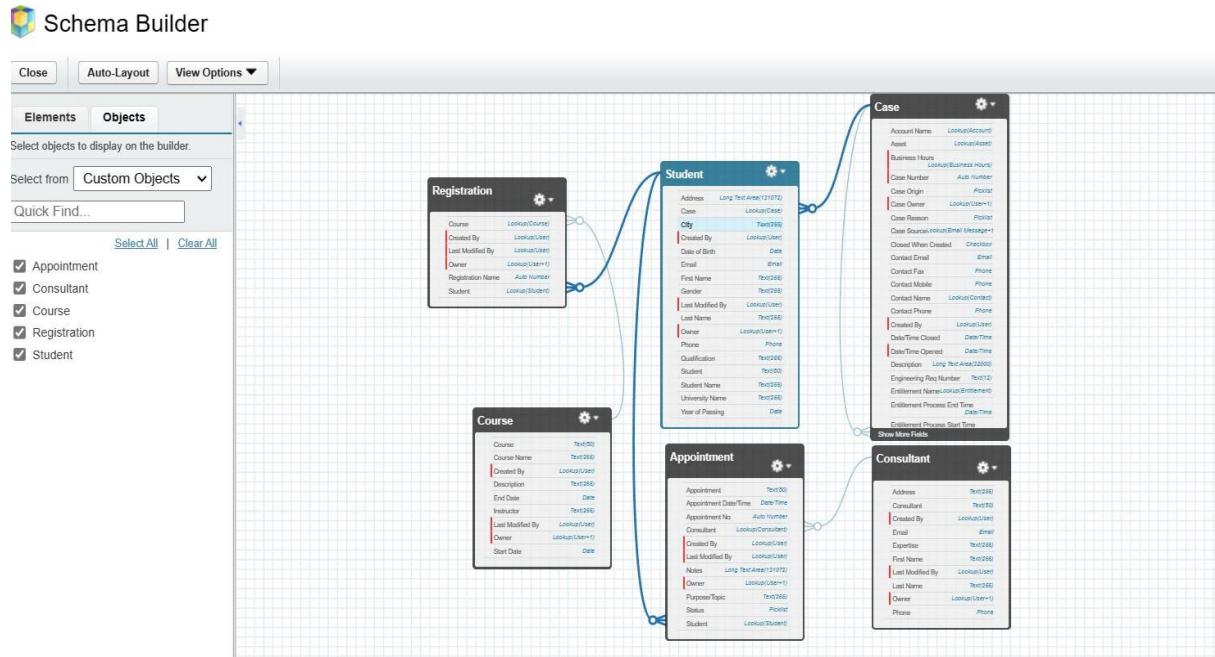
3. Go to the Schema Builder to see the relationship between the objects.



The screenshot shows the Salesforce Object Manager interface. At the top, there's a search bar and various navigation icons. Below that is a header with 'SETUP' and 'Object Manager'. A red box highlights the 'Schema Builder' button in the top right corner of the header. The main area is a table listing various objects with columns for Label, API Name, Type, Description, Last Modified, and Deployed. The 'Student' object is listed under 'Custom Object'.

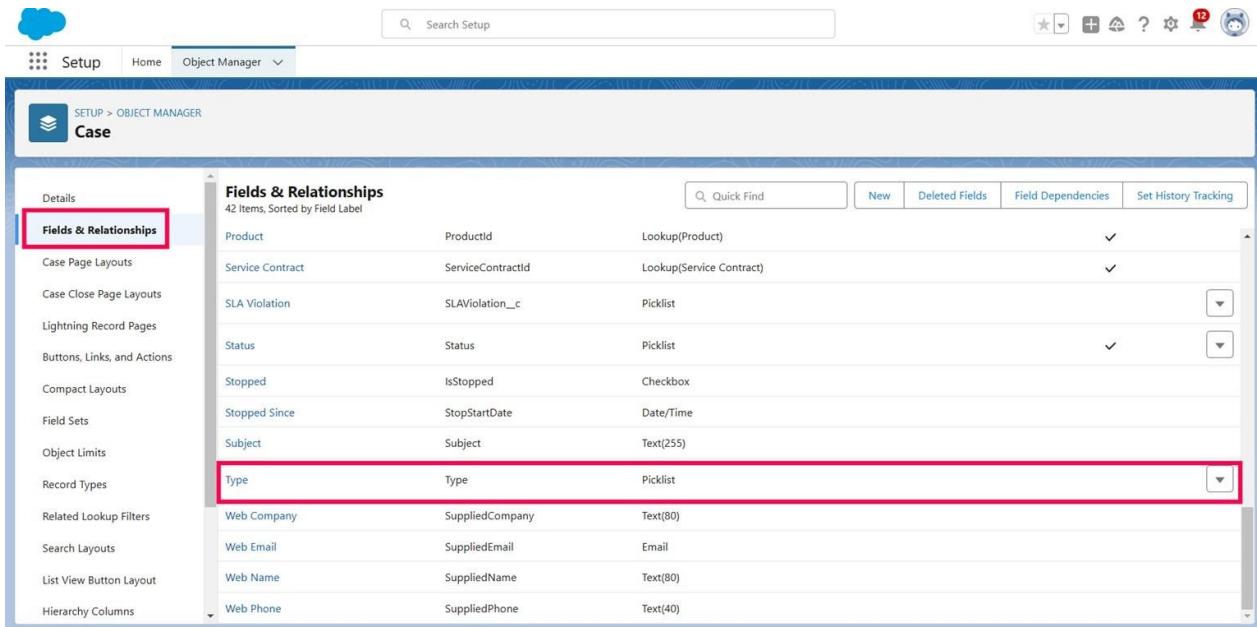
| LABEL | API NAME | TYPE | DESCRIPTION | LAST MODIFIED | DEPLOYED |
|----------------------------|--------------------------|-----------------|-------------|---------------|----------|
| Account | Account | Standard Object | | | |
| Activity | Activity | Standard Object | | | |
| Address | Address | Standard Object | | | |
| Alternative Payment Method | AlternativePaymentMethod | Standard Object | | | |
| API Anomaly Event Store | ApiAnomalyEventStore | Standard Object | | | |
| Appointment | Appointment_c | Custom Object | | 08/08/2024 | ▼ |

4. The data model will look like this:



Subtask 3: Configure the Case Object

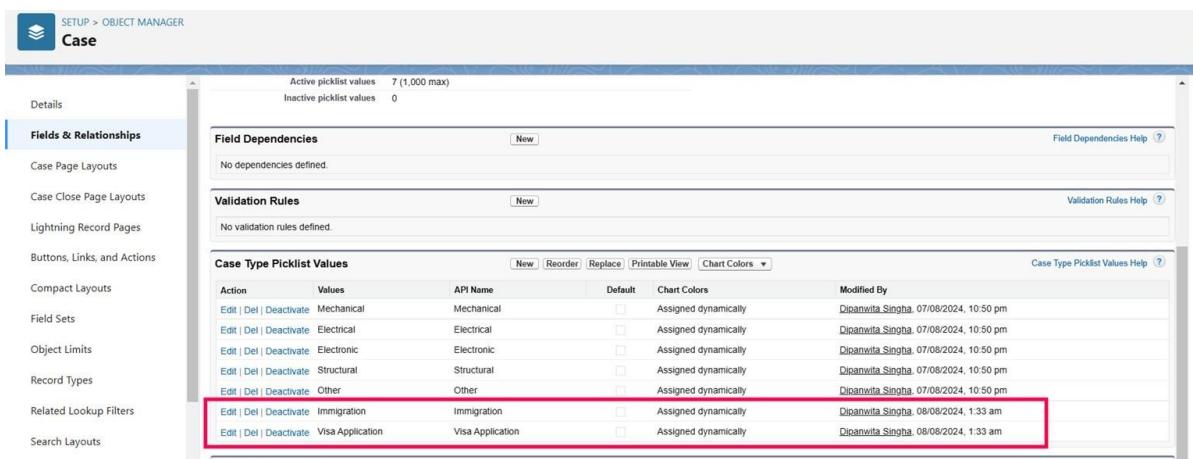
1. Navigate to **Object Manager**, then select the **Case** object.
2. Edit the **Type** field and add the following values:
 - **Immigration**
 - **Visa Application**



SETUP > OBJECT MANAGER
Case

Fields & Relationships
42 items. Sorted by Field Label

| Product | ProductId | Lookup(Product) | ✓ |
|------------------|-------------------|--------------------------|----------|
| Service Contract | ServiceContractId | Lookup(Service Contract) | ✓ |
| SLA Violation | SLAViolation_c | Picklist | ▼ |
| Status | Status | Picklist | ▼ |
| Stopped | IsStopped | Checkbox | ▼ |
| Stopped Since | StopStartDate | Date/Time | ▼ |
| Subject | Subject | Text(255) | ▼ |
| Type | Type | Picklist | ▼ |
| Web Company | SuppliedCompany | Text(80) | ▼ |
| Web Email | SuppliedEmail | Email | ▼ |
| Web Name | SuppliedName | Text(80) | ▼ |
| Web Phone | SuppliedPhone | Text(40) | ▼ |



SETUP > OBJECT MANAGER
Case

Field Dependencies
Active picklist values: 7 (1,000 max)
Inactive picklist values: 0

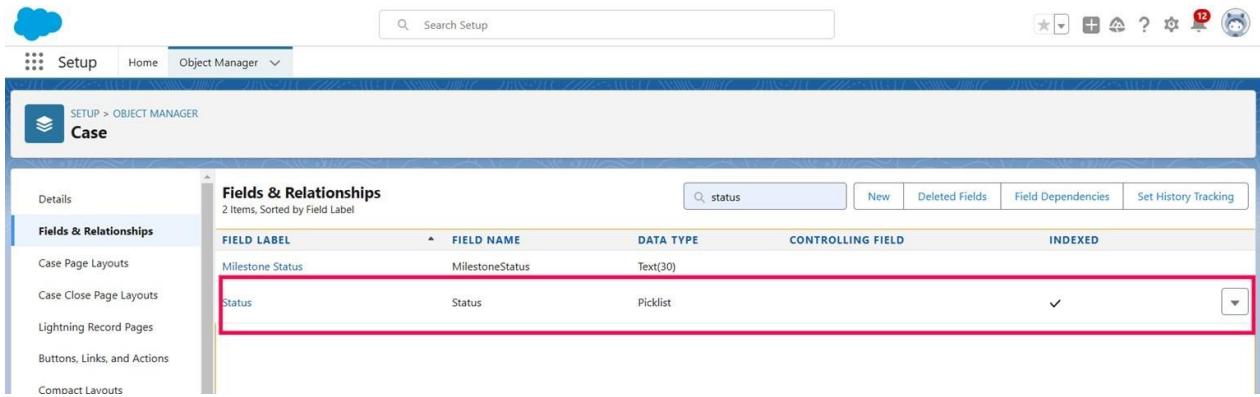
Validation Rules
No validation rules defined.

Case Type Picklist Values

| Action | Values | API Name | Default | Chart Colors | Modified By |
|-------------------------|-------------------------|------------------|--------------------------|----------------------|--|
| Edit Del Deactivate | Mechanical | Mechanical | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Electrical | Electrical | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Electronic | Electronic | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Structural | Structural | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Other | Other | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Immigration | Immigration | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 08/08/2024, 1:33 am |
| Edit Del Deactivate | Visa Application | Visa Application | <input type="checkbox"/> | Assigned dynamically | Dipanwita.Singha, 08/08/2024, 1:33 am |

3. Edit the **Status** field and add the following values:

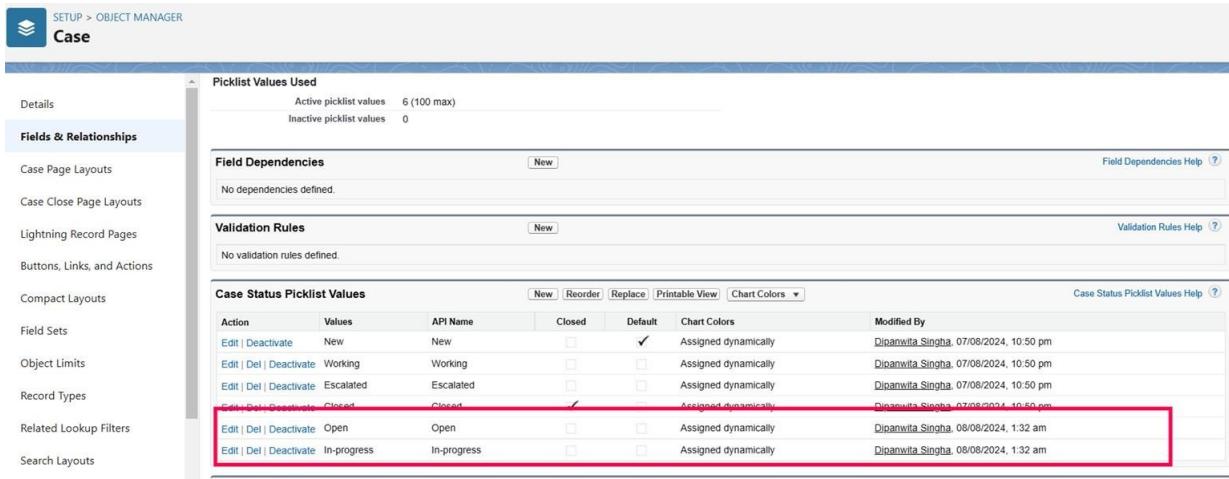
- **Open**
- **In-progress**



SETUP > OBJECT MANAGER
Case

Fields & Relationships
2 Items, Sorted by Field Label

| FIELD LABEL | FIELD NAME | DATA TYPE | CONTROLLING FIELD | INDEXED |
|------------------|-----------------|-----------|-------------------|---------|
| Milestone Status | MilestoneStatus | Text(30) | | |
| Status | Status | Picklist | | ✓ |



