Student Enquiry CRM

## Phase 1 – Problem Understanding & Industry Analysis

## 1. Requirement Gathering

**Description:**  
Many educational institutions receive student enquiries from multiple channels such as website forms, referrals, and events. Tracking these manually can lead to missed follow-ups and lost potential students.

**The Student Enquiry CRM aims to:**

* Record all student enquiries in a structured format.
* Automate follow-up reminders for counselors.
* Track enquiry sources to evaluate marketing effectiveness.
* Improve conversion rates from enquiries to enrolled students.

## 2. Stakeholder Analysis

**Description:**  
Identifying stakeholders helps understand who will interact with the CRM and their requirements.

**Table of Stakeholders:**

| Stakeholder | Role | Requirement from CRM |
| --- | --- | --- |
| Counselor | Manage enquiries | View new enquiries, receive follow-up reminders, update status |
| Admin | Configure CRM | Create users, manage fields, automate flows |
| Student | Submit enquiry | Receive timely follow-up from counselors |

## 3. Business Process Mapping

**Description:**  
The business process shows the lifecycle of a student enquiry:

1. Student submits enquiry (via Website, Referral, Event).
2. CRM records the enquiry in the Student Enquiry object.
3. Follow-Up Task is automatically created for counselors.
4. Counselor completes follow-up.
5. Status updated to Converted or Lost.

## 4. Industry-specific Use Case Analysis

**Description:**  
Education CRMs commonly use automated reminders and source tracking to improve student engagement and conversion.  
Key benefits include:

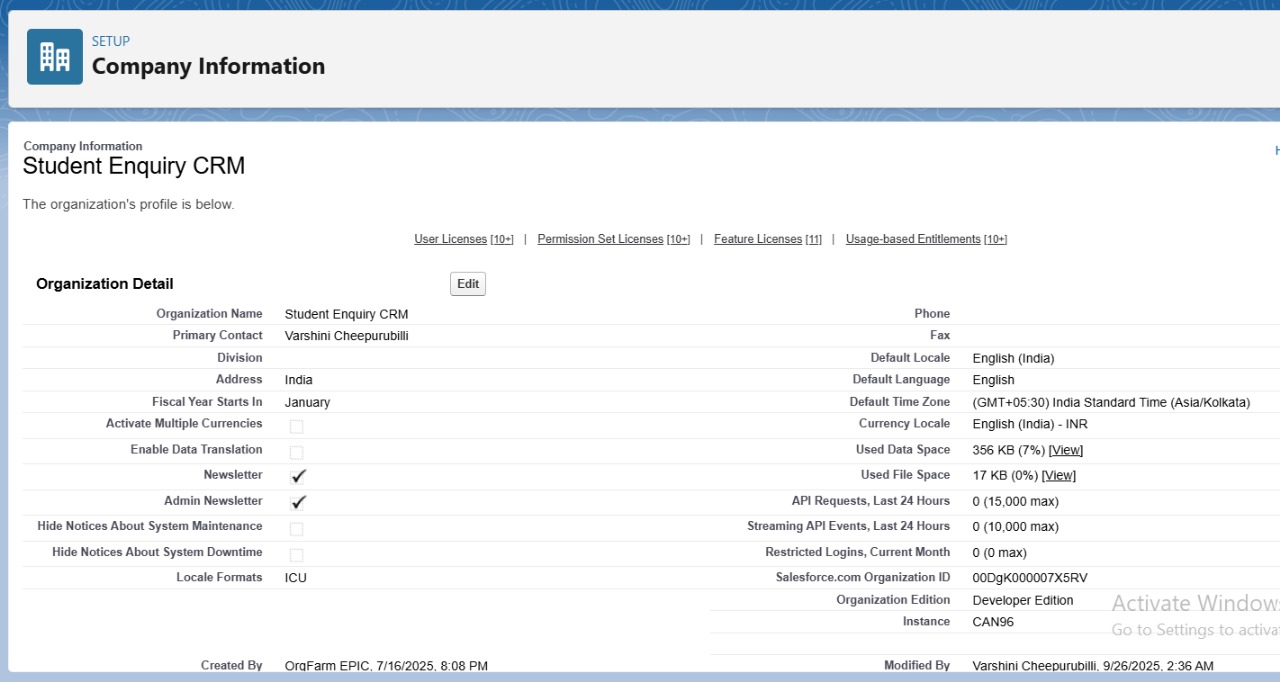
* Ensures no enquiry is missed.
* Provides data for evaluating marketing channels.
* Improves counselor productivity.
* Generates reports for management to track performance.

## 5. AppExchange Exploration

**Description:**  
Salesforce AppExchange offers applications for lead and student enquiry management. Exploring these apps helped define features for our project, such as:

