Child Sponsorship CRM: Project Documentation

Project Mentor: Gurram Annapurna

Project Overview

The **Child Sponsorship CRM** is a Salesforce-based solution designed for nonprofit organizations to efficiently manage sponsors, contributions, and beneficiary children. The CRM automates sponsor-child linkages, facilitates tracking of donations, and enables transparent communication via development updates and automated emails. This project integrates custom objects, declarative automation, and Apex programming to create a scalable and user-friendly application tailored for child sponsorship programs.

Objective

To design and implement a Salesforce-based Child Sponsorship CRM that automates and streamlines the management of sponsors, their contributions and beneficiary children. This system aims to improve operational efficiency for nonprofit staff, ensure transparent donor communication, and provide real-time tracking of sponsorship impact.

Key Features Implemented

- **Custom Data Model:** Created custom objects including Sponsor, Child, Sponsorship, Contribution, and Development Update with master-detail relationships to establish a robust, scalable data structure.
- **Automated Contribution Aggregation**: Developed an Apex trigger on the Contribution object that auto-calculates and updates total contributions on the related Sponsorship records after insert, update, or delete operations, showcasing programmatic automation.
- **Automated Email Communications**: Implemented record-triggered flows to send thank-you emails upon new contributions and scheduled flows to send reminders before sponsorship end dates.
- **Validation Rules**: Added a validation rule to ensure Sponsorship End Date must be after Start Date, maintaining data integrity.
- **Roll-Up Summary Fields**: Configured roll-up summaries for contribution totals and counts, visible on Sponsorship and Sponsor records.

- **Custom Tabs and Page Layouts**: Developed custom tabs for all objects with page layouts including relevant fields and related lists to enhance user experience.
- **Reporting and Dashboards**: Created reports and dashboards for monitoring active sponsorships, total contributions, and development updates, providing NGO management with key insights.
- **Security Model**: Set up profiles, field-level security, and permission sets to control access to sensitive data based on user roles.
- **Sample Data and Testing**: Populated sample data for sponsors, children, contributions, and sponsorships and validated workflows and triggers by manual testing.

System Requirements

- **Platform:** Salesforce (Developer Edition)
- **IDE:** Microsoft Visual Studio Code
- **Deployment Tool:** Salesforce CLI (SFDX)
- **Version Control:** Git / GitHub

Phase 1: Problem Understanding & Requirement Analysis

1.1 Problem Statement

Manual tracking and management of child sponsorship contributions, child progress updates, and sponsor communications create inefficiencies and risks for nonprofits. These processes often rely on spreadsheets and informal communication, leading to data inaccuracies, delayed acknowledgments, and reduced donor trust. The objective of this project is to build a centralized, automated Salesforce solution that streamlines contributions, automates communication, and improves transparency and operational efficiency.

1.2 The system must support the following functional requirements

1. **Centralized Sponsor and Child Records:** Create and manage detailed records for sponsors, children, and their sponsorship relationships.
2. **Contribution Tracking:** Capture financial contributions with linkage to sponsorships and automate contribution aggregations.
3. **Development Updates:** Enable NGO staff and volunteers to submit and track child development reports.
4. **Automated Notifications:** Send thank-you emails on contributions and reminders before sponsorship end via automation.
5. **Data Validation:** Ensure data integrity via validation rules such as start/end date correctness and required field enforcement.
6. **Reporting & Dashboards:** Provide real-time analytics for active sponsorships, contribution totals, and child progress for management.

1.3 Stakeholder Analysis

Key stakeholders identified include:

- **Sponsors:** Provide financial support tracked through the system.
- **Children:** Beneficiaries whose progress is documented.
- **NGO Staff and Volunteers:** Manage records, validations, and reporting.
- **Management:** Use dashboards for program oversight and decision making.

1.4 Business Process Mapping

The end-to-end business process is

Primary Workflow

1. Sponsor records are created.
2. Children records enter the system.
3. Sponsorship relationships are established linking sponsors and children.
4. Contributions are recorded against sponsorships.
5. Automated thank-you emails are sent upon each contribution.
6. Staff and volunteers submit development updates on children.
7. Reminder emails alert sponsors as sponsorships near completion.
8. Dashboards and reports provide ongoing performance insights.

Process Exceptions

- Incorrect or incomplete contribution details requiring correction.
- Sponsorships paused or terminated early.
- Child reassignment or sponsor changes.