SETUP > OBJECT MANAGER
Case

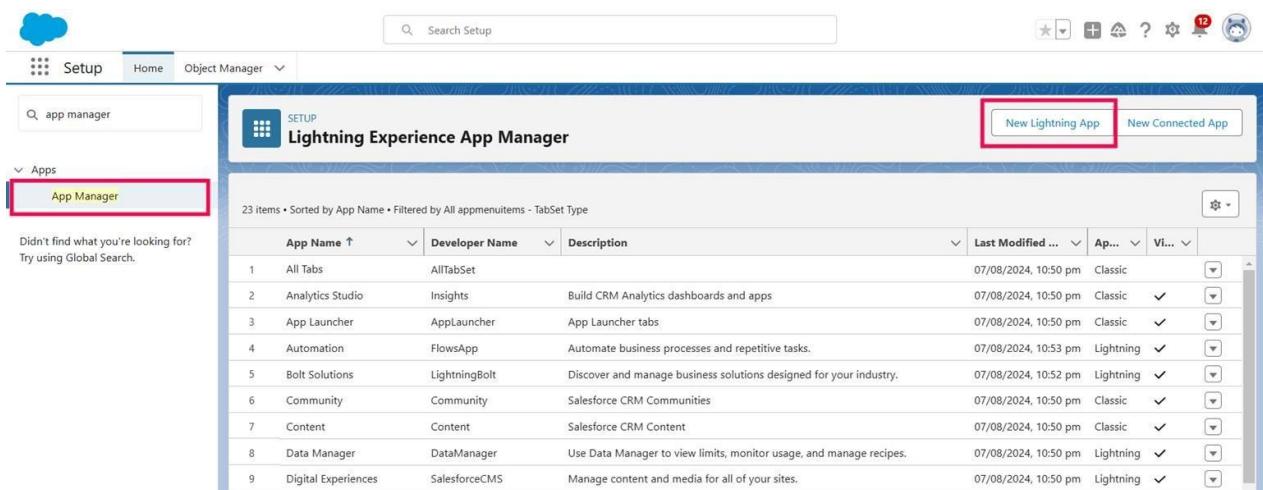
Fields & Relationships

Case Status Picklist Values

| Action | Values | API Name | Closed | Default | Chart Colors | Modified By |
|-------------------------|-------------|-------------|-------------------------------------|-------------------------------------|----------------------|---------------------------------------|
| Edit Deactivate | New | New | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Assigned dynamically | Dipanwita Singha 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Working | Working | <input type="checkbox"/> | <input type="checkbox"/> | Assigned dynamically | Dipanwita Singha 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Escalated | Escalated | <input type="checkbox"/> | <input type="checkbox"/> | Assigned dynamically | Dipanwita Singha 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Closed | Closed | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Assigned dynamically | Dipanwita Singha 07/08/2024, 10:50 pm |
| Edit Del Deactivate | Open | Open | <input type="checkbox"/> | <input type="checkbox"/> | Assigned dynamically | Dipanwita Singha 08/08/2024, 1:32 am |
| Edit Del Deactivate | In-progress | In-progress | <input type="checkbox"/> | <input type="checkbox"/> | Assigned dynamically | Dipanwita Singha 08/08/2024, 1:32 am |

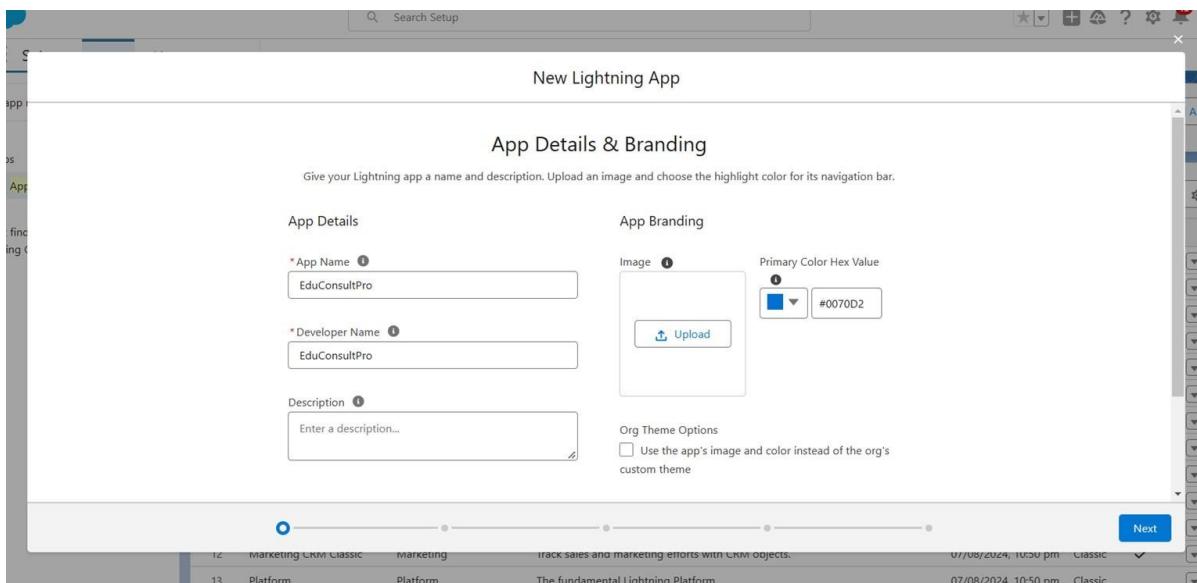
Subtask 4: Create a Lightning App

1. In Setup, search for App Manager in the Quick Find bar.
2. Click on New Lightning App.



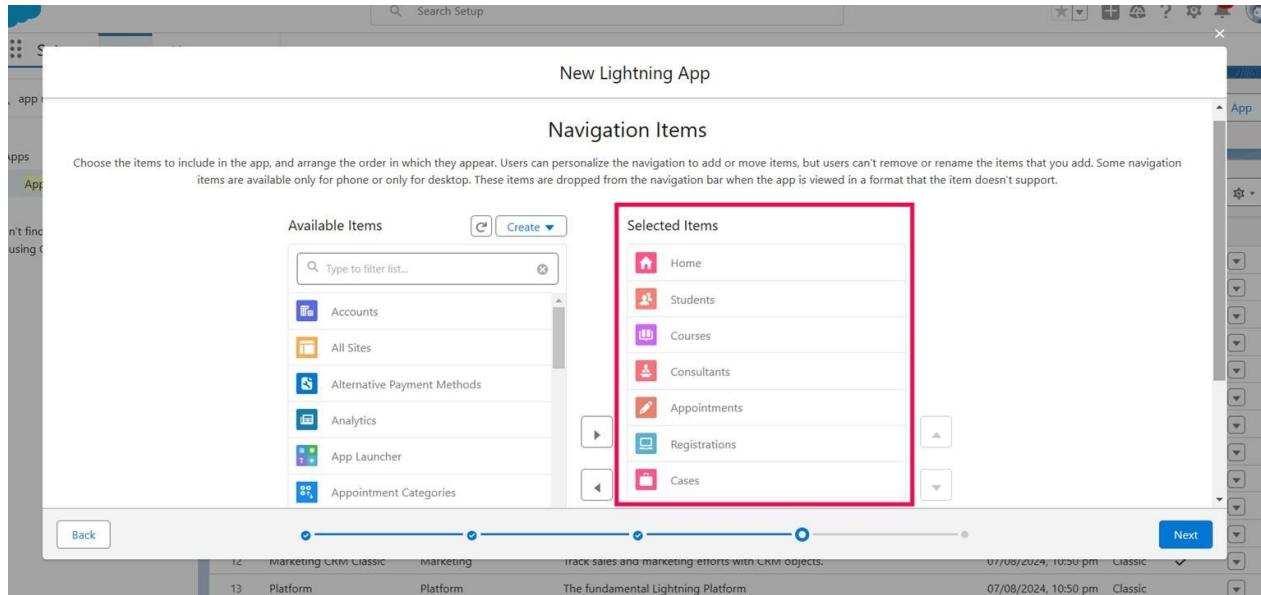
The screenshot shows the Salesforce Setup interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The 'Search Setup' bar contains the query 'app manager'. On the left, a sidebar under 'Apps' has 'App Manager' highlighted with a red box. The main content area is titled 'Lightning Experience App Manager' and displays a table of existing apps. A red box highlights the 'New Lightning App' button in the top right corner of the content area.

3. Name the app **EduConsultPro**.

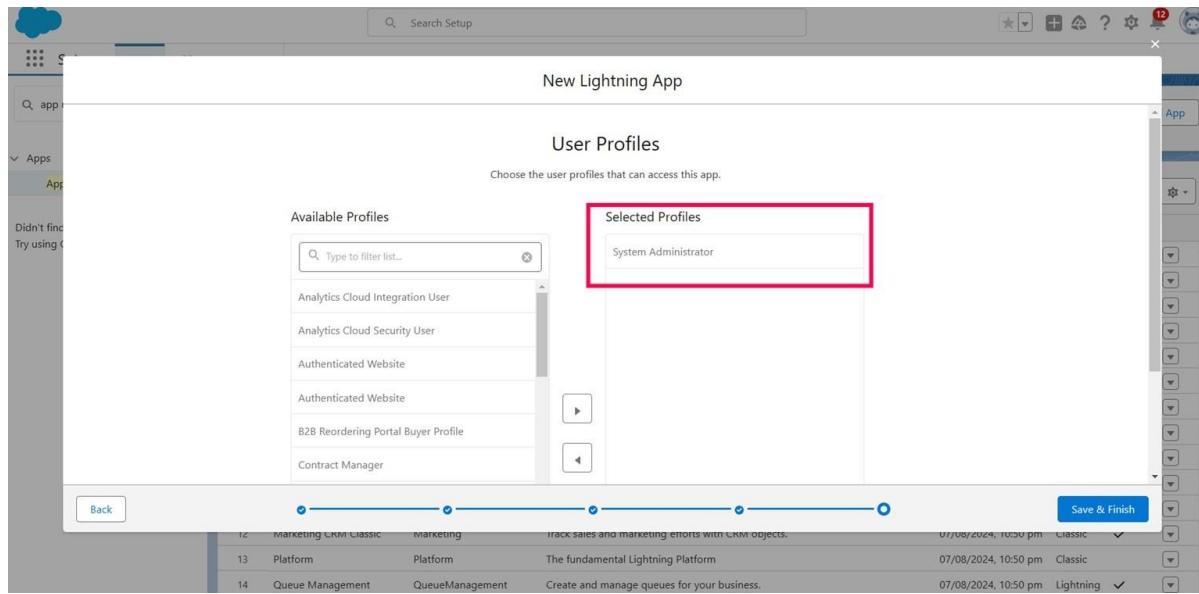


The screenshot shows the 'New Lightning App' configuration page. The title is 'New Lightning App'. The 'App Details & Branding' section contains fields for 'App Name' (set to 'EduConsultPro'), 'Developer Name' (set to 'EduConsultPro'), and 'Description' (with placeholder 'Enter a description...'). The 'App Branding' section includes an 'Image' field with an 'Upload' button and a color picker set to '#0070D2', and an 'Org Theme Options' checkbox. At the bottom, there is a progress bar showing steps 12 and 13 completed, and a 'Next' button.

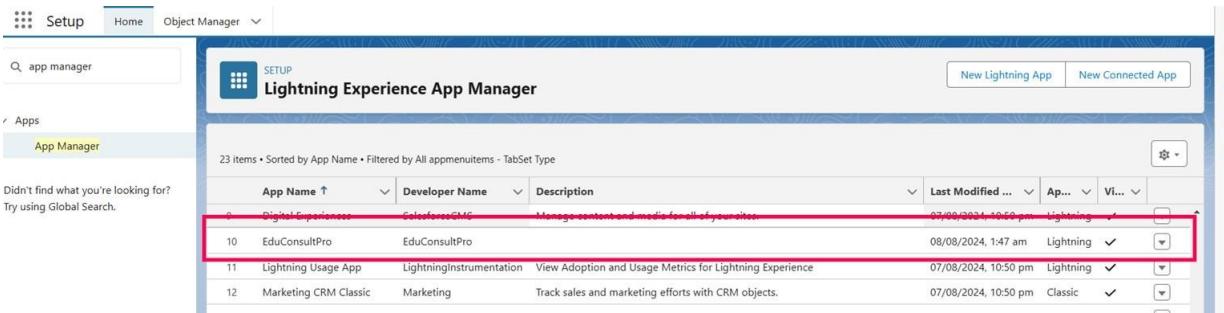
4. Add the following items from the Available Items list to the Selected Items list:
- o **Home, Students, Courses, Consultants, Appointments, Registrations, Cases**



5. Add the **System Administrator** profile from Available Profiles to Selected Profiles and click **Save & Finish**.



6. EduConsultPro app is created.



The screenshot shows the Salesforce App Manager interface. The left sidebar has 'Setup' selected, followed by 'Home' and 'Object Manager'. Under 'Apps', 'App Manager' is selected. A search bar at the top says 'app manager'. The main area is titled 'Lightning Experience App Manager' and displays a table of apps. The table has columns: App Name, Developer Name, Description, Last Modified, Type, and Status. There are 23 items listed. The 'EduConsultPro' app is highlighted with a red box. Its details are as follows:

| App Name | Developer Name | Description | Last Modified | Type | Status |
|-----------------------|--------------------------|--|----------------------|-----------|--------|
| EduConsultPro | EduConsultPro | Manages consulting projects and clients. | 08/08/2024, 1:47 am | Lightning | ✓ |
| Lightning Usage App | LightningInstrumentation | View Adoption and Usage Metrics for Lightning Experience | 07/08/2024, 10:50 pm | Lightning | ✓ |
| Marketing CRM Classic | Marketing | Track sales and marketing efforts with CRM objects. | 07/08/2024, 10:50 pm | Classic | ✓ |

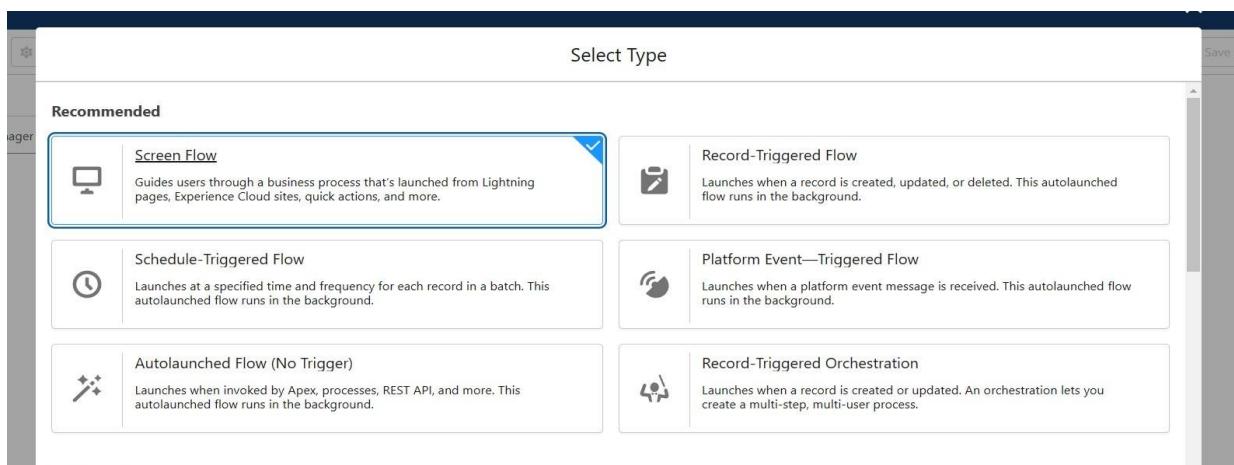
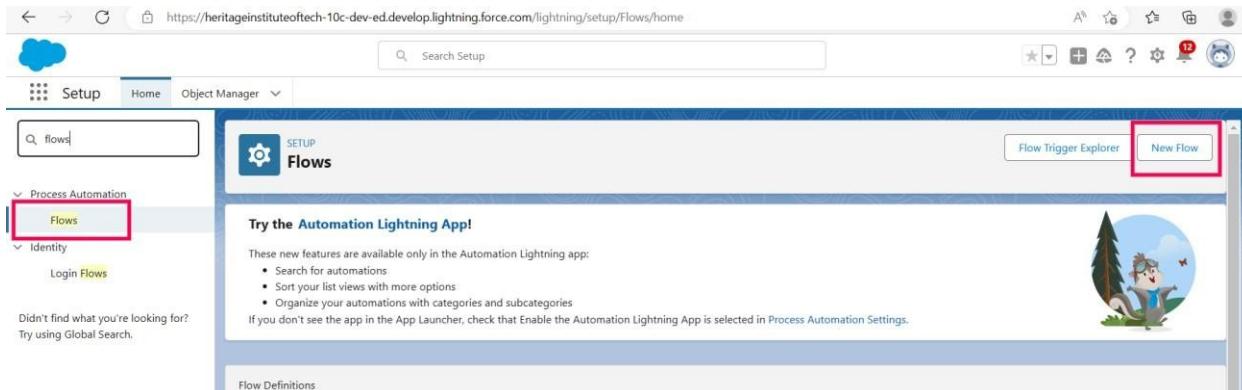
TASK 4: Create A ScreenFlow For Student Admission Application Process

Design a ScreenFlow to automate the student admission process, including data capture, record creation, course selection, and email notifications.