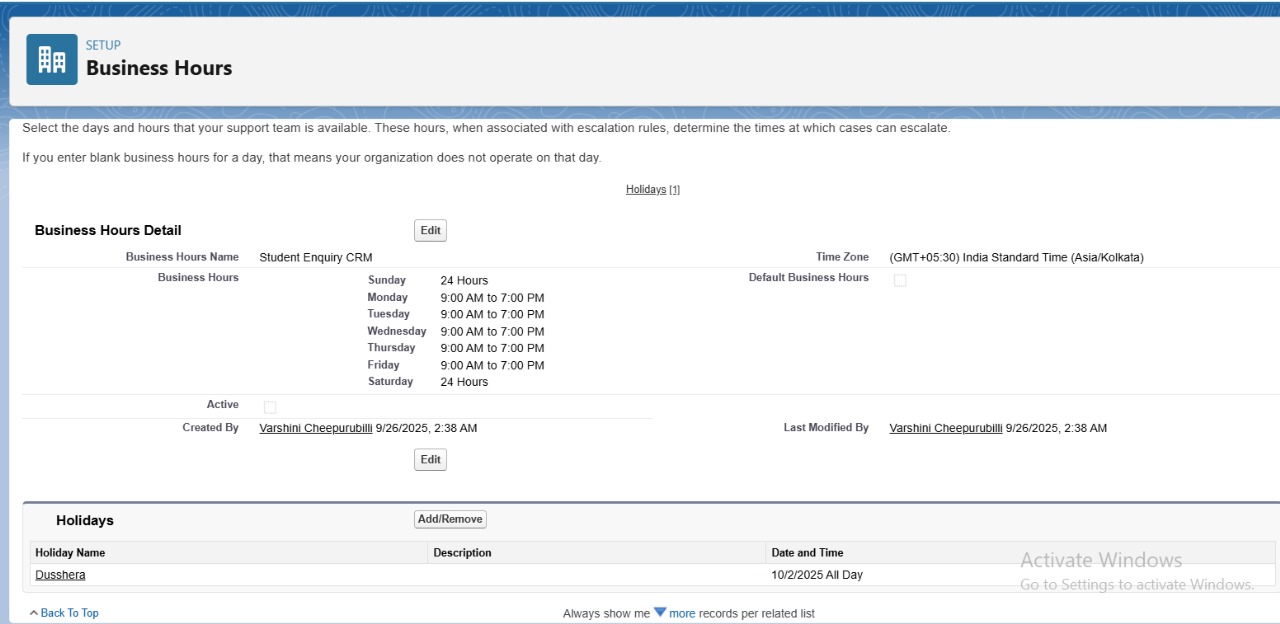
* Automatic follow-up reminders.
* Source tracking.
* Reporting on enquiry conversions.

**Phase 2 – Org Setup & Configuration**  
  
1. Salesforce Edition  
  
Description:  
The Salesforce edition determines the available features, number of users, and storage. For this project, a Developer Edition or Trailhead Playground is sufficient.

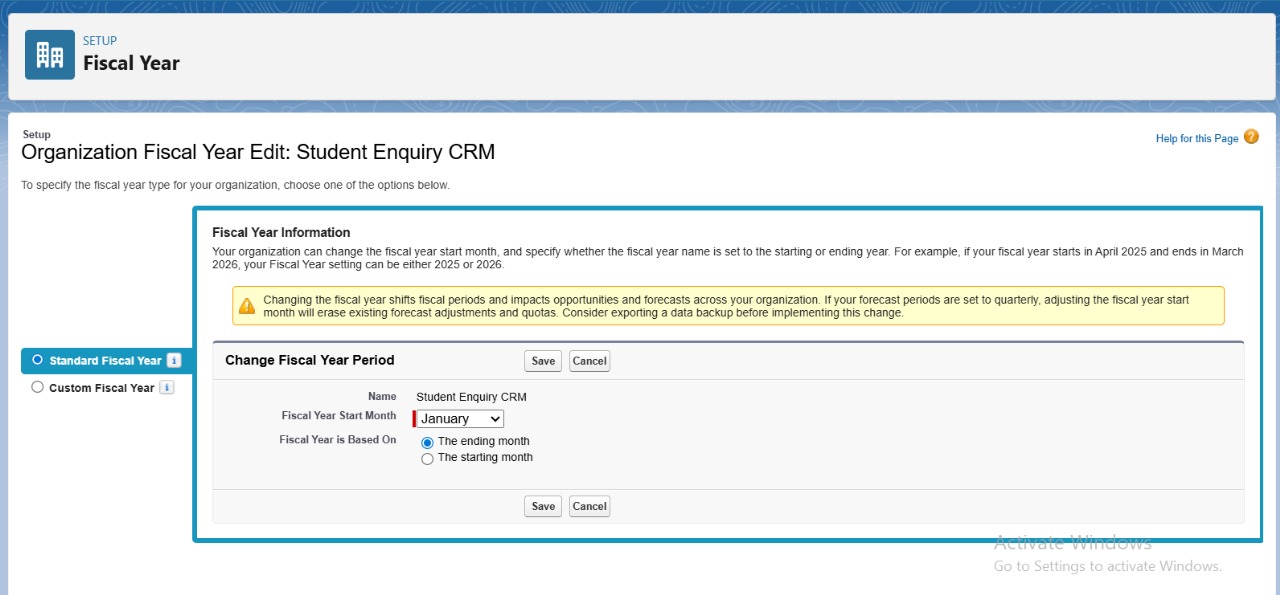


2. Company Profile Setup

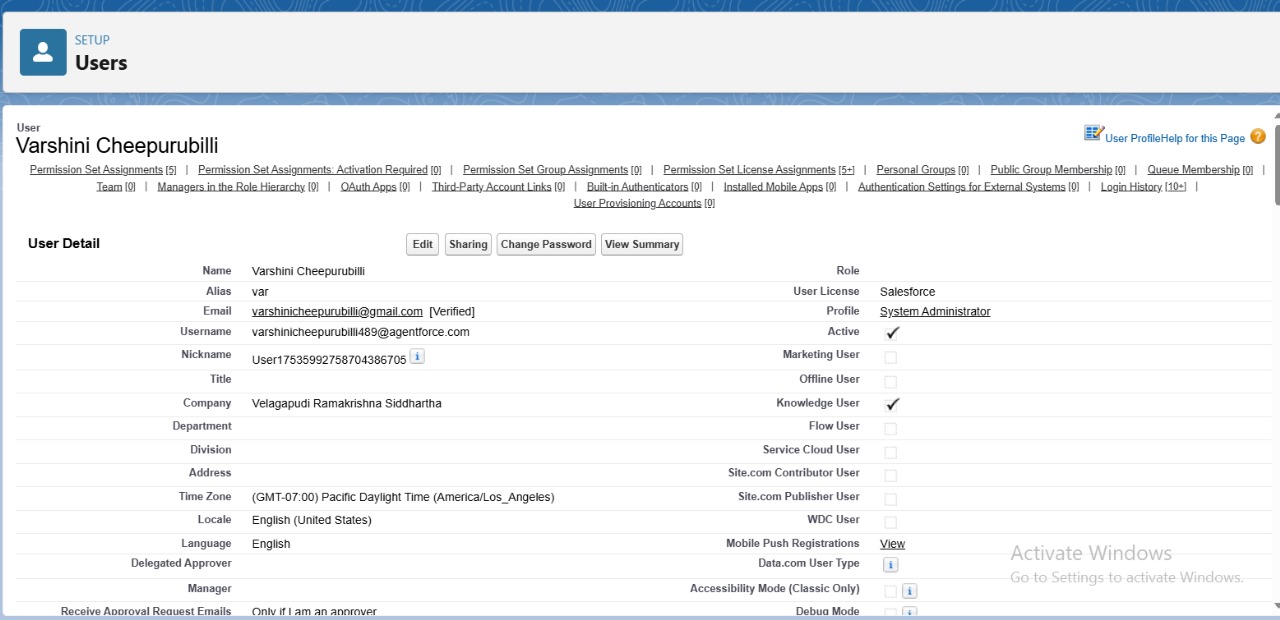
Description:  
Company Profile stores organization information such as name, address, default currency, and time zone. Accurate company info is important for reporting, scheduling, and email communications.  
  