1.5 App Exchange Exploration & Insights

An evaluation of similar nonprofit and child sponsorship management solutions on the Salesforce AppExchange was conducted. Key industry best practices include

- Automated financial tracking and donor communications.
- Child progress documentation modules.
- Strong reporting and transparency features.
- Role-based security models to protect sensitive data.

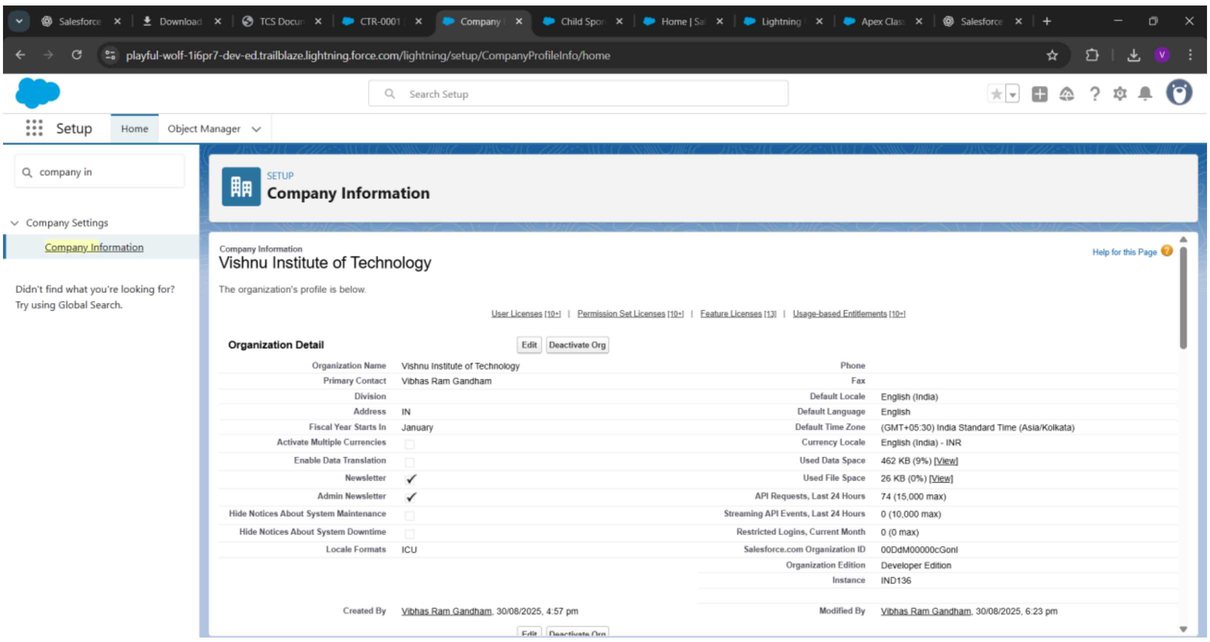
Phase 2: Org Setup & Configuration

Objective

The objective of this phase was to establish foundational org-wide settings and the basic security framework for the Child Sponsorship CRM application. This establishes a consistent user experience and protects sensitive sponsor and child data.