Subtask 1: Add Screen Element

Set up a screen to capture student information.

1. Navigate to **Setup**, enter **Flow** in the Quick Find box, and select **New Flow**. Choose **ScreenFlow**.

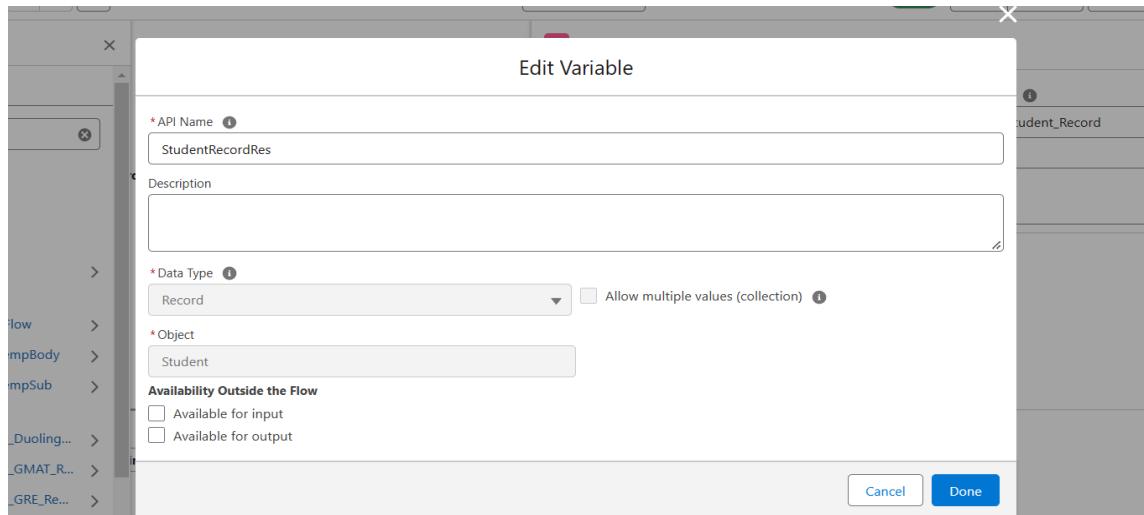
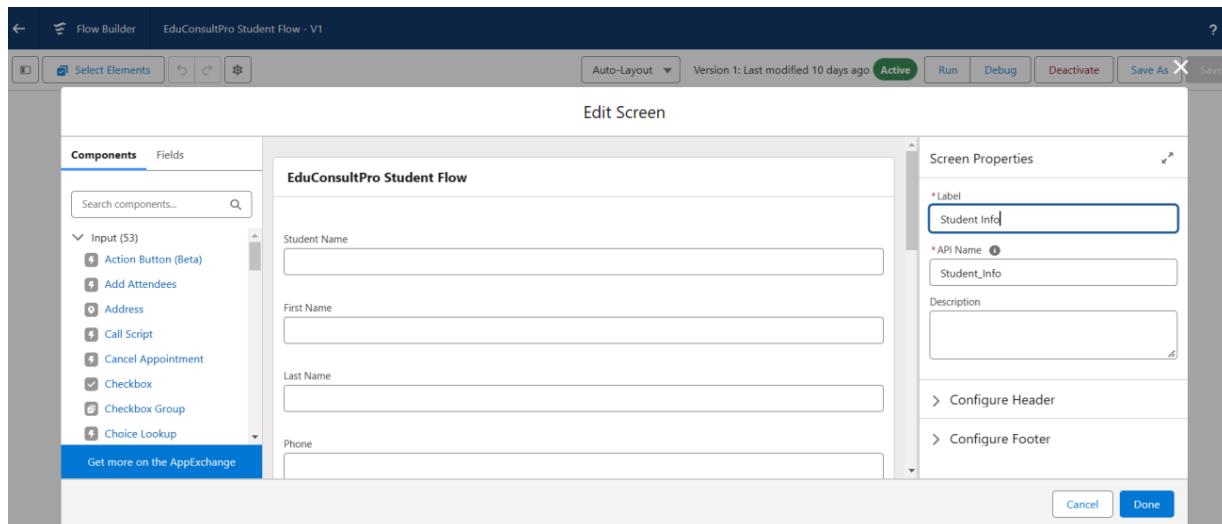




2. Add a **Screen** element.

3. In the **Screen Properties** pane:

- **Label:** Enter “Student Info”.
- **Fields:** Create a new **Resource** (StudentRecordRes) to display all fields from the **Student** object.
- **Action:** Drag and drop the necessary fields to collect student information on the screen like the Student Name, First Name, Last Name, Phone, Email, DoB, Address, City, Qualification.

Subtask 2: Create Student Record Using Create Element

Set up a process to create a student record from the captured information.

1. Add a **Create** element after the Student Info screen element.
2. **Label:** “Create Student Record”.
3. Select “One” under **How many records to Create**.
4. Choose “**Use all values from a record**” under **How to Set the record fields**.
5. Select the **StudentRecordRes** variable from the Student Info screen element under **Create a record from these values**.

 Create Records X

| | |
|--|--|
| * Label Create Student Record | * API Name  Create_Student_Record |
| Description <div style="border: 1px solid #ccc; height: 40px; margin-top: 5px;"></div> | |
| * How to set record field values From a Record Variable ▼ | |
| How Many Records to Create <input checked="" type="radio"/> One <input type="radio"/> Multiple | |
| Create a Record from These Values * Record  StudentRecordRes X | |

Make sure that ID is blank. After the flow creates the records, ID is set to match the record that was created. 

Subtask 3: Add Course Selection Screen

Provide a screen for students to select a course.

1. Add a **Screen** element after the Create Student Record element.
2. **Label:** “Course Screen”.

Edit Screen

Components Fields

▼ Input (3)

- Dependent Picklists
- Multi-Select Picklist
- Picklist

Get more on the AppExchange

EduConsultPro Student Flow

Select Course

--None--

Pause
Previous
Finish

Screen Properties

***Label**

***API Name**

Description

> Configure Header

> Configure Footer

Cancel
Done

3. Add a **Picklist** component:
 - **Label:** “Select Course”.
 - **Choices:** Enter “IELTS”, “GRE”, “GMAT”, “Duolingo”, and “TOEFL”.

Edit Screen

Components Fields

▼ Input (3)

- Dependent Picklists
- Multi-Select Picklist
- Picklist

Get more on the AppExchange

EduConsultPro Student Flow

Picklist

Select Course

--None--

Pause
Previous
Finish

Picklist

Component type

| | | | |
|---------------|---|--|--|
| Choice | <input type="text" value="({IELTS})"/> | Edit | Delete |
| Choice | <input type="text" value="({GRE})"/> | Edit | Delete |
| Choice | <input type="text" value="({GMAT})"/> | Edit | Delete |
| Choice | <input type="text" value="({Duolingo})"/> | Edit | Delete |
| Choice | <input type="text" value="({TOEFL})"/> | Edit | Delete |

Cancel
Done

Subtask 4: Add Decision Element

Define logic to determine which course has been selected.

1. Add a **Decision** element after the Course Screen.
2. **Label:** “Selecting Course”.
3. Define outcomes for each course:
 - **Outcome Label:** “Selected IELTS”.
 - **Condition:**
 - **Resource:** Select_Course (Screen Component from Select Course Screen Element)
 - **Operator:** Equals
 - **Value:** IELTS (Choice Variable from Select Course Screen Element)
 - Repeat for GRE, GMAT, Duolingo, TOEFL.

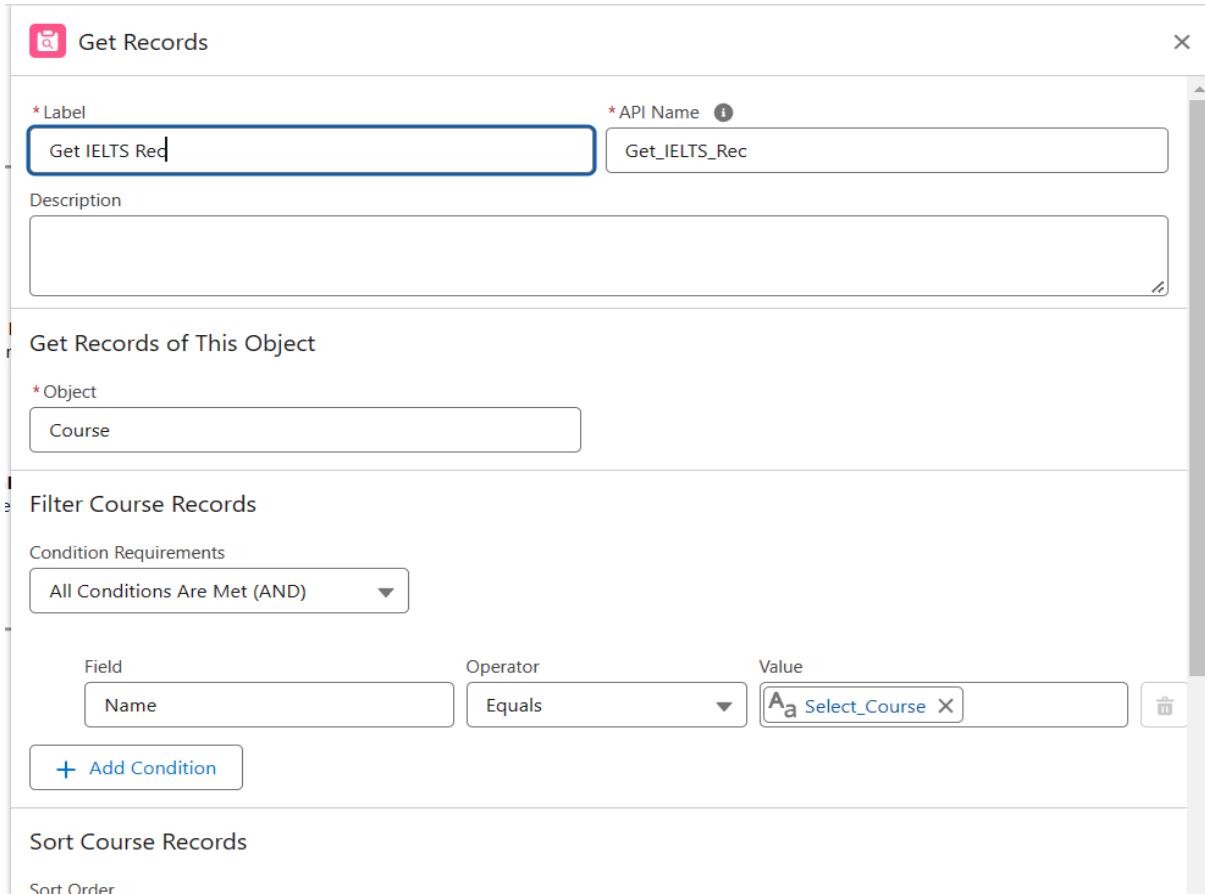
 **Decision** x

| | |
|---|--|
| * Label <input style="width: 100%;" type="text" value="Selecting Course"/> | * API Name (i) <input style="width: 100%;" type="text" value="Selecting_Course"/> |
| Description <div style="border: 1px solid #ccc; height: 40px; width: 100%;"></div> | |
| Outcomes For each path the flow can take, create an outcome. For each outcome, specify the conditions that must be met for the flow to take that path. | |
| OUTCOME ORDER (i) + | OUTCOME DETAILS <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> * Label <input style="width: 100%;" type="text" value="Selected IELTS"/>* Outcome API Name (i) <input style="width: 100%;" type="text" value="Selected_IELTS"/> </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> Condition Requirements to Execute Outcome All Conditions Are Met (AND) </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> Resource <input style="width: 100%;" type="text" value="A_a Course Screen > Select_Course"/> Operator <input style="width: 100%;" type="text" value="Equals"/> Value <input style="width: 100%;" type="text" value="A_a IELTS"/> </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> Default Outcome (i) + Add Condition </div> |

Subtask 5: Add GET Record Element

Retrieve course details based on the selected course.

1. Add a **GET Record** element after each Decision path (IELTS, GRE, GMAT, etc.).
2. **Label:** “Get [Course Name] Rec”.
3. **Object:** Course
4. **Condition Requirement:** All Conditions are Met (AND)
5. **Field:** Course Name
6. **Operator:** Equals
7. **Value:** {!Select_Course}



The screenshot shows the configuration of a 'Get Records' element. The 'Label' is set to 'Get IELTS Rec' and the 'API Name' is 'GetIELTS_Rec'. The 'Object' is set to 'Course'. The 'Condition Requirements' dropdown is set to 'All Conditions Are Met (AND)'. A single condition is defined with the 'Field' being 'Name', the 'Operator' being 'Equals', and the 'Value' being '{!Select_Course}'.

Subtask 6: Create Registration Record Using Create Records Element

Create a registration record for the selected course.

1. Add a **Create** element after each GET Record element.
2. **Label:** “Create [Course Name] Registration Rec”.
3. **How many records to Create:** “One”.
4. **How to Set the record fields:** Use separate resources, and literal values.
5. **Object:** Registration
6. Set fields:
 - **Course_Name_c:** {!Get_[Course Name]_Rec.Id}
 - **Student_Name_c:** {!StudentRecordRes.Id}

 Create Records X

| | |
|--|--|
| *Label | *API Name <small>i</small> |
| <input type="text" value="Create IELTS Registration Rec"/> | <input type="text" value="Create_IELTS_Registration_Rec"/> |

Description

***How to set record field values**

Manually ▼

Create a Record of This Object

***Object**

Registration

Set Field Values for the Registration

| | |
|--|--|
| Field | Value |
| <input type="text" value="Course_c"/> | ← <input type="text" value="A_a Course from GetIELTS_Rec > Record ID X"/> ✖ trash |
| Field | Value |
| <input type="text" value="Student_c"/> | ← <input type="text" value="A_a StudentRecordRes > Record ID X"/> ✖ trash |

+ Add Field

Subtask 7: Create Email Text Template Variables

Set up email templates for student registration confirmation.