Implementation:  
  
Setup → Company Information → Edit → Fill in Name, Address, Default Currency, Time Zone.  
  
  
3. Business Hours & Holidays  
  
Use Case / Description:  
Defines working hours and holidays used in automation and task/case management.  
  
Implementation:  
  
Setup → Business Hours → New → Define hours (e.g., 9 AM – 6 PM, Monday to Friday)  
  
Setup → Holidays → New → Define public holidays



4. Fiscal Year Settings  
  
Use Case / Description:  
Defines the organization’s fiscal period, used in reporting and forecasting student conversions.  
  
Implementation:  
  
Setup → Fiscal Year → Use Standard or Custom Fiscal Year → Save

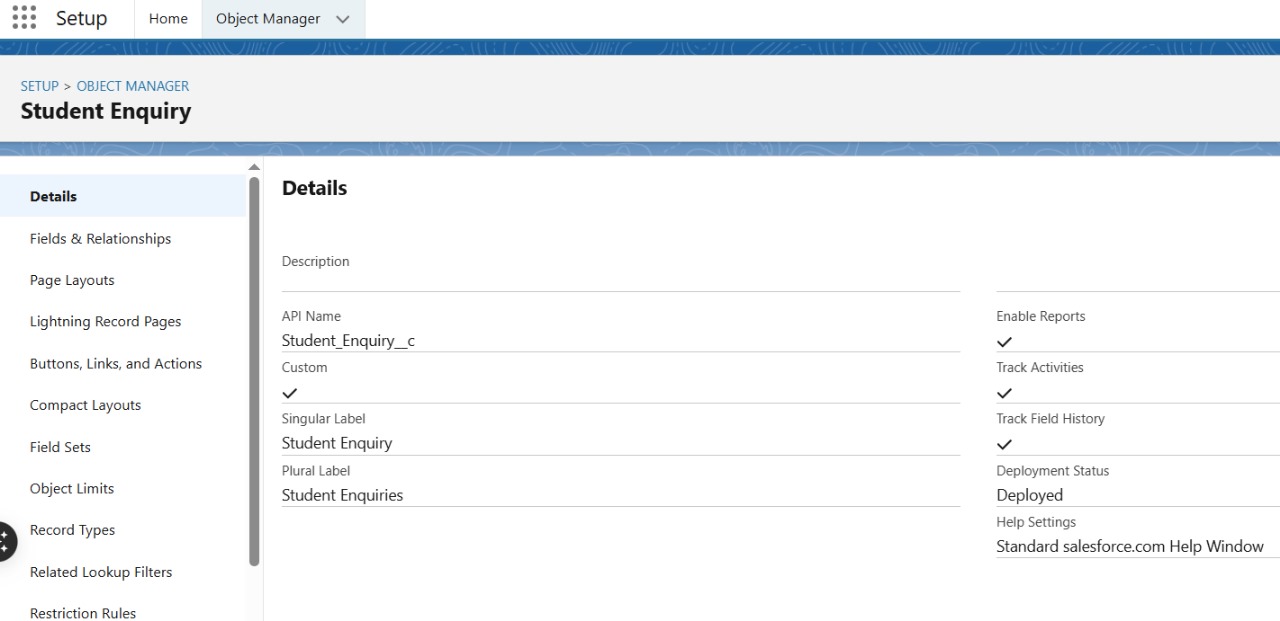
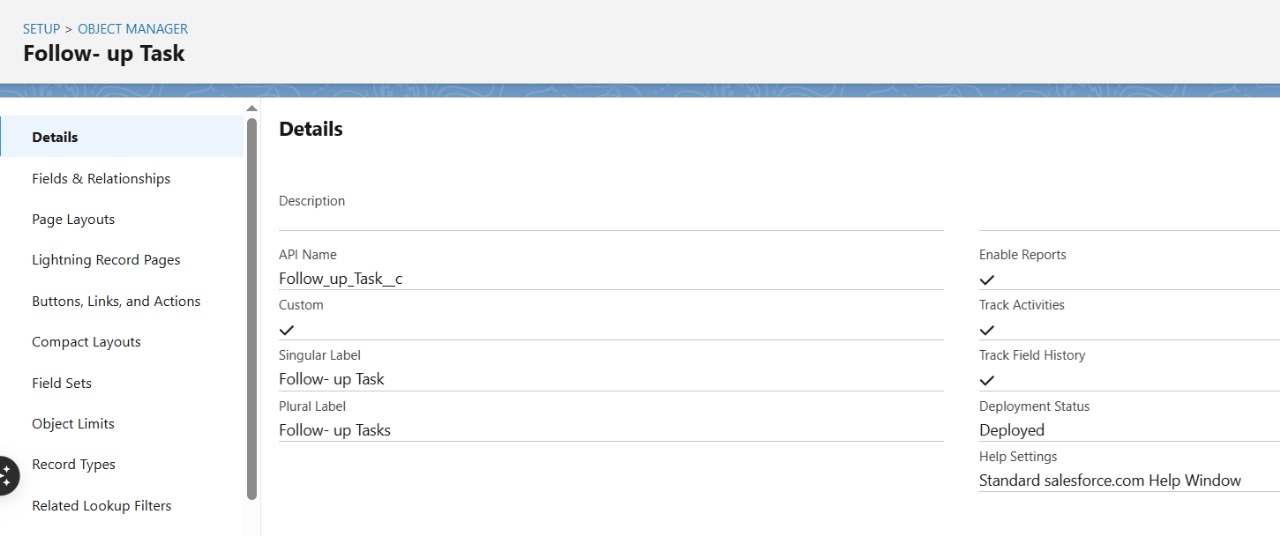


5. User Setup & Licenses  
  
Description:  
Users are individuals who can access Salesforce. Licenses determine access level. Roles and profiles control permissions.  
  