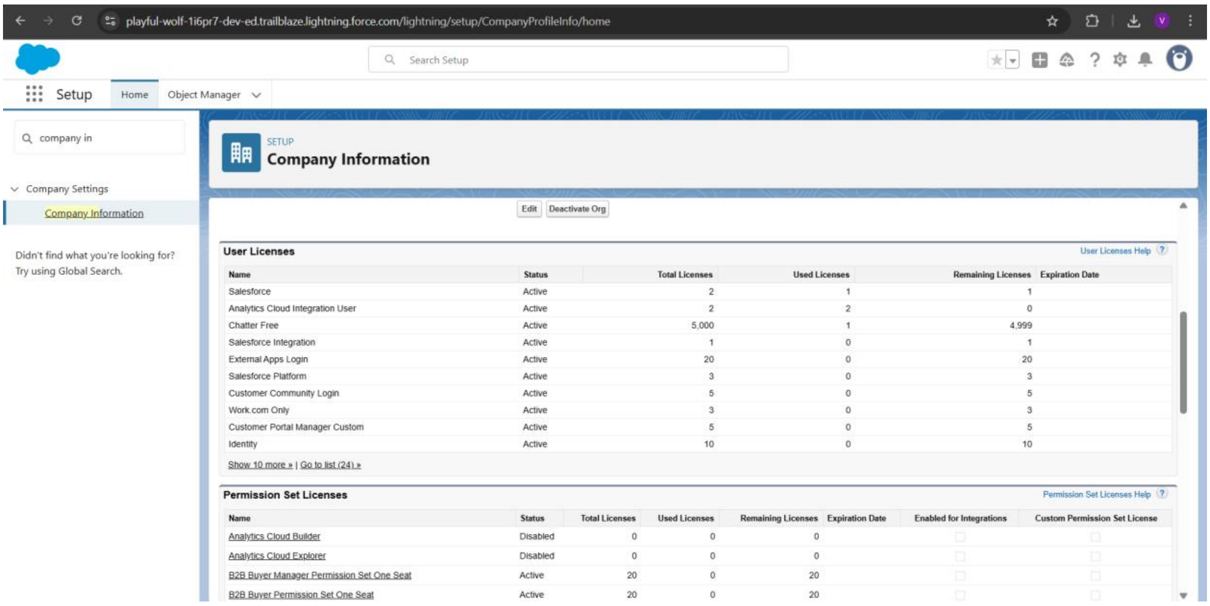
2.1 Company Information

- **Purpose:** Set the organization's default locale, time zone, and currency to ensure uniform display of dates, times, and currency throughout the application.
- **Implementation:**
    - Organization Name: [Your Organization Name]
    - Default Locale: English (India)
    - Default Time Zone: (GMT+05:30) India Standard Time (Asia/Kolkata)
    - Default Currency: Indian Rupee (INR) or USD, as per your context



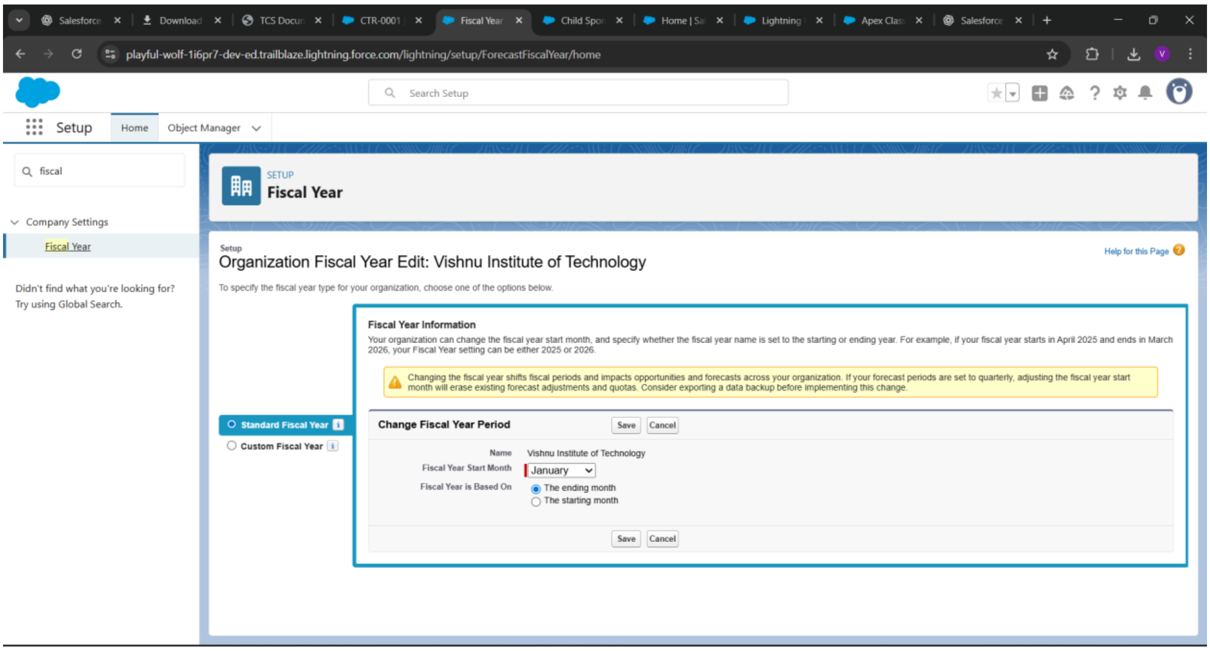
2.2 Business Hours

- **Purpose:** Define the working hours relevant for nonprofit operations to regulate time-based automations and notifications.
- **Implementation:**
  - Created "Nonprofit Business Hours": Monday to Friday, 9 AM to 6 PM IST.
  - No hours set on weekends; holidays not configured in this phase.