1. Click the **toggle toolbox** on the left corner, select **New Resource**, and choose **Text Template**.
2. **API Name:** “StuRegistrationEmailTextTempBody”.
3. **Type:** View as plain text.
4. **Body:** Write a registration confirmation text.

Edit Text Template

* API Name i

Description

* Body i

Insert a resource... View as Plain Text ▾

Dear {!StudentRecordRes.Name},

Congratulations and welcome to EduConsultantPro!

We are delighted to inform you that your registration on our platform has been successfully completed. You are now part of our esteemed community dedicated to empowering students like you to achieve their educational and immigration aspirations.

At EduConsultantPro, we understand the importance of your academic and career goals, and we are committed to providing you with the highest level of support and guidance throughout your journey.

Cancel Done



5. Repeat to create an email text template for the subject,
API Name: “StuRegistrationEmailTextTempSub”.

Edit Text Template

* API Name ⓘ

Description

* Body ⓘ

Insert a resource...

Dear {!StudentRecordRes.Name},

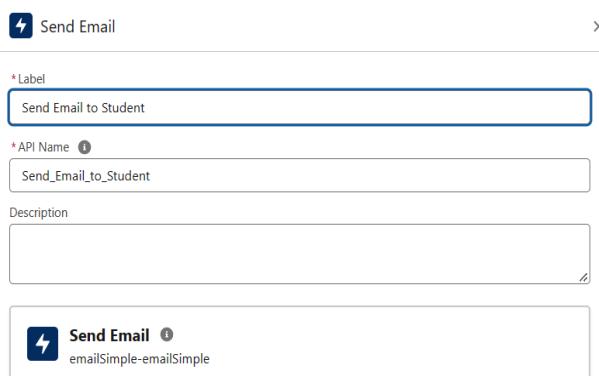
Congratulations and welcome to EduConsultantPro!

We are delighted to inform you that your registration on our platform has been successfully completed. You are now part of our esteemed community dedicated to empowering students like you to achieve their educational and immigration aspirations.

Subtask 8: Add Action Element

Configure the action to send an email to the student.

1. Add an **Action** element after all Decision paths.
2. **Label:** “Send Email to Student”.
3. **Input Values:**
 - **Body:** {!StuRegistrationEmailTextTempBody}
 - **Recipient Address List:** {!StudentRecordRes.Email_c}
 - **Subject:** {!StuRegistrationEmailTextTempSub}



***Label**
Send Email to Student

***API Name** [?](#)
Send_Email_to_Student

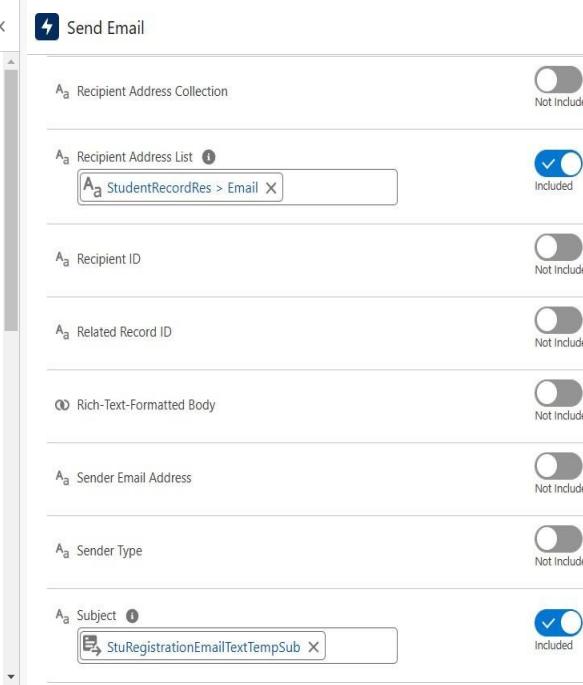
Description

Send Email [?](#)
emailSimple-emailSimple

Use values from earlier in the flow to set the inputs for the “Send Email” core action. To use its outputs later in the flow, store them in variables.

Set Input Values for the Selected Action

| | |
|---|--|
| ○○ Add Threading Token to Body | <input type="checkbox"/> Not Included |
| ○○ Add Threading Token to Subject | <input type="checkbox"/> Not Included |
| A_a Body ? | <input checked="" type="checkbox"/> Included |
| <input type="text"/> StuRegistrationEmailTextTempBody X | |



A_a Recipient Address Collection Not Include

A_a Recipient Address List [?](#)
 A_a StudentRecordRes > Email X [X](#)

A_a Recipient ID Not Include

A_a Related Record ID Not Include

○○ Rich-Text-Formatted Body Not Include

A_a Sender Email Address Not Include

A_a Sender Type Not Include

A_a Subject [?](#)
 StuRegistrationEmailTextTempSub X [X](#)

Included

Subtask 9: Add Success Screen

Display a confirmation message to the student after the registration.

1. Add a **Screen** element after the Send Email to Student Action Element.
2. **Label:** “Success Screen”.
3. Add a **Display Text** component:
 - **Label:** “SuccessMessage”.
 - **Text:** Write the success message.

Edit Screen

Components

Search components...

- ✓ Input (53)
 - Action Button (Beta)
 - Add Attendees
 - Address
 - Call Script
 - Cancel Appointment
 - Checkbox
 - Checkbox Group
 - Choice Lookup

Get more on the AppExchange

EduConsultPro Student Flow

Dear {!StudentRecordRes.Name},

Congratulations and welcome to EduConsultantPro!

We are delighted to inform you that your registration on our platform has been successfully completed. You are now part of our esteemed community dedicated to empowering students like you to achieve their educational and immigration aspirations.

Your Registration details have been sent through mail kindly check it once.

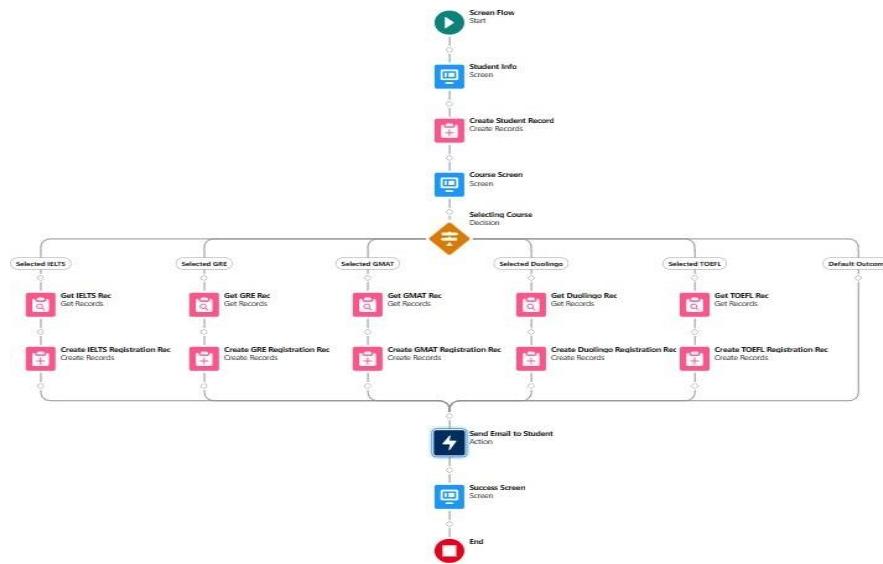
Thank you.

Pause
Previous
Finish

Screen Properties

| | |
|------------------|---|
| * Label | <input type="text" value="Success Screen"/> |
| * API Name | <input type="text" value="Success_Screen"/> |
| Description | <input type="text"/> |
| Configure Header | |
| Configure Footer | |

4. Save the flow and name it “EduConsultPro Student Flow”.



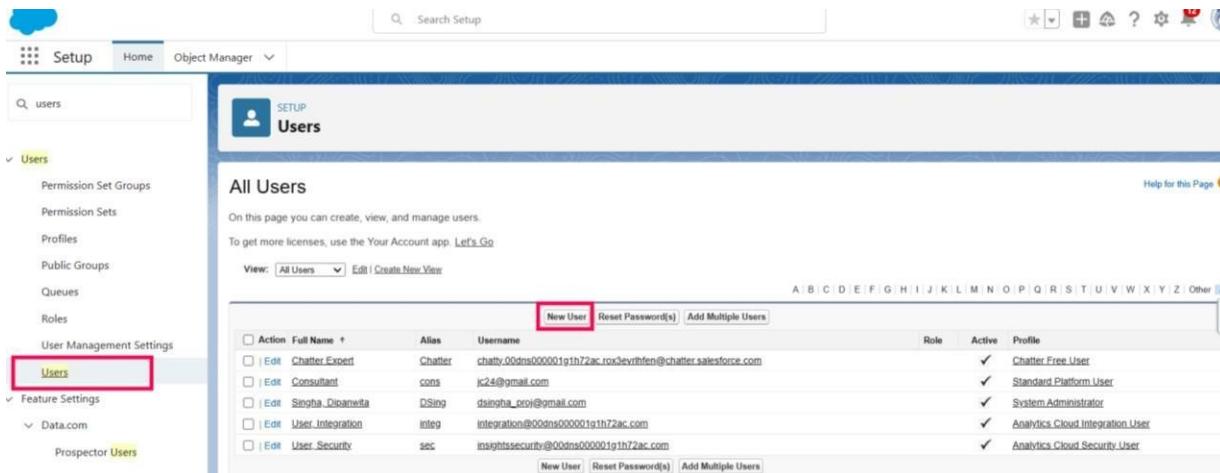
Task 5: Create Users

Set up a new user with the Standard Platform User profile and configure the user settings.

Subtask 1: Create a User

Create a new user with the Standard Platform User profile.

1. Navigate to **Setup > Users > New User**.

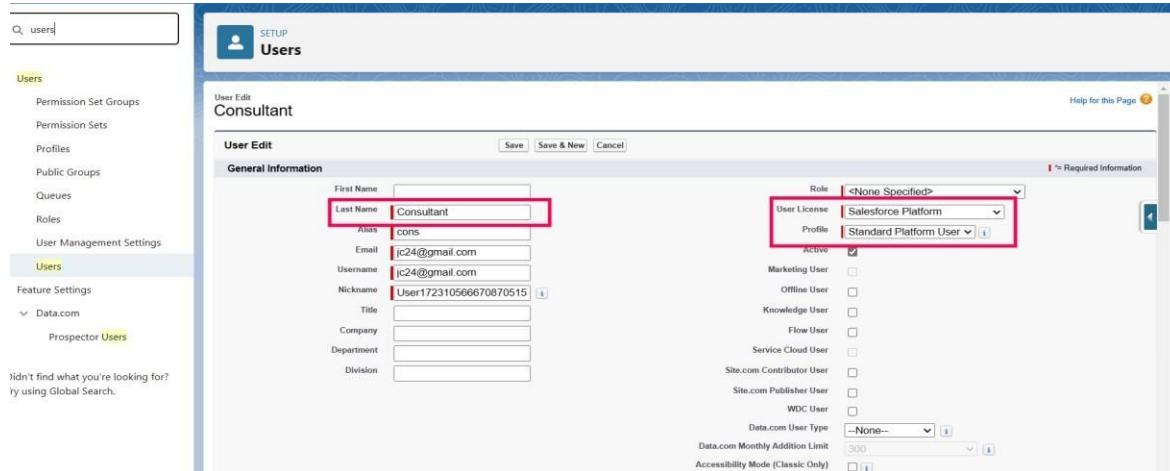


The screenshot shows the Salesforce Setup interface with the 'Users' page selected. The left sidebar includes links for Permission Set Groups, Profiles, Public Groups, Queues, Roles, User Management Settings (with 'Users' highlighted), Feature Settings, Data.com, and Prospector Users. The main content area displays a table of existing users with columns for Action, Full Name, Alias, Username, Role, Active status, and Profile. A red box highlights the 'New User' button at the top right of the table. The URL in the browser address bar is [https://**eu18**.salesforce.com/setup/objects/Users/list](#).

| Action | Full Name | Alias | Username | Role | Active | Profile |
|---|-------------------|---------|--|------|-------------------------------------|----------------------------------|
| <input type="checkbox"/> Edit | Chatter Expert | Chatter | chatty_00dns000001gth72ac.rox3evrhfen@chatter.salesforce.com | | <input checked="" type="checkbox"/> | Chatter Free User |
| <input type="checkbox"/> Edit | Consultant | cons | ic24@gmail.com | | <input checked="" type="checkbox"/> | Standard Platform User |
| <input type="checkbox"/> Edit | Sneatha_Dipamwita | DSing | sneatha_cro@gmail.com | | <input checked="" type="checkbox"/> | System Administrator |
| <input type="checkbox"/> Edit | User_Integration | Integ | integration@00dns000001gth72ac.com | | <input checked="" type="checkbox"/> | Analytics Cloud Integration User |
| <input type="checkbox"/> Edit | User_Security | sec | insightssecurity@00dns000001gth72ac.com | | <input checked="" type="checkbox"/> | Analytics Cloud Security User |

2. Enter the following details:
 - **Last Name:** Consultant
 - **License:** Salesforce Platform
 - **Profile:** Standard Platform User

3. Complete all mandatory fields as required and save.

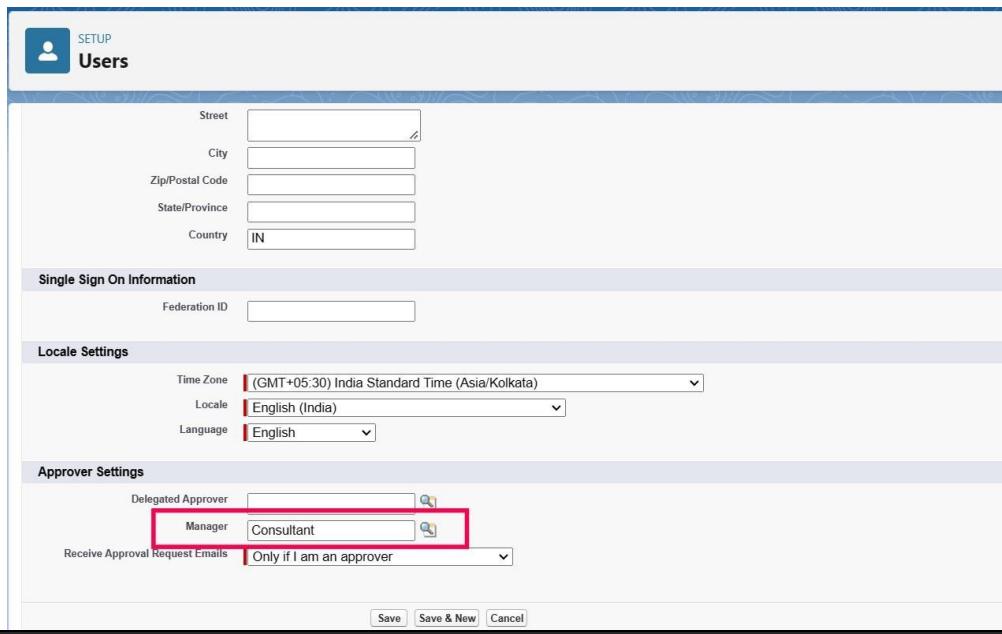


The screenshot shows the Salesforce Setup interface under the 'Users' section. A user named 'Consultant' is being edited. The 'Last Name' field and the 'Role' dropdown are both highlighted with red boxes, indicating they are the focus of configuration.