Implementation:  
  
Setup → Users → New User → Fill Name, Email, Role, Profile, License

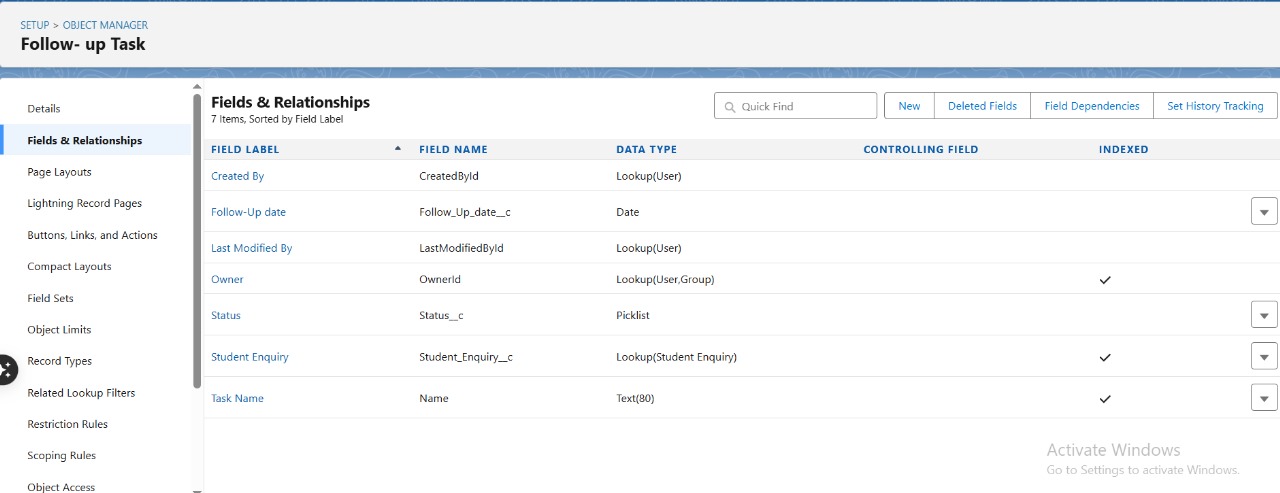
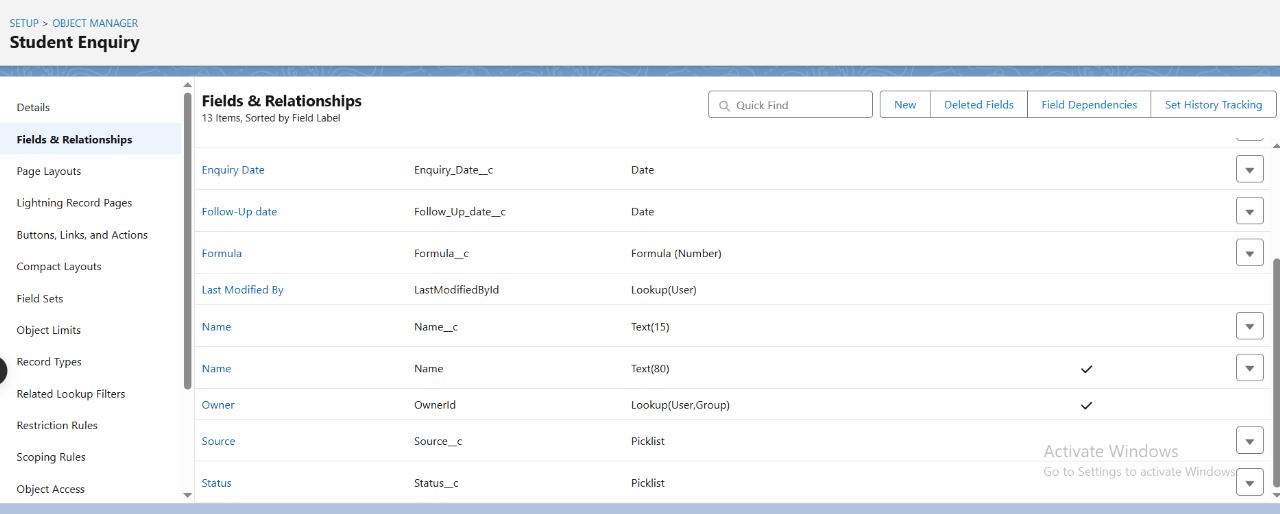


6. Login Access Policies  
  
 Description:  
Defines login restrictions and security policies to ensure only authorized users access Salesforce.  
  
Implementation:  
Setup → Security → Login Access Policies → Enable/Configure  
  
7. Developer Org Setup  
  
Description:  
Developer Org or Trailhead Playground is used to build and test the CRM project without affecting production.  
  
Implementation:  
  
Use Trailhead Playground → Connect to Salesforce → Create your objects, fields, and flows  
  
  
  
8. Sandbox Usage   
  
Description:  
Sandboxes allow testing new features safely without impacting live data. In a beginner project, the Developer Org acts as a sandbox.  
  
  
9. Deployment Basics – Outbound Change Set  
  
Description:  
Outbound Change Sets allow transferring components from one org to another ( Dev Org → Production).

**Phase 3 – Data Modeling & Relationships**  
  
  
1. Standard & Custom Objects  
  
 Description:  
  
Standard objects like Contacts and Accounts can be used for student and institution info.  
  