2.3 Fiscal Year

- **Purpose:** Set the financial year aligning reporting and forecasting periods with organizational accounting standards.
- **Implementation:**
  - Standard fiscal year configured, starting in January.

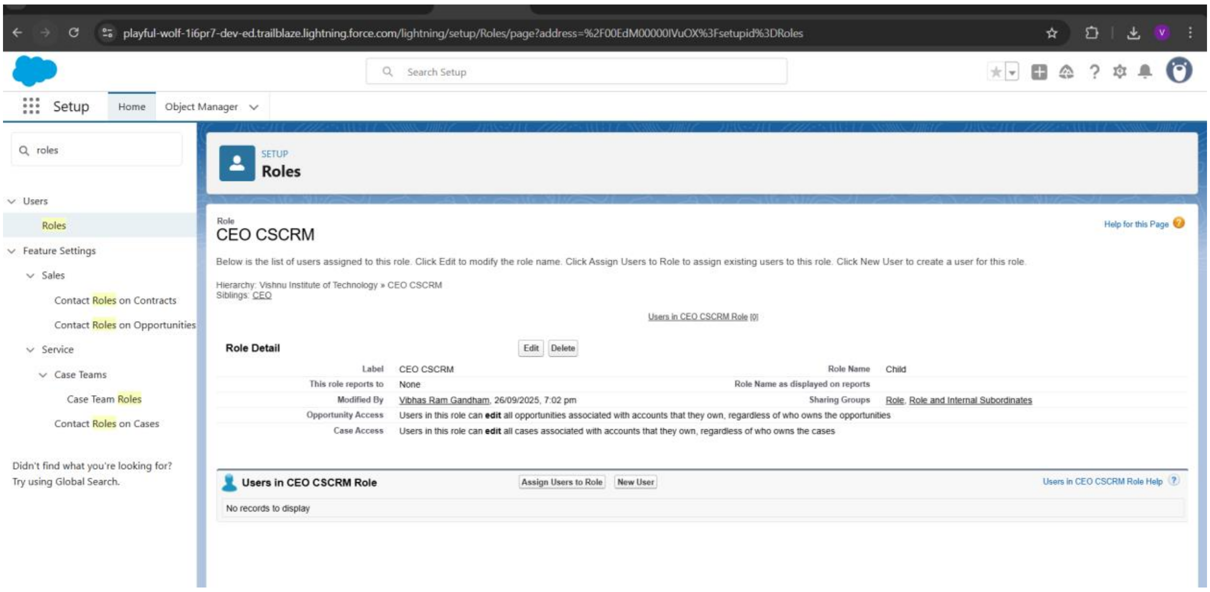


2.4 User Setup

- **Purpose:** Create and manage user records for all organizational roles accessing the CRM, assigning licenses and profiles tailored to responsibilities.
- **Implementation:**
    - Alongside system administrators, created users for NGO staff and volunteers with appropriate permissions to reflect operational needs.