Subtask 2: Configure the User Settings

Update the user settings to include the appropriate approver settings.

1. Navigate to **Setup > Administration > Users** > Locate the newly created user and click **Edit** next to their name.
2. Scroll down to the **Approver Settings** section.
3. In the **Manager Field**, select “Consultant” and save.



The screenshot shows the 'Approver Settings' section of the User Edit screen. The 'Manager' field is highlighted with a red box, indicating it is the current focus of configuration.

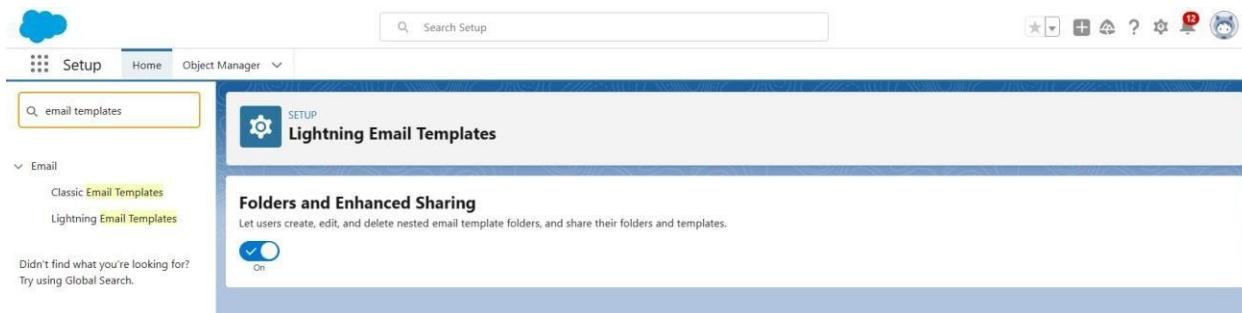
Task 6: Create an Approval Process for the Property Object

Establish an approval process for the Property object, including creating necessary email templates for notifications.

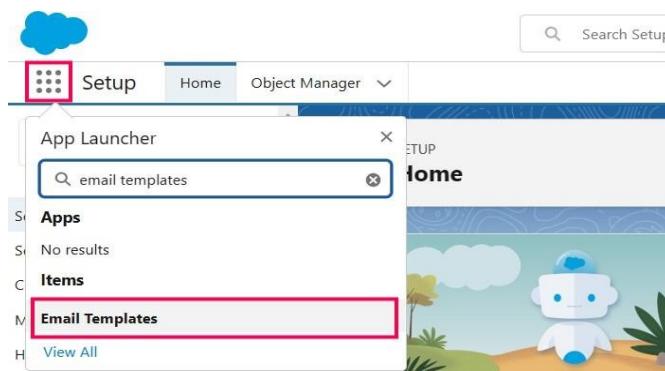
Subtask 1: Create Email Templates

Create and configure email templates to be used in the approval process.

1. Navigate to **Setup**, enter **Templates** in the Quick Find box, and select **Lightning Email Templates**. Toggle on.



2. Go to the **App Launcher**, search for **Email Templates**, and create a new folder with the desired name.





3. Create a new email template within the created folder:

- **Template Name:** Submission Template
- Write a message in the HTML Value

Edit Submission Template

* = Required Information

Information

| | |
|--|---|
| * Email Template Name | Related Entity Type |
| <input type="text" value="Submission Template"/> | <input type="text" value="-- None --"/> |
| Description | Folder |
| <input type="text"/> | <input type="text" value="EduConsultantpro_Email"/> Select Folder |

Message Content

| | |
|---|--|
| Subject | Enhanced Letterhead |
| <input type="text" value="Appointment Request with EduConsultantpro Consulta"/> | <input type="text" value="Search Enhanced Letterheads..."/>  |

HTML Value

Dear {{Appointment__c.Student_Name__c}},

I hope this email finds you well. I am writing to confirm the details of our upcoming appointment scheduled for {{Appointment__c.Appointment_DateTime__c}} regarding {{Appointment__c.PurposeTopic__c}}.

Appointment Details:
 Appointment No : {{Appointment__c.Name}},
 Student Name : {{Appointment__c.Student_Name__c}},
 Consultant Name : {{Appointment__c.Consultant__c}},
 Date & Time : {{Appointment__c.Appointment_DateTime__c}},
 Purpose : {{Appointment__c.PurposeTopic__c}}

I want to assure you that I am looking forward to our meeting and am fully prepared to address any questions or concerns you may have regarding {{Appointment__c.PurposeTopic__c}}. Your success and satisfaction are my top priorities, and I am committed to providing you with the guidance and support you need.

If you have any specific topics or questions you would like to discuss during our appointment, please feel free to share them with me in advance. This will help ensure that our time together is as productive and beneficial as possible.

If for any reason you need to reschedule or cancel our appointment, please notify me at your earliest convenience so that we can make alternative arrangements.

Once again, thank you for choosing to work with me on this matter. I am confident that our collaboration will

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



4. Create two additional email templates:

- **Approval Template:** Similar to the Submission Template but tailored for approval notifications.

- **Rejection Template:** Similar to the Submission Template but tailored for rejection notifications.

Edit Approval Request Template

* = Required Information

Information

* Email Template Name

Related Entity Type

-- None --

Description

Folder

EduConsultantpro_Email

[Select Folder](#)

Edit Rejection Request Template

* = Required Information

Information

* Email Template Name

Related Entity Type

-- None --

Description

Folder

EduConsultantpro_Email

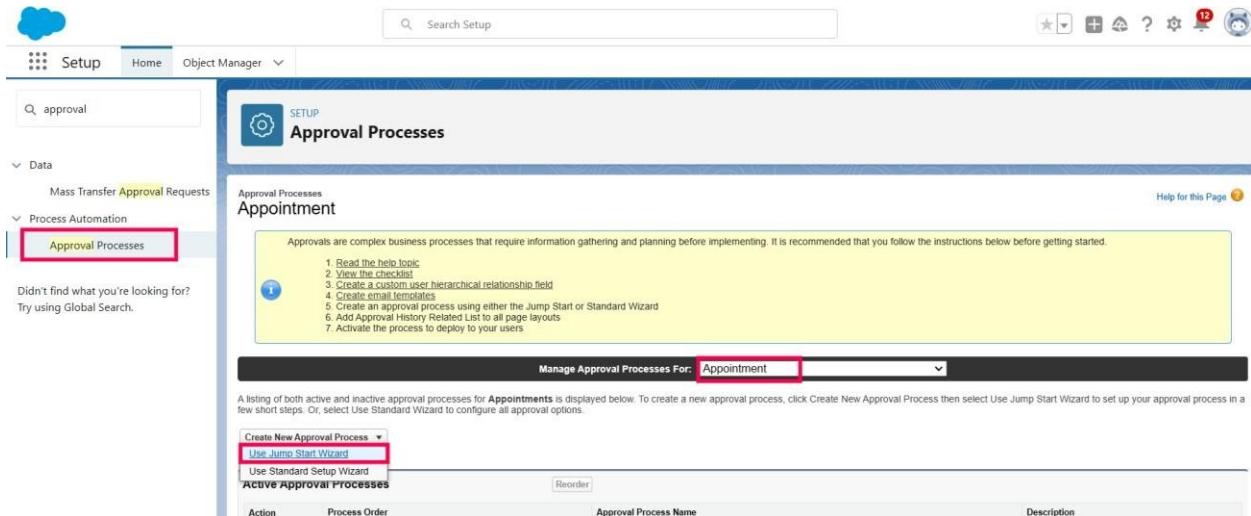
[Select Folder](#)

| Email Templates | | | | | | |
|-------------------------|----------------------------|-------------|------------------------|------------------|--------------------|--|
| Recent | | | | | | |
| 3 items | | | | | | |
| Email Templates | Email Template Name | Description | Folder | Last Modified By | Last Modified Date | |
| Recent | Approval Request Template | | EduConsultantpro_Email | Dipanwita Singha | 8/8/2024, 2:22 pm | |
| Created by Me | Submission Template | | EduConsultantpro_Email | Dipanwita Singha | 8/8/2024, 2:09 pm | |
| Private Email Templates | Rejection Request Template | | EduConsultantpro_Email | Dipanwita Singha | 8/8/2024, 2:21 pm | |
| Public Email Templates | | | | | | |

Subtask 2: Create An Approval Process

Configure an approval process for the Appointment object to manage request approvals.

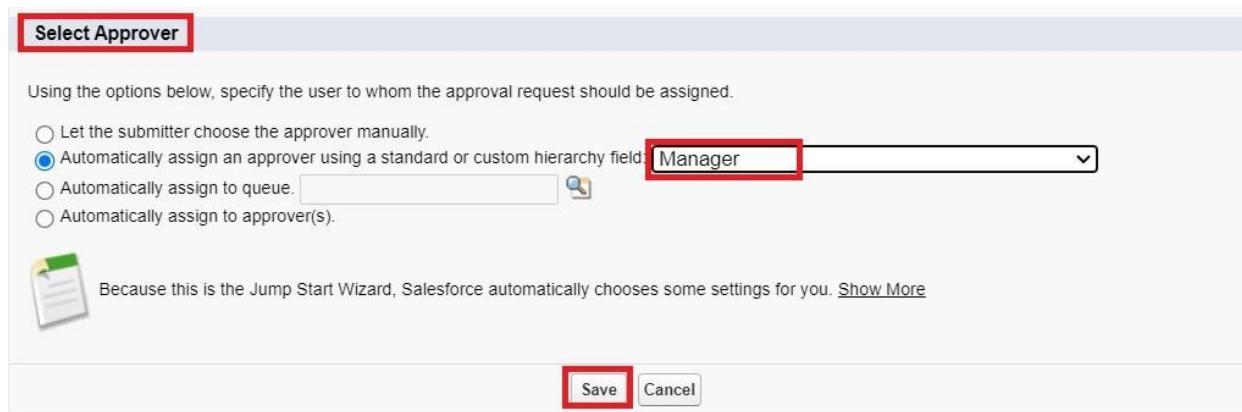
1. Navigate to **Setup**, enter **Approval** in the Quick Find box, and select **Approval Processes**.
2. In **Manage Approval Processes For**, select **Appointment**.
3. Click **Create New Approval Process** and choose **Use Jump Start Wizard**.



The screenshot shows the Salesforce Setup interface. The top navigation bar includes 'Search Setup', 'Home', 'Object Manager', and various icons. The left sidebar has sections for 'Data', 'Mass Transfer Approval Requests', and 'Process Automation', with 'Approval Processes' being the active item. A search bar at the top says 'approval'. The main content area is titled 'Approval Processes' and contains a help section with steps 1-7 for creating approval processes. Below this is a dropdown labeled 'Manage Approval Processes For: Appointment'. At the bottom, there's a list of 'Active Approval Processes' with a 'Create New Approval Process' button that has 'Use Jump Start Wizard' highlighted with a red box.

4. Configure the approval process:

- **Process Name:** Appointment Approval
- **Approver Selection:** Select **Manager** for “Automatically assign an approver using a standard or custom hierarchy field.”



The screenshot shows the 'Select Approver' configuration screen. It asks to specify the user to whom the approval request should be assigned. There are four options:

- Let the submitter choose the approver manually.
- Automatically assign an approver using a standard or custom hierarchy field. A dropdown menu is open, showing 'Manager' highlighted with a red box.
- Automatically assign to queue. A dropdown menu is open, showing a placeholder and a magnifying glass icon.
- Automatically assign to approver(s).

A note below states: "Because this is the Jump Start Wizard, Salesforce automatically chooses some settings for you. [Show More](#)". At the bottom are 'Save' and 'Cancel' buttons, both highlighted with red boxes.



5. Click **Next** and select **Manager** for the option **Automated Approver Determined By**.

6. Under **Record Editability Properties**, select **Administrators OR** the currently assigned approver can edit records during the approval process.
7. Click **Save**.

Approval Process Edit
Appointment Approval

Step 3. Specify Approver Field and Record Editability Properties

When you define approval steps, you can assign approval requests to different users. One of your options is to use a user approval steps, select a field from the picklist below. Also, when a record is in the approval process, it will always be locked by the currently assigned approver to edit the record.

Select Field Used for Automated Approval Routing

Next Automated Approver Determined By 
Use Approver Field of Appointment
Owner

Record Editability Properties

- Administrators **ONLY** can edit records during the approval process.
 Administrators **OR** the currently assigned approver can edit records during the approval process.

8. Click **View Approval Process Detail Page**.



9. Under **Initial Submission Actions**, click **Add New**:

- Select **Field Update**:
 - **Field to Update:** Appointment: Status
 - **Value:** Pending

Field Update Edit

Identification

| | |
|---|--|
| Name | Submitted |
| Unique Name | Submitted |
| Description | (empty) |
| Object | Appointment |
| Field to Update | Appointment: Status |
| Field Data Type | Picklist |
| Re-evaluate Workflow Rules after Field Change | <input type="checkbox"/> i |

Specify New Field Value

Picklist Options

The value above the current one
 The value below the current one
 A specific value [Pending](#)

Buttons: Save | Save & New | Cancel

10. Click **Add New**:

- Select **Email Alert**:
 - **Description:** Submission Email Alert
 - **Email Template:** Submission Template
 - **Recipient Type:** Select your Name

Email Alert Edit

Edit Email Alert

| | |
|---------------------|--------------------------|
| Description | Submission Email Alert |
| Unique Name | Submission_Email_Alert |
| Object | Appointment |
| Email Template | Submission Template |
| Protected Component | <input type="checkbox"/> |

Recipients

Search: for: Find

| Available Recipients | Selected Recipients |
|---|------------------------|
| User: Consultant User: Integration User User: Security User | User: Dipanwita Singha |

Add [▶](#) Remove [◀](#)

11. Repeat steps 9 and 10 for **Final Approval** and **Final Rejection** actions using the appropriate templates and values.

| Final Approval Actions  | | |
|--|--------------|--------------------------------------|
| Action | Type | Description |
| Edit | Record Lock | Lock the record from being edited |
| Edit Remove | Email Alert | Approval Email Alert |
| Edit Remove | Field Update | Approved |

| Final Rejection Actions  | | |
|---|--------------|---------------------------------------|
| Action | Type | Description |
| Edit | Record Lock | Unlock the record for editing |
| Edit Remove | Field Update | Rejected |
| Edit Remove | Email Alert | Rejection Email Alert |

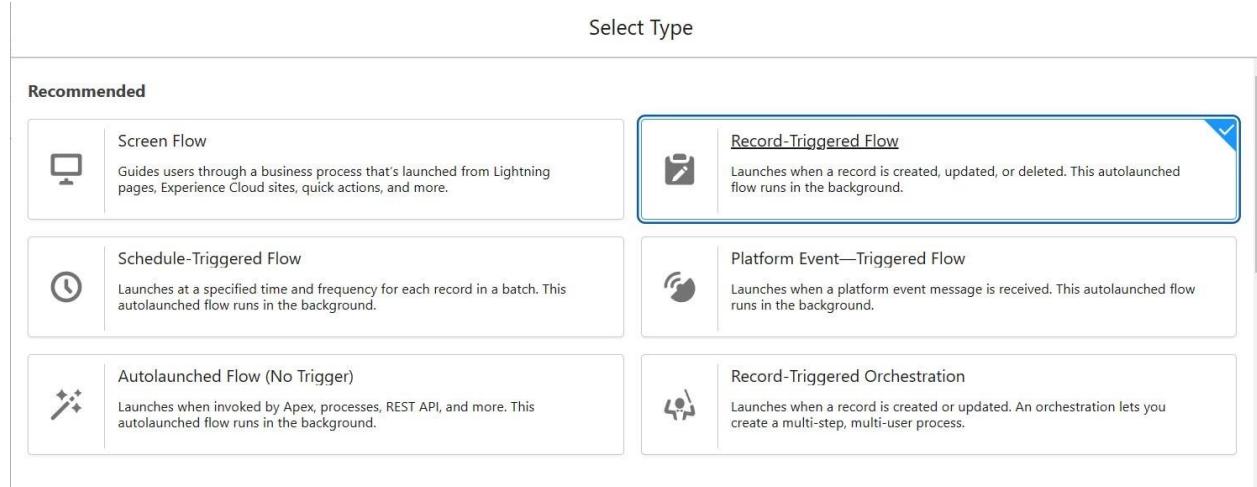
Task 7: Create A Record-Triggered Flow

Set up a Record-Triggered Flow to automate the submission of Appointment records for approval.