Custom objects track project-specific data, e.g., Student Enquiry and Follow-Up Task.  
  
Custom objects store relevant fields such as student name, contact info, course interest, and follow-up date.  
  
  
Implementation:  
  
Setup → Object Manager → Create → Custom Object → Fill Label, Plural Label, Record Name → Save

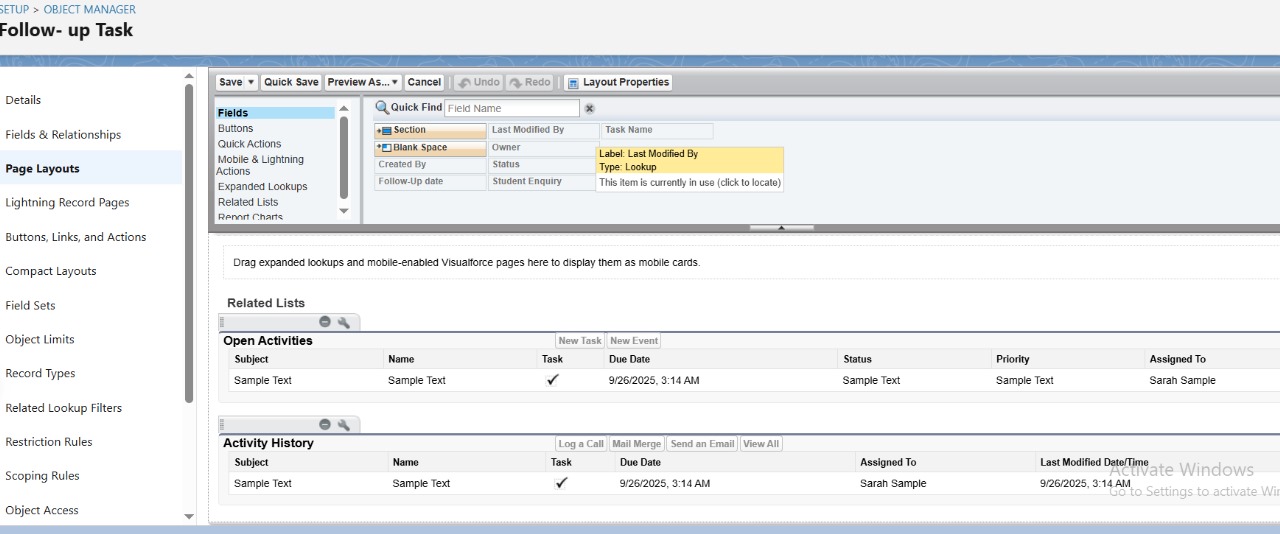
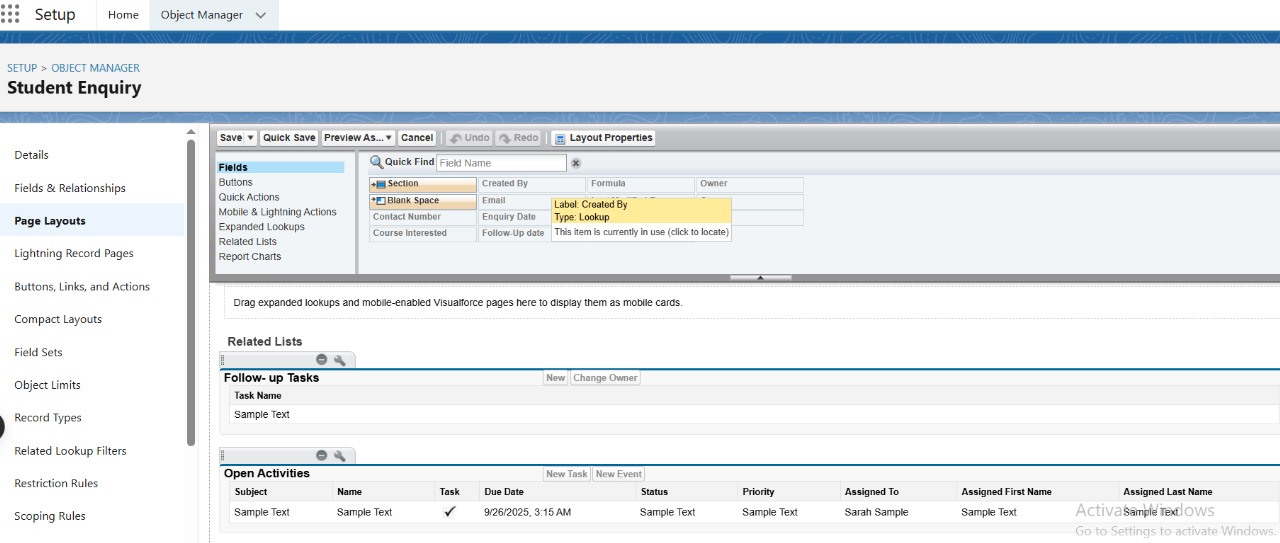


2. Fields  
  
Description:  
Fields store information for each record.  
  
Example for Student Enquiry: Name, Email, Phone, Status, Source, Course Interested, Follow-Up Date  
  
Example for Follow-Up Task: Related Enquiry (Lookup), Follow-Up Date, Status, Notes  
  
  
Implementation:  
  
Setup → Object Manager → Object → Fields & Relationships → New → Choose field type → Save