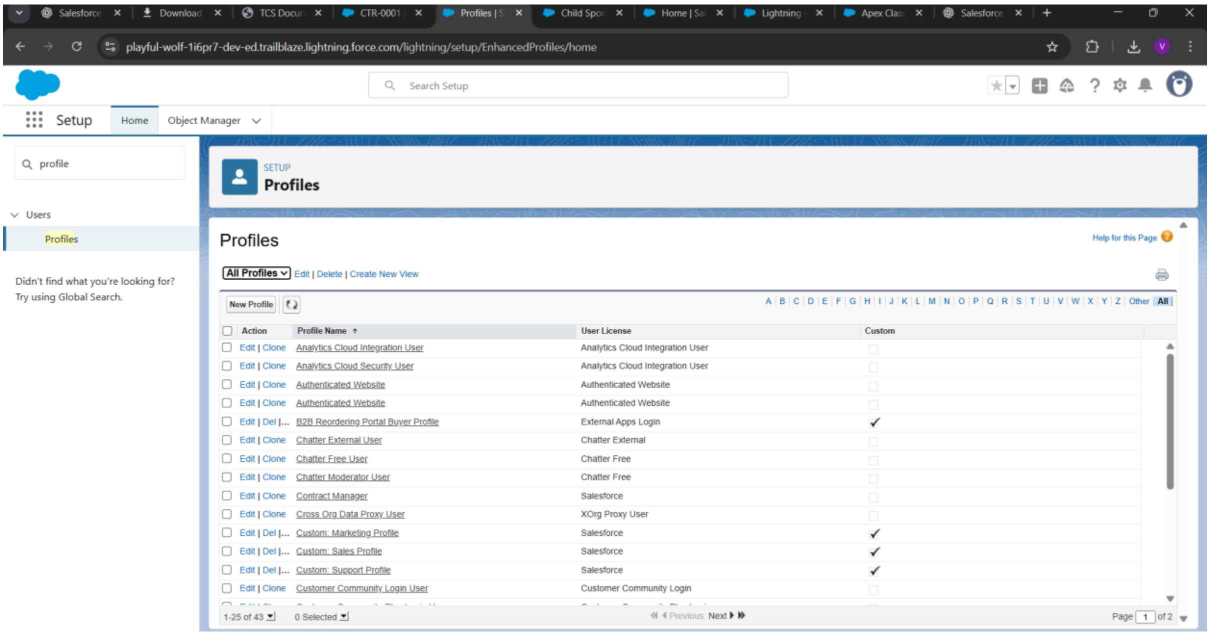
2.5 Roles

- **Purpose:** Define a role hierarchy to control data visibility, ensuring proper access for managers and field staff according to the nonprofit structure.
- **Implementation:**
    - Established a role hierarchy including roles like NGO Manager and Volunteer Worker reporting to Executive Director roles.



2.6 Profiles

- **Purpose:** Profiles define detailed object and field permissions controlling what users can see or modify in the CRM.
- **Implementation:**
    - Created custom profiles like "NGO Staff Profile" from standard profiles.
    - Granted CRED permissions on Sponsor, Child, Sponsorship, Contribution, and Development Update objects.
    - Enabled access to related tabs and page layouts.



2.7 Other Security Configurations

- **Permission Sets:** A baseline permission set was provisioned for future feature extensions. Currently, permissions are managed within profiles for simplicity.
- **Organization-Wide Defaults (OWD):** OWDs are set to private for core objects to protect confidential data.
- **Sharing Rules:** No custom sharing rules configured at this stage; possible future enhancements.

Phase 3: Data Modeling & Relationships

Objective:

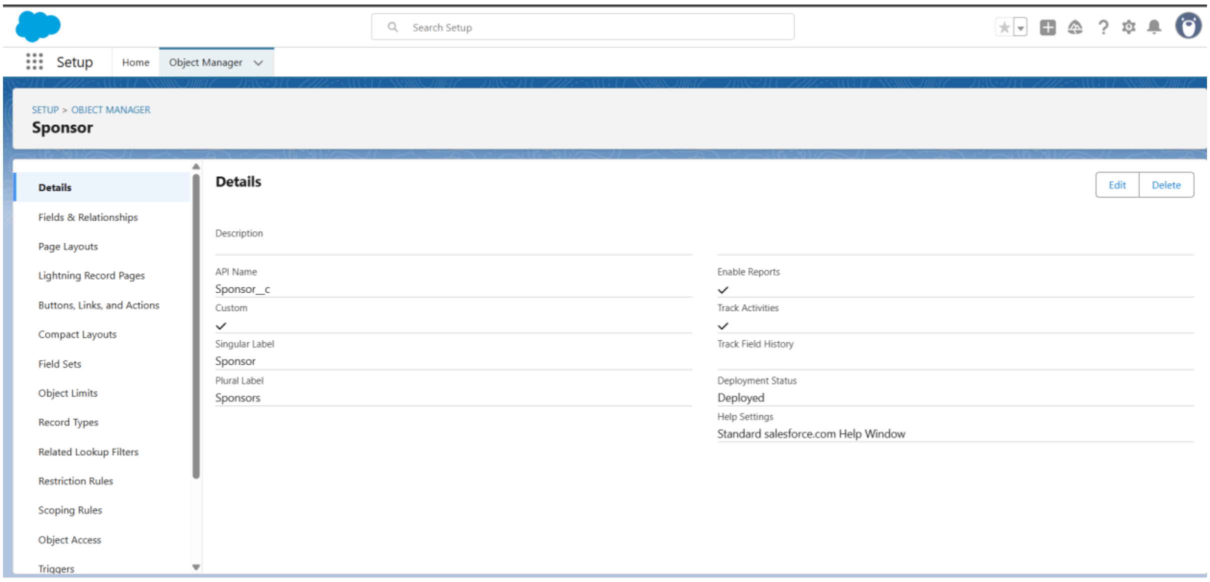
The aim of this phase was to build a robust, scalable data model that captures sponsors, children, their relationships, and transactions. A correct data model ensures data integrity, enables roll-ups and reporting, supports automation and apex, and provides a foundation for future expansions.