Subtask 1: Configure the Start Element

Initiate a flow that triggers upon the creation of an Appointment record.

1. From **Setup**, enter **Flows** in the Quick Find box, then select **Flows**.
2. Click **New Flow**.
3. Select **Record-Triggered Flow**.



The screenshot shows the 'Select Type' window in the Salesforce Flow setup. It lists several flow types under 'Recommended':

- Screen Flow**: Guides users through a business process that's launched from Lightning pages, Experience Cloud sites, quick actions, and more.
- Record-Triggered Flow**: (Selected) Launches when a record is created, updated, or deleted. This autolaunched flow runs in the background.
- Schedule-Triggered Flow**: Launches at a specified time and frequency for each record in a batch. This autolaunched flow runs in the background.
- Platform Event—Triggered Flow**: Launches when a platform event message is received. This autolaunched flow runs in the background.
- Autolaunched Flow (No Trigger)**: Launches when invoked by Apex, processes, REST API, and more. This autolaunched flow runs in the background.
- Record-Triggered Orchestration**: Launches when a record is created or updated. An orchestration lets you create a multi-step, multi-user process.

4. Click **Create**. The **Configure Start** window opens.
5. For **Object**, select **Appointment**.
6. For **Trigger the Flow When**, select **A record is created**.



Select Object

Select the object whose records trigger the flow when they're created, updated, or deleted.

* Object

Appointment

Configure Trigger

* Trigger the Flow When:

- A record is created
- A record is updated
- A record is created or updated
- A record is deleted

Set Entry Conditions

Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

Condition Requirements

None ▾

Subtask 2: Add An Action Element

Add an Action element to submit the Appointment record for approval automatically.

1. Add an **Action** element after the **Start** element.
2. Select the **Submit for approval** action and label it as **Approval SubFlow**.
3. Set the **RecordId** to **{!\$Record.Id}**.



Submit for Approval

* Label
Approval SubFlow

* API Name ⓘ
Approval_SubFlow

Description

Submit for Approval ⓘ
submit-submit

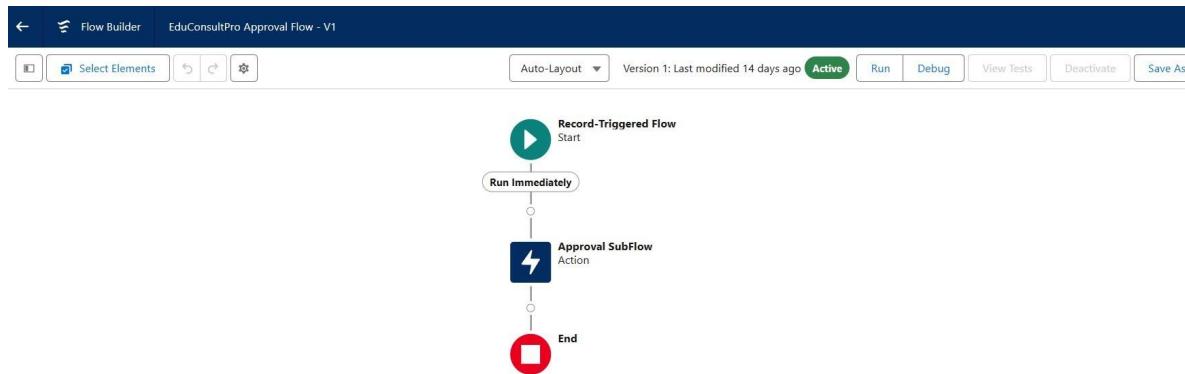
Use values from earlier in the flow to set the inputs for the "Submit for Approval" core action. To use its outputs later in the flow, store them in variables.

Set Input Values for the Selected Action

A_a * Record ID ⓘ
A_a Triggering Appointment__c > Record ID X

A_a Approval Process Name Or ID
Not Included

4. Save the Flow, label it as **EduConsultPro Approval Flow**, and click **Activate**.

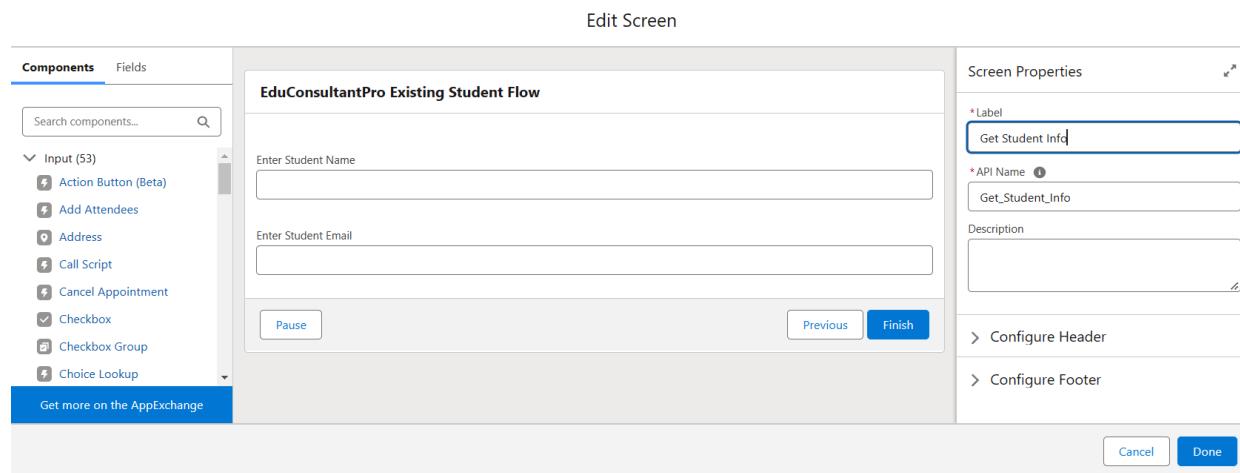


Task 8: Create A ScreenFlow For Existing Students To Book An Appointment

Subtask 1: Add Screen Element

Start the flow by gathering student information using a Screen element.

1. From **Setup**, enter **Flow Builder** in Quick Find and select **New Flow → ScreenFlow**.
2. Add a **Screen** element.
3. In the **Screen Properties** pane, for **Label**, enter “**Get Student Info**”.
4. Add two **Text** components from the left-side panel.
 - 1st Text Component Label: **Enter Student Name**
 - 2nd Text Component Label: **Enter Student Email**



5. Click **Done**.

Subtask 2: Get Record

1. Add a **GET Record** element after the Screen element, label it as “Get Rec”.
2. Configure the GET Record element:
 - **Select Object:** Student
 - **Condition Requirement:** All Conditions are Met (AND)
 - **Field:** Student Name
 - **Operator:** Equals
 - **Value:** {!Enter_Student_Name}
 - **Field:** Email_c
 - **Operator:** Equals
 - **Value:** {!Enter_Student_Email}

 **Get Records** X

| * Label <input type="text" value="Get Rec"/> | * API Name i <input type="text" value="Get_Rec"/> | | | | | | |
|--|--|--|----------|-------|---|---|--|
| Description <div style="border: 1px solid #ccc; height: 40px; margin-top: 5px;"></div> | | | | | | | |
| Get Records of This Object | | | | | | | |
| * Object <input type="text" value="Student"/> | | | | | | | |
| Filter Student Records | | | | | | | |
| Condition Requirements <div style="border: 1px solid #ccc; padding: 2px; width: fit-content;">All Conditions Are Met (AND) ▼</div> | | | | | | | |
| <table style="width: 100%; border-collapse: collapse;"> <tr> <th style="width: 30%;">Field</th> <th style="width: 30%;">Operator</th> <th style="width: 40%;">Value</th> </tr> <tr> <td><input type="text" value="Student_Name__c"/></td> <td>Equals ▼</td> <td><input type="text" value="Aa Enter_Student_Name"/> X Delete</td> </tr> </table> | | Field | Operator | Value | <input type="text" value="Student_Name__c"/> | Equals ▼ | <input type="text" value="Aa Enter_Student_Name"/> X Delete |
| Field | Operator | Value | | | | | |
| <input type="text" value="Student_Name__c"/> | Equals ▼ | <input type="text" value="Aa Enter_Student_Name"/> X Delete | | | | | |
| <table style="width: 100%; border-collapse: collapse;"> <tr> <th style="width: 30%;">Field</th> <th style="width: 30%;">Operator</th> <th style="width: 40%;">Value</th> </tr> <tr> <td>AND <input type="text" value="Email_c"/></td> <td>Equals ▼</td> <td><input type="text" value="Aa Enter_Student_Email"/> X Delete</td> </tr> </table> | | Field | Operator | Value | AND <input type="text" value="Email_c"/> | Equals ▼ | <input type="text" value="Aa Enter_Student_Email"/> X Delete |
| Field | Operator | Value | | | | | |
| AND <input type="text" value="Email_c"/> | Equals ▼ | <input type="text" value="Aa Enter_Student_Email"/> X Delete | | | | | |
| + Add Condition | | | | | | | |

Subtask 3: Add a Decision Element

1. Add a **Decision** element after the GET Record element, label it as “**Appointment or Case**”.
2. Configure the Decision element:
 - **Outcome Label:** Appointment
 - **Resource:** {!How_may_I_Help_you}
 - **Operator:** Equals
 - **Value:** {!Book_an_Appointment}
 - Click on the “+” icon to add paths for other options such as Case, Default.

Decision

| *Label | *API Name | | | | | | | | | | | | | | | | | | | | |
|--|---|---|---|----------------|-------------|-----------------------|----------------------------------|------|--|--|-----------------|---|--|--|---|---|---|--|---------------------------------|--|--|
| Appointment or Case | Appointment_or_Case | | | | | | | | | | | | | | | | | | | | |
| Description | | | | | | | | | | | | | | | | | | | | | |
| <p>Outcomes For each path the flow can take, create an outcome. For each outcome, specify the conditions that must be met for the flow to take that path.</p> <table border="1"> <thead> <tr> <th>OUTCOME ORDER</th> <th>OUTCOME DETAILS</th> <th>Delete Outcome</th> </tr> </thead> <tbody> <tr> <td>Appointment</td> <td>*Label Appointment</td> <td>*Outcome API Name Appointment</td> </tr> <tr> <td>Case</td> <td></td> <td></td> </tr> <tr> <td>Default Outcome</td> <td>Condition Requirements to Execute Outcome All Conditions Are Met (AND)</td> <td></td> </tr> <tr> <td></td> <td>Resource <input type="text" value="A_a How_may_I_Help_you X"/></td> <td>Operator <input type="text" value="Equals"/></td> <td>Value <input type="text" value="A_a Book_an_Appointment X"/></td> </tr> <tr> <td></td> <td colspan="3">+ Add Condition</td> </tr> </tbody> </table> | | OUTCOME ORDER | OUTCOME DETAILS | Delete Outcome | Appointment | *Label Appointment | *Outcome API Name Appointment | Case | | | Default Outcome | Condition Requirements to Execute Outcome All Conditions Are Met (AND) | | | Resource <input type="text" value="A_a How_may_I_Help_you X"/> | Operator <input type="text" value="Equals"/> | Value <input type="text" value="A_a Book_an_Appointment X"/> | | + Add Condition | | |
| OUTCOME ORDER | OUTCOME DETAILS | Delete Outcome | | | | | | | | | | | | | | | | | | | |
| Appointment | *Label Appointment | *Outcome API Name Appointment | | | | | | | | | | | | | | | | | | | |
| Case | | | | | | | | | | | | | | | | | | | | | |
| Default Outcome | Condition Requirements to Execute Outcome All Conditions Are Met (AND) | | | | | | | | | | | | | | | | | | | | |
| | Resource <input type="text" value="A_a How_may_I_Help_you X"/> | Operator <input type="text" value="Equals"/> | Value <input type="text" value="A_a Book_an_Appointment X"/> | | | | | | | | | | | | | | | | | | |
| | + Add Condition | | | | | | | | | | | | | | | | | | | | |

Subtask 4: Add Screen Element

1. Add a **Screen** element after the Decision element, on the Appointment path, and label it as “**Appointment Booking Screen**”.
2. Click on **Fields**, then **Record Variable Input**, and create a new Resource (AppointmentRecordRes) to display all the fields in the Appointment object.
3. Drag the necessary fields onto the screen to collect student information.
4. Click **Done**.

Edit Screen

Components Fields

Search components... 