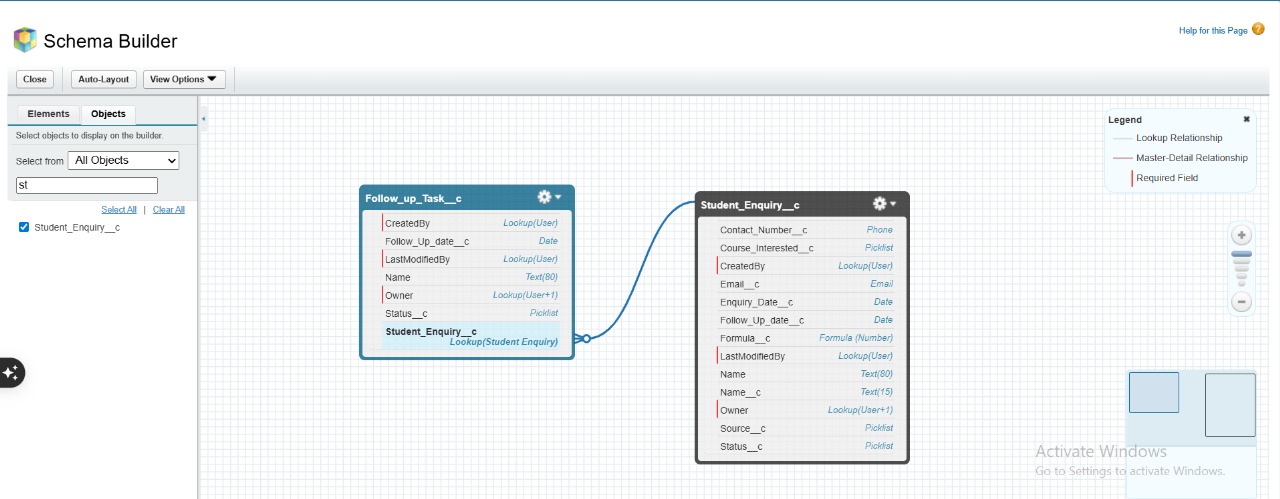
3. Record Types

Description:  
Record Types allow different business processes or layouts for the same object.  
  
Example: Enquiry Type could have Online vs Offline forms with different page layouts.  
  
  
Implementation:  
  
Setup → Object Manager → Object → Record Types → New → Name → Assign Page Layout → Save  
  
  
4. Page Layouts & Compact Layouts  
  
Description:  
  
Page Layouts: Control which fields, related lists, and buttons appear on record pages.  
  
Compact Layouts: Control which key fields appear in record highlights and mobile view.  
  
  
Implementation:  
  
Setup → Object Manager → Object → Page Layouts / Compact Layouts → New / Edit → Drag & Drop fields → Save



5. Schema Builder

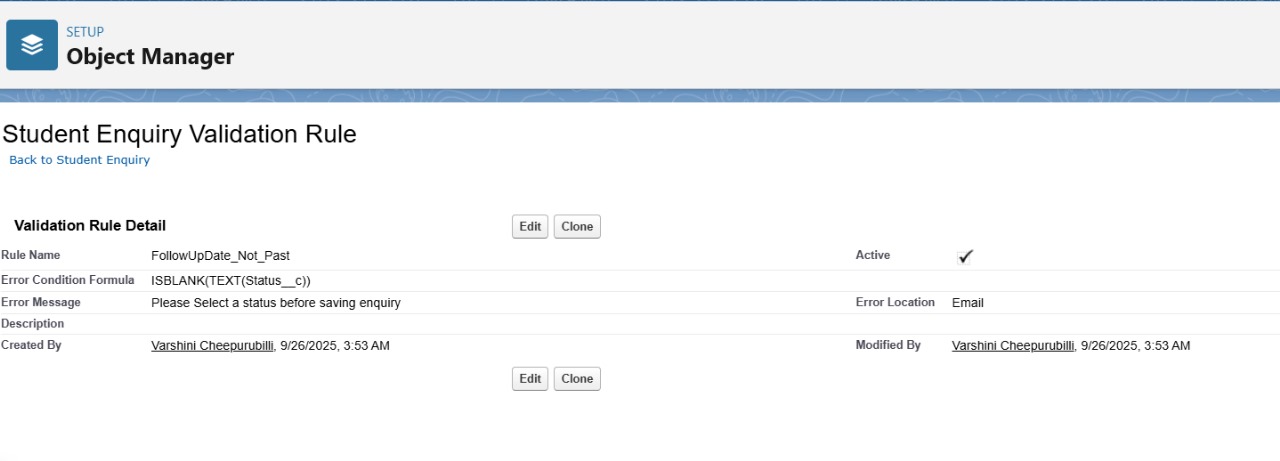
Description:  
Schema Builder visually displays all objects, fields, and relationships in the org.  
  
Implementation:  
  
Setup → Schema Builder → Select objects → View relationships



6. Lookup vs Master-Detail vs Hierarchical Relationships

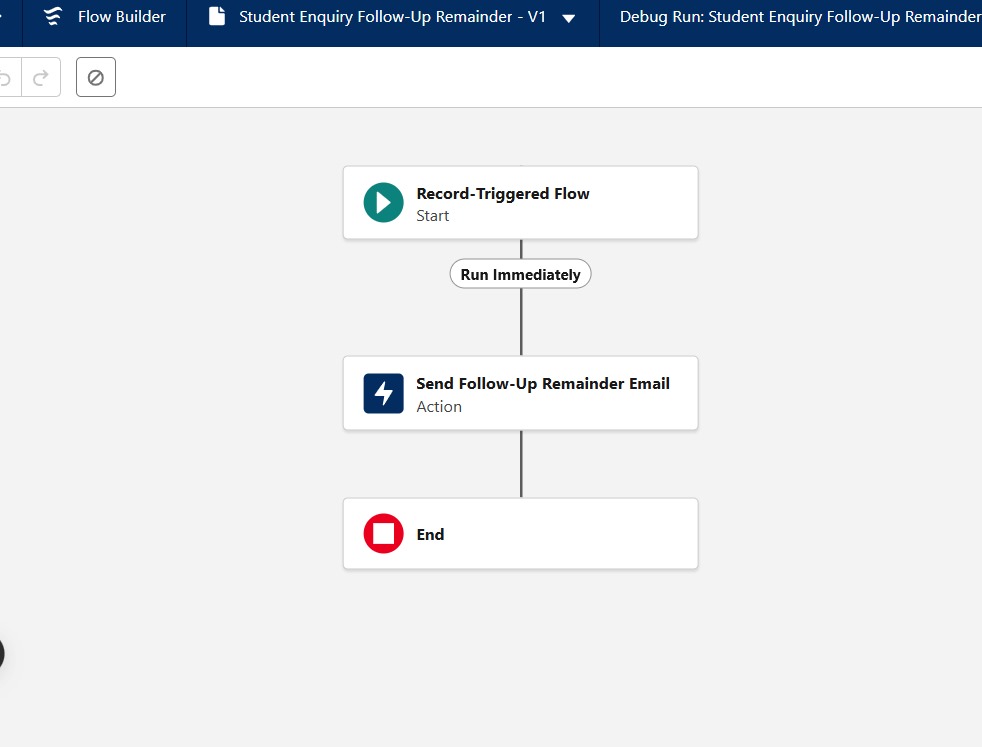
Description: **Lookup**: Relates two objects loosely ( Follow-Up Task → Student Enquiry) **Master-Detail**: Strong relationship; detail inherits security & ownership of master **Hierarchical**: Used for user object ( manager hierarchy)  
  
  
Implementation:  
  
Setup → Object Manager → Object → Fields & Relationships → New → Choose Relationship Type → Save

**Phase 4 – Process Automation (Admin)**  
  
1. Validation Rules  
  
Description:  
Validation rules ensure data integrity by preventing users from entering invalid data.  
  
Example: Prevent Follow-Up Date from being set in the past.  
  
  
Implementation:  
  
Setup → Object Manager → Student Enquiry → Validation Rules → New → Formula → Save

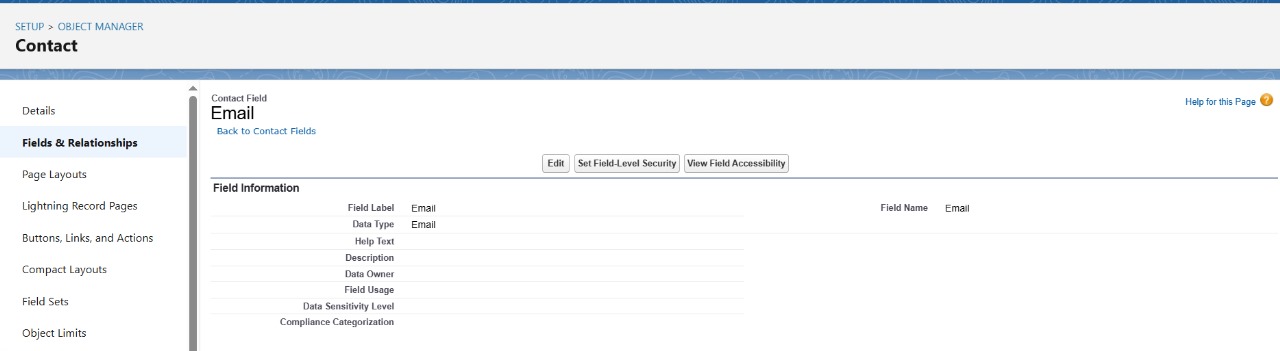


2. Workflow Rules   
  
Description:  
Workflow rules automate simple actions when criteria are met.  
  
Example: Send email to counselor when Status = “New”  
  
  
Implementation:  
  
Setup → Workflow Rules → New Rule → Select Object → Define Criteria → Add Workflow Action → Save  
  
  
3. Process Builder  
  
Use Case / Description:  
Process Builder automates multi-step processes like record updates and email alerts.  
  
Example: When Status = “Converted”, create a Student record automatically.  
  
  
Implementation:  
  
Setup → Process Builder → New → Select Object → Define Criteria → Add Action → Save  
  
  
4. Flow Builder  
  
A. Record-Triggered Flow – Follow-Up Task Creation

Description:  
Automatically create a follow-up task when a Student Enquiry is created or updated.  
  
Implementation Steps:  
  
1. Setup → Flows → New Flow → Record-Triggered Flow  
  
  
2. Object: Student Enquiry  
  
  
3. Trigger: When record is created or updated  
  
  
4. Condition: Follow-Up Date is not blank  
  
  
5. Action: Create Follow-Up Task → Set fields (Related Enquiry, Due Date, Status)  
  
  
6. Save → Activate



B. Email Alert for Follow-Up  
  
Description:  
Send an automated email to counselor when Follow-Up Date = TODAY.  
  
Implementation Steps:  
  
1. Setup → Email Alerts → New → Select Flow or Workflow  
  
  
2. Action: Send email → Select template and recipient (Counselor)  
  
  
3. Save → Activate



5. Tasks & Custom Notifications  
  
Description:  
  
Tasks track actionable items like follow-ups.  
  
Custom notifications alert users in Salesforce when a follow-up is due.  
  
  
Implementation Steps:  
  
Setup → Object Manager → Follow-Up Task → New Field / Layouts  
  
Setup → Notification Builder → New Custom Notification → Assign to Profile

**Phase 5 – Apex Programming (Developer)**  
  
1. Classes & Objects  
  
Description:  
Apex classes allow you to write reusable logic that can be called from triggers, Lightning components, or Flows.  
  
Example: StudentEnquiryHandler class to manage follow-ups and conversions.  
  
  
Implementation:  
  
Setup → Apex Classes → New → Write class → Save  
  
  
  
2. Apex Triggers (before/after insert/update/delete)  
  
Description:  
Triggers automatically perform actions when records are created, updated, or deleted.  
  
Example: When a Student Enquiry’s Status = “Converted”, automatically create a Student record.  
  
  
Implementation:  
  
Setup → Object Manager → Student Enquiry → Triggers → New → Write trigger → Save  
  
  
Example Trigger (Before Update):

3. Trigger Design Pattern  
  
 Description:  
Using a handler class pattern separates logic from trigger to improve maintainability.  
  
Implementation:  
  
Trigger calls a class method in StudentEnquiryHandler instead of containing logic directly.  
  
  
  
4. SOQL & SOSL  
  
Description:  
  
SOQL: Query Salesforce records.  
  
SOSL: Search text across multiple objects.  
  
Example: Retrieve all enquiries with Status = “New”.  
  