Input (53)

- Action Button (Beta)
- Add Attendees
- Address
- Call Script
- Cancel Appointment
- Checkbox
- Checkbox Group
- Choice Lookup

Get more on the AppExchange 

EduConsultantPro Existing Student Flow

Appointment Date/Time

Date  Time 

Purpose/Topic

Notes

Screen Properties

*Label

*API Name 

Description 

> Configure Header

> Configure Footer

Subtask 5: Add GET Record Element

1. Add a **GET Record** element after the Decision element, under the Appointment path, and label it as “**Get Consultant Rec**”.
2. Configure the GET Record element:
 - **Select Object:** Consultant
 - **Condition Requirement:** All Conditions are Met (AND)
 - **Field:** Name
 - **Operator:** Equals
 - **Value:** {!AppointmentRecordRes.Consultant_Name_c}

Get Records

| | | |
|--------------------------------|--------------------|------------------------------------|
| * Label | * API Name | |
| Get Consultant Rec | Get_Consultant_Rec | |
| Description | | |
| Get Records of This Object | | |
| * Object | | |
| Consultant | | |
| Filter Consultant Records | | |
| Condition Requirements | | |
| All Conditions Are Met (AND) ▾ | | |
| Field | Operator | Value |
| Name | Equals | A_a AppointmentRecordRes > Ap... X |
| + Add Condition | | |

Subtask 6: Create Appointment Record Using Create Records Element

1. Add a **Create** element after the GET Consultant Rec element and label it as “**Create Appointment**”.
2. Configure the Create element:
 - **Select Object:** Appointment
 - **Field Values:**
 - **Appointment_DateTime_c:**
 {!AppointmentRecordRes.Appointment_DateTime_c}
 - **Consultant_c:** {!Get_Consultant_Rec.Id}
 - **Notes_c:** {!AppointmentRecordRes.Notes_c}
 - **PurposeTopic_c:** {!AppointmentRecordRes.PurposeTopic_c}
 - **Student_Name_c:** {!Get_Rec.Id}

 **Create Records** ×

| | |
|--|---|
| * Label | * API Name <small>i</small> |
| Create Appointment | Create_Appointment |
| Description | |
| | |
| * How to set record field values | |
| <input style="border: 1px solid #ccc; padding: 2px 10px; width: 150px; height: 20px; margin-bottom: 5px;" type="button" value="Manually"/> | |
| Create a Record of This Object | |
| * Object | |
| <input style="border: 1px solid #ccc; padding: 2px 10px; width: 150px; height: 20px; margin-bottom: 5px;" type="button" value="Appointment"/> | |
| Set Field Values for the Appointment | |
| Field | Value |
| <input style="border: 1px solid #ccc; padding: 2px 10px; width: 250px; height: 20px; margin-bottom: 5px;" type="button" value="Appointment_DateTime_c"/> |  AppointmentRecordRes > Appointment Date/... X Delete |
| Field | Value |
| <input style="border: 1px solid #ccc; padding: 2px 10px; width: 250px; height: 20px; margin-bottom: 5px;" type="button" value="Consultant_c"/> |  Consultant from Get_Consultant_Rec > Record ... X Delete |
| Field | Value |
| <input style="border: 1px solid #ccc; padding: 2px 10px; width: 250px; height: 20px; margin-bottom: 5px;" type="button" value="Notes_c"/> |  AppointmentRecordRes > Notes X Delete |

Subtask 7: Add Screen Element

1. Add a **Screen** element after the Create Appointment element, and label it as **“Confirmation Screen”**.
2. From the left side panel, search for the **Display Text** component and drag it to the main panel, label it as **“Appointment_Confirmation”**.
3. Paste the following text in the Resource picker box:
 Consultant Name : {!Get_Consultant_Rec.Name},
 Date & Time : {!AppointmentRecordRes.Appointment_DateTime_c},
 Notes : {!AppointmentRecordRes.Notes_c}
4. Click Done.

Edit Screen

Components Fields

Search components...

- ✓ Input (53)
- Action Button (Beta)
- Add Attendees
- Address
- Call Script
- Cancel Appointment
- Checkbox
- Checkbox Group
- Choice Lookup

Get more on the AppExchange [Get more](#)

EduConsultantPro Existing Student Flow

```
Consultant Name : {!Get_Consultant_Rec.Name},
Date & Time : {!AppointmentRecordRes.Appointment_DateTime_c},
Notes : {!AppointmentRecordRes.Notes_c}
```

Pause Previous **Finish**

Screen Properties

*Label

*API Name

Description

> Configure Header

> Configure Footer

Cancel **Done**

Edit Screen

Components Fields

Search components...

- ✓ Input (53)
- Action Button (Beta)
- Add Attendees
- Address
- Call Script
- Cancel Appointment
- Checkbox
- Checkbox Group
- Choice Lookup

Get more on the AppExchange [Get more](#)

EduConsultantPro Existing Student Flow

Display Text

```
Consultant Name : {!Get_Consultant_Rec.Name},
Date & Time : {!AppointmentRecordRes.Appointment_DateTime_c},
Notes : {!AppointmentRecordRes.Notes_c}
```

Pause Previous **Finish**

Display Text

*API Name

Insert a resource...

```
Consultant Name : {!Get_Consultant_Rec.Name},
Date & Time : {!AppointmentRecordRes.Appointment_DateTime_c},
Notes : {!AppointmentRecordRes.Notes_c}
```

Salesforce Sans

Subtask 8: Add a Subflow Element

1. Add a **Subflow** element after the Decision element, on the Case path, and search and select “Create a Case”, label it as “Create Student Case”.
2. Configure the Subflow element as needed.

 **Create a Case** X

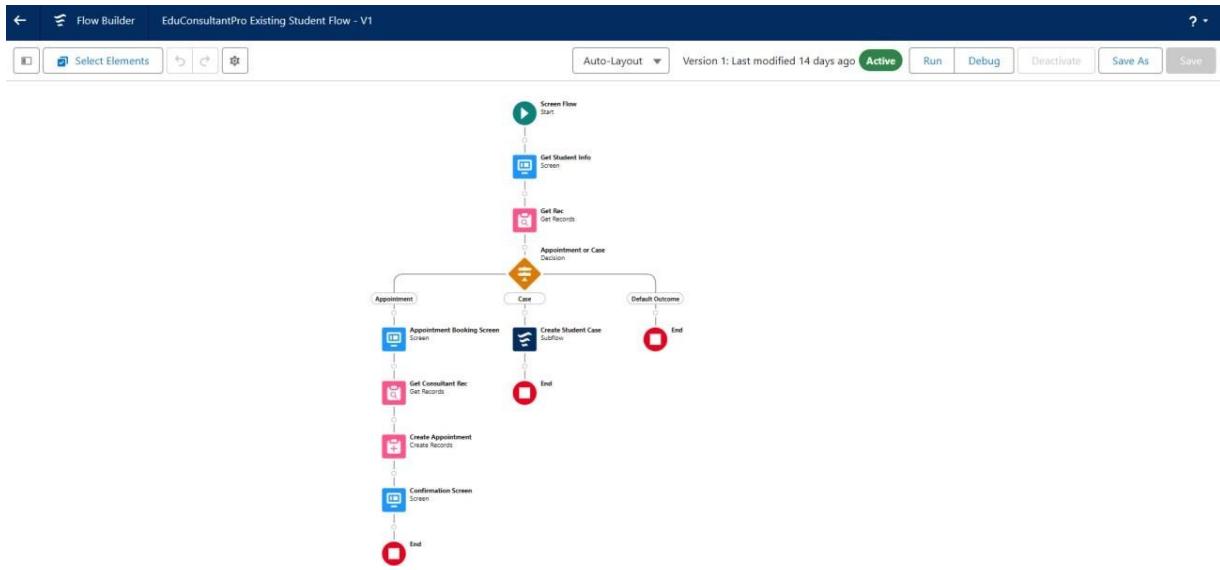
* Label

* API Name i

Description

Referenced Flow
 **Create a Case**
setup_service_experience__Create_Case ▼

3. Save the flow, label it as “EduConsultantPro Existing Student Flow”, and activate it.



Task 9: Create A ScreenFlow to Combine All The Flows At One Place

This task involves creating a central ScreenFlow that integrates various existing flows, allowing users to choose and access different functionalities from a single interface.

Subtask 1: Add Welcome Screen Element

1. From Setup, enter **Flow Builder** in the Quick Find box, select **New Flow → Screen Flow**.
2. Add a **Screen** element and label it as “**Welcome Screen**”.
3. From the left side panel, drag the **Display Text** component to the main panel.
4. Label the Display Text component as “**SuccessMessage**”.
5. Paste the following text into the Resource picker box:

“Welcome to EduConsultantPro

your premier destination for education and immigration solutions!

At EduConsultantPro, we understand that embarking on educational or immigration journeys can be both exhilarating and daunting. That's why we're here to guide you every step of the way with expertise, dedication, and personalized support.

Whether you're seeking to pursue your academic dreams abroad, navigate the complexities of immigration processes, or enhance your professional skills through international opportunities, EduConsultantPro is your trusted partner.

Our team of seasoned consultants is committed to understanding your unique aspirations and crafting tailored strategies to help you achieve your goals efficiently and effectively. From selecting the right educational institution to navigating visa procedures, our comprehensive services cover all aspects of your journey.

At EduConsultantPro, we believe in fostering inclusive communities and unlocking the full potential of every individual. With our unwavering commitment to excellence and integrity, we strive to make your experience with us seamless and rewarding.



Welcome to EduConsultantPro – where your aspirations meet our expertise, and together, we pave the path to success. Let's embark on this transformative journey together!"

6. Click **Done**.

Edit Screen

Components Fields

Search components...

- ▼ Input (53)
 - Action Button (Beta)
 - Add Attendees
 - Address
 - Call Script
 - Cancel Appointment
 - Checkbox
 - Checkbox Group
 - Choice Lookup

[Get more on the AppExchange](#)

Welcome to EduConsultantPro

your premier destination for education and immigration solutions!

At EduConsultantPro, we understand that embarking on educational or immigration journeys can be both exhilarating and daunting. That's why we're here to guide you every step of the way with expertise, dedication, and personalized support.

Whether you're seeking to pursue your academic dreams abroad, navigate the complexities of immigration processes, or enhance your professional skills through international opportunities, EduConsultantPro is your trusted partner. Our team of seasoned consultants is committed to understanding your unique aspirations and crafting tailored strategies to help you achieve your goals efficiently and effectively. From selecting the right educational institution to navigating visa procedures, our comprehensive services cover all aspects of your journey.

At EduConsultantPro, we believe in fostering inclusive communities and unlocking the full potential of every individual. With our unwavering commitment to excellence and integrity, we strive to make your experience with us seamless and rewarding.

Welcome to EduConsultantPro – where your aspirations meet our expertise, and together, we pave the path to success. Let's embark on this transformative journey together!

Screen Properties

*Label

*API Name

Description

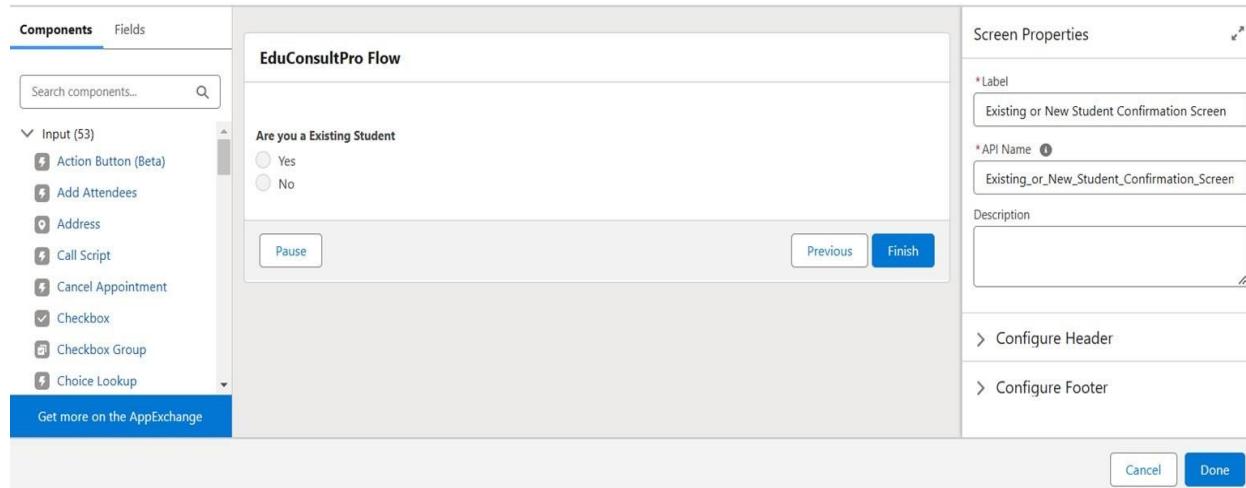
> Configure Header

> Configure Footer

Subtask 2: Add Existing or New Student Confirmation Screen

1. Add a **Screen** element after the Welcome Screen element and label it as “**Existing or New Student Confirmation Screen**”.
2. Add a **Radio Button** component from the left side panel.
 - **Label:** Are you an Existing Student?
 - **Choice 1:** “Yes”
 - **Choice 2:** “No”
3. Click **Done**.

Edit Screen



EduConsultPro Flow

Are you a Existing Student

Yes
 No

Pause Previous Finish

Screen Properties

*Label: Existing or New Student Confirmation Screen

*API Name: Existing_or_New_Student_Confirmation_Screen

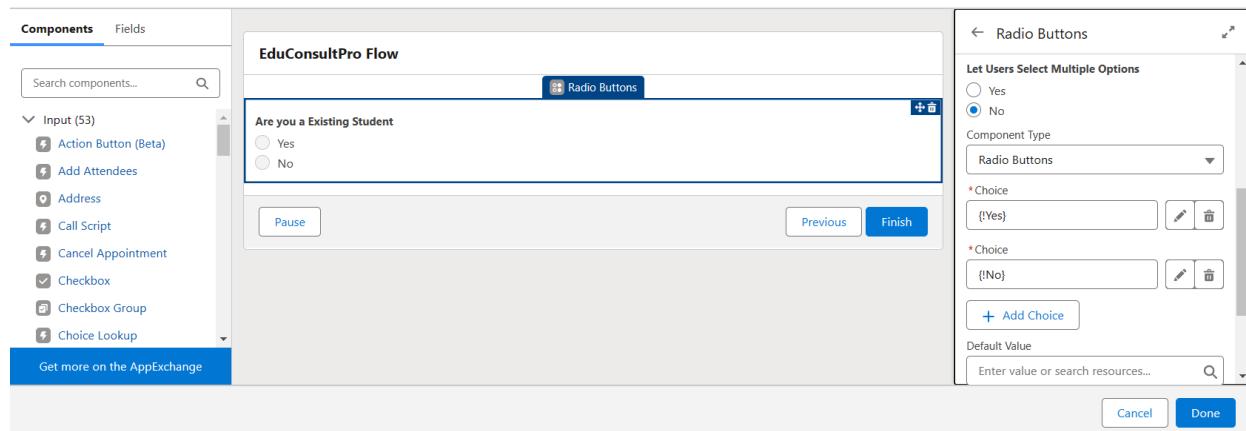
Description:

> Configure Header

> Configure Footer

Cancel Done

Edit Screen



EduConsultPro Flow

Are you a Existing Student

Yes
 No

Pause Previous Finish

← Radio Buttons

Let Users Select Multiple Options
 Yes
 No

Component Type: Radio Buttons

*Choice: {!Yes}

*Choice: {!No}

+ Add Choice

Default Value: Enter value or search resources...