  
Implementation:  
  
List<Student\_Enquiry\_c> newEnquiries = [SELECT Name, Emailc FROM Student\_Enquiryc WHERE Status\_c = 'New'];  
  
  
5. Collections: List, Set, Map  
  
Description:  
Collections store multiple records in memory.  
  
Example: List for batch operations, Map for lookup by ID.  
  
  
Implementation:  
  
Map<Id, Student\_Enquiry\_c> enquiryMap = new Map<Id, Student\_Enquiryc>([SELECT Id, Statusc FROM Student\_Enquiry\_c]);  
  
  
6. Control Statements  
  
Description:  
Used for conditional logic and loops.  
  
Example: Loop through enquiries to update Status.  
  
  
Implementation:  
  
for(Student\_Enquiry\_\_c e : newEnquiries){  
 if(e.Status\_\_c == 'New'){  
 e.Status\_\_c = 'Contacted';  
 }  
}  
update newEnquiries;  
  
  
  
  
7. Exception Handling  
  
Description:  
Catches and handles runtime errors to prevent process failures.  
  
Implementation:  
  
try {  
 insert newStudent;  
} catch (DmlException e) {  
 System.debug('Error creating student: ' + e.getMessage());  
}  
  
  
  
10. Test Classes  
  
Description:  
Salesforce requires at least 75% code coverage for deploying Apex to production.  
  
Example: Test creation of Student records when an enquiry is converted.  
  
  
Implementation:  
  
@isTest  
public class TestConvertEnquiryTrigger {  
 @isTest static void testConvertEnquiry() {  
 Student\_Enquiry\_c enquiry = new Student\_Enquiry\_c(  
 Name='Test Student',  
 Status\_\_c='Converted',  
 Email\_\_c='test@student.com'  
 );  
 insert enquiry;  
   
 Student\_Enquiry\_c insertedEnquiry = [SELECT Id, Student\_Createdc FROM Student\_Enquiry\_c WHERE Id = :enquiry.Id];  
 System.assert(insertedEnquiry.Student\_Created\_\_c == true);  
 }  
}

**Phase 6: User Interface Development (UI)**  
  
**Lightning App Builder / Record Pages:** Customized record pages for Student Enquiry and Student objects.  
  
**Tabs & Home Page Layouts:** Created separate tabs for Enquiries, Students, and Reports; home page layout shows follow-ups and tasks.  
  
**Utility Bar:** Added shortcuts for quick access to tasks, notifications, and reports.  
  
**Lightning Web Components (LWC):** Optional dashboard to display pending follow-ups or recent enquiries.  
  
**Apex Integration with LWC:** Allows dynamic data display and actions like creating a student record from LWC.  
  
  
  
**Phase 7: Integration & External Access**  
  
**Named Credentials / Remote Site Settings:** Configured to allow secure API calls to external services if needed.  
  
**External Services & Web Services (REST/SOAP):** Enables integration with other applications like email systems or ERP.  
  
P**latform Events / Change Data Capture:** Used for real-time updates and notifications if data changes occur externally.  
  
  
**Phase 8: Data Management & Deployment**  
  
**Data Import Wizard / Data Loader:** Imported sample student enquiries for testing.  
  
**Duplicate Rules:** Prevented duplicate student or enquiry records.  
  
**Data Export & Backup:** Periodic backup of all records for safety.  
  
**Change Sets / VS Code / SFDX:** Deployed objects, flows, and triggers from sandbox to production safely.  
  
  
  
**Phase 9: Reporting, Dashboards & Security Review**  
  
**Reports:** Created tabular and summary reports for enquiries, follow-ups, and student conversions.  
  
**Dashboards:** Visual representation of enquiry status, pending follow-ups, and conversion rate.  
  
**Profiles, Roles & Permission Sets:** Defined user access for counselors, admins, and managers.  
  
S**haring Rules / OWD / Field-Level Security:** Ensured correct visibility and security of sensitive student data.  
  
**Phase 10: Quality Assurance Testing**  
  
**Test Cases:** Created test cases for all major functionalities including:  
  
-Student Enquiry creation and validation rules  
  
-Follow-Up task automation  
  
-Email alerts for pending follow-ups  
  
-Conversion trigger from enquiry to student record  
  
  
  
Sample Test Table:

| **Use Case** | **Test Steps** | **Expected Result** | **Actual Result** |  |
| --- | --- | --- | --- | --- |
| Convert Enquiry | Update status to “Converted” | Student record created, checkbox marked | Passed |  |
| Follow-Up Task | Set follow-up date | Task created automatically | Passed |  |
| Email Alert | Follow-up date is today | Email sent to counselor | Passed |  |

**Conclusion**  
  
The Student Enquiry Management System project demonstrates the complete lifecycle of a Salesforce implementation, starting from problem understanding to testing and deployment. By dividing the project into multiple phases, we were able to cover both administrative and developmental aspects of Salesforce, ensuring a well-rounded learning experience.  
  
Key highlights include:  
  
**Data Modeling & Relationships:** Designed standard and custom objects such as Student Enquiry and Student, with proper fields, record types, and relationships to support the use case.  
  
**Process Automation:** Implemented validation rules, flows, and triggers to automate repetitive tasks such as enquiry conversion and student creation.  
  
**User Interface Enhancements:** Configured record pages, home page layouts, and list views for a user-friendly experience.  
  
**Integration & Deployment:** Used named credentials and change sets to prepare the system for real-world extensibility and migration.  
  
**Reports & Dashboards:** Enabled stakeholders to track enquiries, conversions, and follow-ups effectively with interactive charts and reports.  
  
**Quality Assurance Testing:** Validated every automation and trigger through systematic test cases and Apex test classes, ensuring reliability.  
  
  
Overall, the project illustrates how Salesforce can streamline enquiry-to-admission processes in an educational setup by improving efficiency, data accuracy, and decision-making through automation and analytics.  
  
For future enhancements, this system can be extended with AI-driven lead scoring, chatbot integration, and Einstein Analytics to make the enquiry process even smarter and more predictive.