Cancel Done

Subtask 3: Add Decision Element

1. Add a **Decision** element after the Existing or New Student Confirmation Screen element and label it as “**Decision 1**”.
2. Create an outcome:
 - **Label:** If Existing Student



- **Resource:** {!Are_you_a_Existing_Student}
- **Operator:** Equals
- **Value:** {!Yes}

3. Click the “+” icon to add more outcomes for “No” and other cases as required.

 **Decision**

| | |
|------------|-------------------------------|
| * Label | * API Name <small>(i)</small> |
| Decision 1 | Decision_1 |

Description

Outcomes For each path the flow can take, create an outcome. For each outcome, specify the conditions that must be met for the flow to take that path.

| | | | | | | | |
|--|--|---|----------|-------|--|-------------------------------------|---|
| OUTCOME ORDER <small>(i) +</small> <ul style="list-style-type: none"> <input type="checkbox"/> If Existing Student <input type="checkbox"/> If not a existing user <p>Default Outcome</p> | OUTCOME DETAILS <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <p>* Label <input type="text" value="If Existing Student"/></p> <p>* Outcome API Name <small>(i)</small> <input type="text" value="if_Existing_Student"/></p> <p>Condition Requirements to Execute Outcome <input type="button" value="All Conditions Are Met (AND)"/></p> <table border="0" style="margin-top: 10px;"> <tr> <td style="width: 30%;">Resource</td> <td style="width: 30%;">Operator</td> <td style="width: 40%;">Value</td> </tr> <tr> <td><input type="text" value="A_a ... > Are_you_a_Existing_Student X"/></td> <td><input type="text" value="Equals"/></td> <td><input type="text" value="A_a Yes X"/> <small>(i)</small></td> </tr> </table> <p><input type="button" value="+ Add Condition"/></p> </div> | Resource | Operator | Value | <input type="text" value="A_a ... > Are_you_a_Existing_Student X"/> | <input type="text" value="Equals"/> | <input type="text" value="A_a Yes X"/> <small>(i)</small> |
| Resource | Operator | Value | | | | | |
| <input type="text" value="A_a ... > Are_you_a_Existing_Student X"/> | <input type="text" value="Equals"/> | <input type="text" value="A_a Yes X"/> <small>(i)</small> | | | | | |

| | | | | | | | |
|--|--|--|----------|-------|--|-------------------------------------|--|
| OUTCOME ORDER <small>(i) +</small> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> If Existing Student <input checked="" type="checkbox"/> If not a existing user <p>Default Outcome</p> | OUTCOME DETAILS <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <p>* Label <input type="text" value="If not a existing user"/></p> <p>* Outcome API Name <small>(i)</small> <input type="text" value="if_Not_a_Existing_Student"/></p> <p>Condition Requirements to Execute Outcome <input type="button" value="All Conditions Are Met (AND)"/></p> <table border="0" style="margin-top: 10px;"> <tr> <td style="width: 30%;">Resource</td> <td style="width: 30%;">Operator</td> <td style="width: 40%;">Value</td> </tr> <tr> <td><input type="text" value="A_a ... > Are_you_a_Existing_Student X"/></td> <td><input type="text" value="Equals"/></td> <td><input type="text" value="A_a No X"/> <small>(i)</small></td> </tr> </table> <p><input type="button" value="+ Add Condition"/></p> </div> | Resource | Operator | Value | <input type="text" value="A_a ... > Are_you_a_Existing_Student X"/> | <input type="text" value="Equals"/> | <input type="text" value="A_a No X"/> <small>(i)</small> |
| Resource | Operator | Value | | | | | |
| <input type="text" value="A_a ... > Are_you_a_Existing_Student X"/> | <input type="text" value="Equals"/> | <input type="text" value="A_a No X"/> <small>(i)</small> | | | | | |

Subtask 4: Add Subflow for Existing Students

1. Add a **Subflow** element after the **Decision 1** element on the **If Existing Student** path.
2. Search for and select “**EduConsultantPro Existing Student Flow**”.
3. Label it as “**Existing Student Flow**”.
4. Click **Done**.

 EduConsultantPro Existing Student Flow X

*Label

*API Name i

Description

Referenced Flow

 **EduConsultantPro Existing Student Flow**
EduConsultantPro_Existing_Student_Flow ▼

Use values from the parent flow to set the inputs for the "EduConsultantPro Existing Student Flow" flow. By default, the parent flow stores all outputs. You can either reference outputs via the API name of the Subflow element or manually assign variables in the parent flow to store individual outputs from the "EduConsultantPro Existing Student Flow" flow.

Subtask 5: Add Subflow for New Students

1. Add a **Subflow** element after the **Decision 1** element on the **If Not an Existing Student**



path.

2. Search for and select “EduConsultantPro Student Flow”.
3. Label it as “New Student Flow”.
4. Click **Done**.



EduConsultPro Student Flow

* Label
New Student Flow

* API Name i
New_Student_Flow

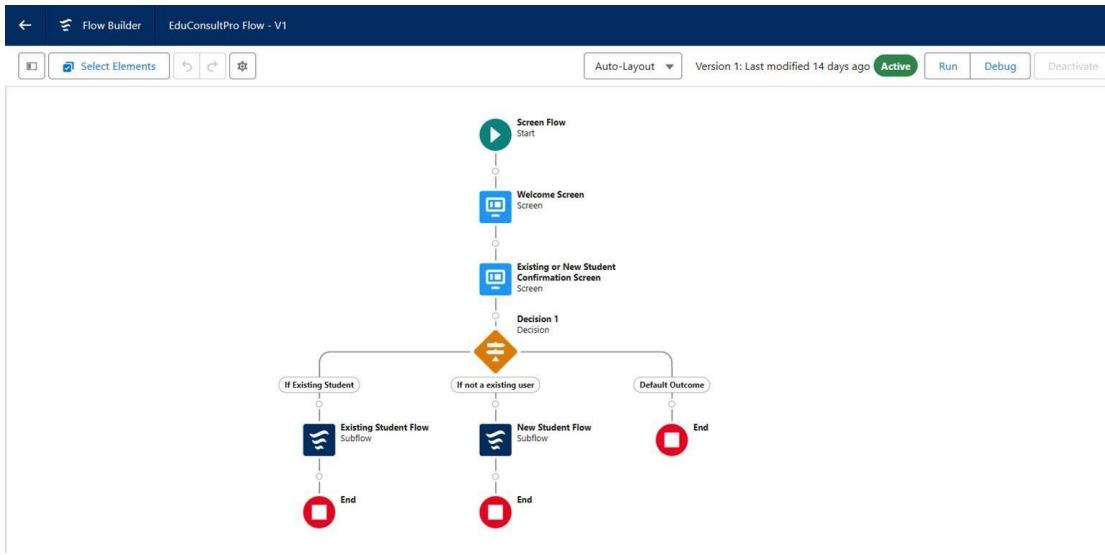
Description

Referenced Flow

EduConsultPro Student Flow
EduConsultPro_Student_Flow

Use values from the parent flow to set the inputs for the "EduConsultPro Student Flow" flow. By default, the parent flow stores all outputs. You can either reference outputs via the API name of the Subflow element or manually assign variables in the parent flow to store individual outputs from the "EduConsultPro Student Flow" flow.

5. Save the flow, label it as “**EduConsultantPro Flow**”, and activate it.



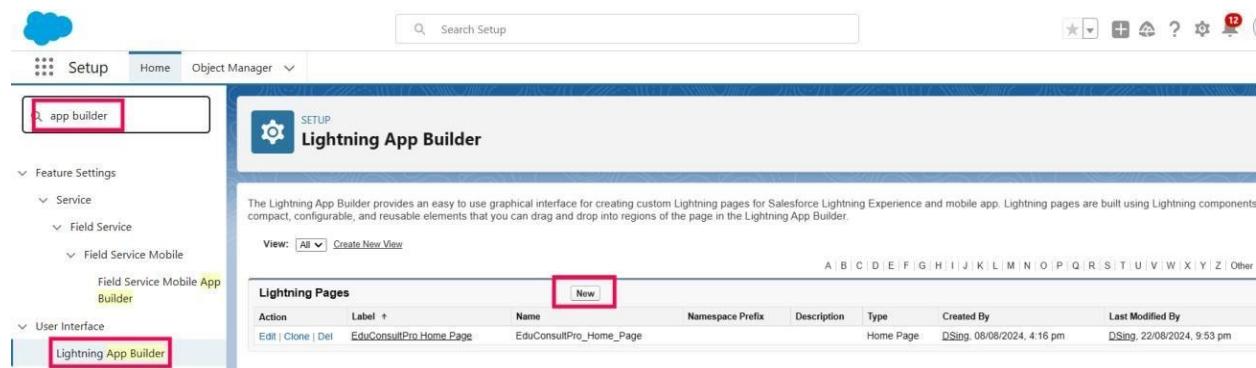
Task 10: Create A Lightning App Page

This task involves creating a Lightning App Page and making it available in the application. The new page will feature the "EduConsultantPro Flow" component.

Subtask 1: Create A Lightning App Page

1. Access Lightning App Builder:

- From Setup, enter **App Builder** in the Quick Find box.
- Click **Lightning App Builder**.



2. Create a New Home Page:

- Click **New**.
- Select **Home Page** and click **Next**.

Create a new Lightning page

App Page

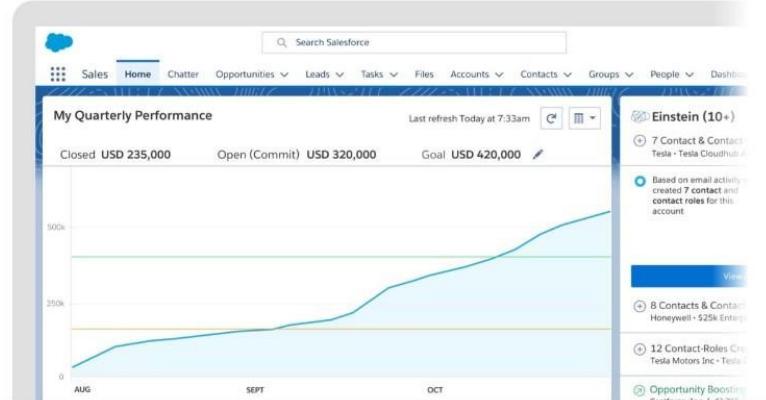
[Home Page](#)

Record Page

Embedded Service Page

Voice Extension

Customize the Lightning Experience Home page.



Next

3. Configure the Home Page:

- **Name the Page:** Enter “EduConsultPro Home Page”.
- **Select a Template:** Choose the **Standard Home Page** template.
- **Click Done.**

Create a new Lightning page

CHOOSE PAGE TEMPLATE

CLONE SALESFORCE DEFAULT PAGE

STANDARD (7)

Header and Three Regions



Home Template One Region



Home Template One Region



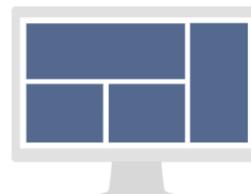
Home Template One Region



Home page header two columns left side bar

[Standard Home Page](#)

CUSTOM (0)



Lightning Experience Home page template with a header above two equal-width regions, and a full-height sidebar.

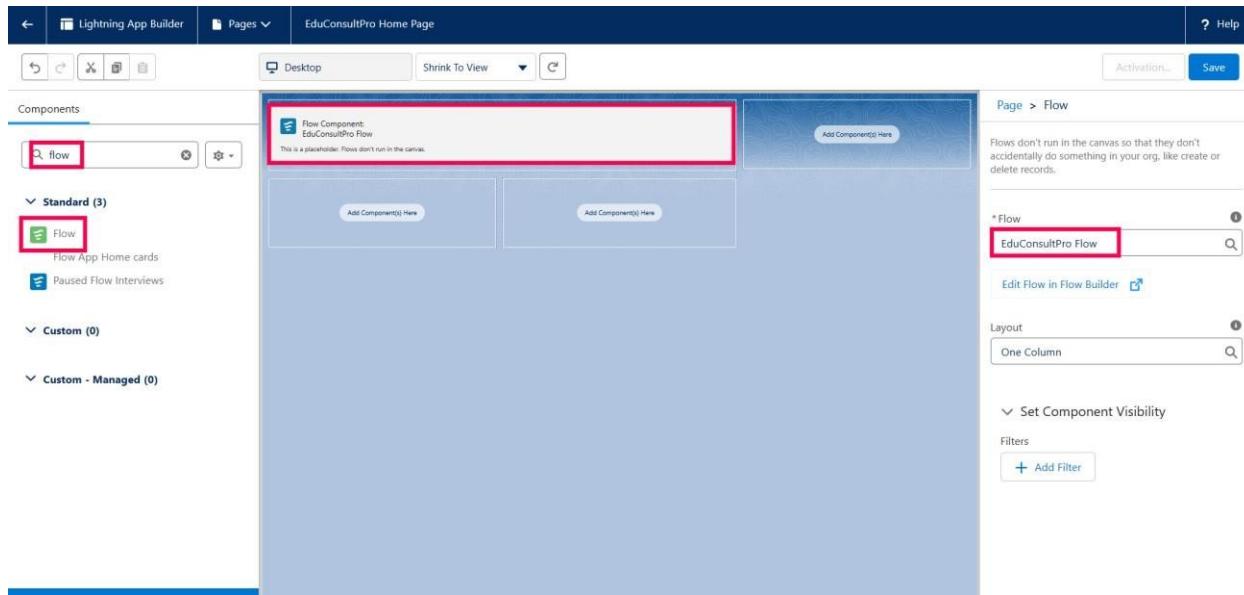
Supported form factors: desktop.

Back

Done

4. Add the Flow Component:

- Drag the **Flow** component to the **top-right** region of the page layout.
- In the component properties pane, search for and select “**EduConsultantPro Flow**”.



5. Save and Activate the Page:

- Click **Save**.
- Click **Activate**.

6. Assign the Page to Apps and Profiles:

- Click **App and Profile**.
- Click **Assign to Apps and Profiles**.
- **Select the Sales App:** Choose **Sales** and click **Next**.
- **Assign to Profiles:** Scroll down and select **System Administrator** profile. Click **Next**.
- **Review and Save:** Review the assignment details and click **Save**.



Select Apps

First, select the Lightning apps to display "EduConsultPro Home Page" as the home page. You'll select the related profiles next.

| Lightning Apps (12) | | 1 Selected |
|---|---|------------|
| App Name | Description | |
| <input checked="" type="checkbox"/> Sales | Manage your sales process with accounts, leads, opportunities, and m... | |
| <input type="checkbox"/> Sales Console | (Lightning Experience) Lets sales reps work with multiple records on o... | |

Select Profiles

Select the profiles to display "EduConsultPro Home Page" as the home page.

| Profiles (41) | | 1 Selected |
|--|-------------|------------|
| Profile | Description | |
| <input type="checkbox"/> Silver Partner User | | |
| <input type="checkbox"/> Solution Manager | | |
| <input type="checkbox"/> Standard Platform User | | |
| <input type="checkbox"/> Standard User | | |
| <input checked="" type="checkbox"/> System Administrator | | |

Activation: EduConsultPro Home Page

- The org default** home page is displayed unless more specific assignments are made.
- The app default** home page is displayed for specified apps, and overrides the org default.
- Any app and profile** assignments are displayed for specified app and profile combinations, and they override all other assignments.

[Learn more about forecast page assignment in Salesforce Help.](#)

Org Default App Default App and Profile

Set the home page for different user profiles when they're using certain apps. These assignments are the most specific, and override all other home page assignments.

| Assignments (1) | | Add Assignments | Remove Assignments |
|-----------------|----------------------|-----------------|--------------------|
| App | Profile | | |
| Sales | System Administrator | | |

[Close](#)