3.1 Sponsor Object (Sponsor__c)

This object represents individuals or entities who sponsor children. It is a key parent object in the data model.

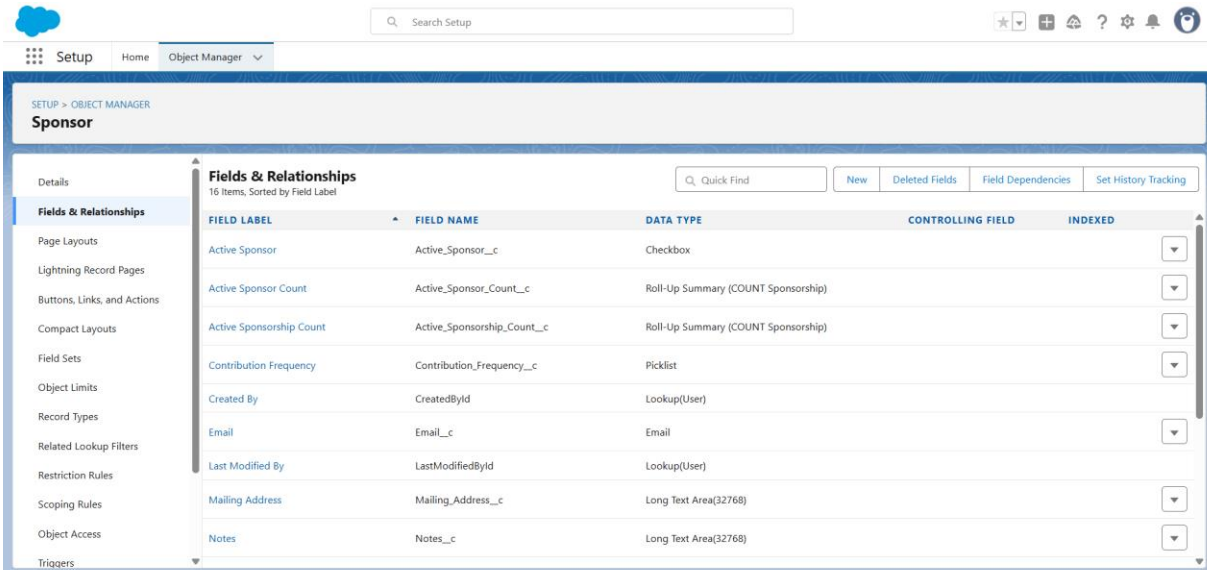
Property / Setting

- API Name: Sponsor__c
- Record Name Field: Sponsor Name (Text)
- Features enabled: Reports, Activities, Field History Tracking



Fields

Field Label	API Name	Data Type	Description / Purpose
Sponsor Name	Name (standard)	Text	The name of the sponsor
Email	Email__c	Email	Sponsor’s email address
Phone	Phone__c	Phone	Contact number
Sponsor ID	Sponsor_ID__c	Text	Unique identifier / external ID
Contribution Frequency	Contribution_Frequency__c	Picklist	E.g. One-time, Monthly, Quarterly
Preferred Payment Method	Preferred_Payment_Method__c	Picklist	E.g. Online, Bank Transfer, Cheque
Total Pledged	Total_Pledged__c	Currency	Aggregate pledged amount (optional)
Active Sponsor	Active__c	Checkbox	Mark if sponsor is active
Address	Mailing_Address__c	Long Text Area	Sponsor address
Notes	Notes__c	Long Text Area	Additional notes or remarks

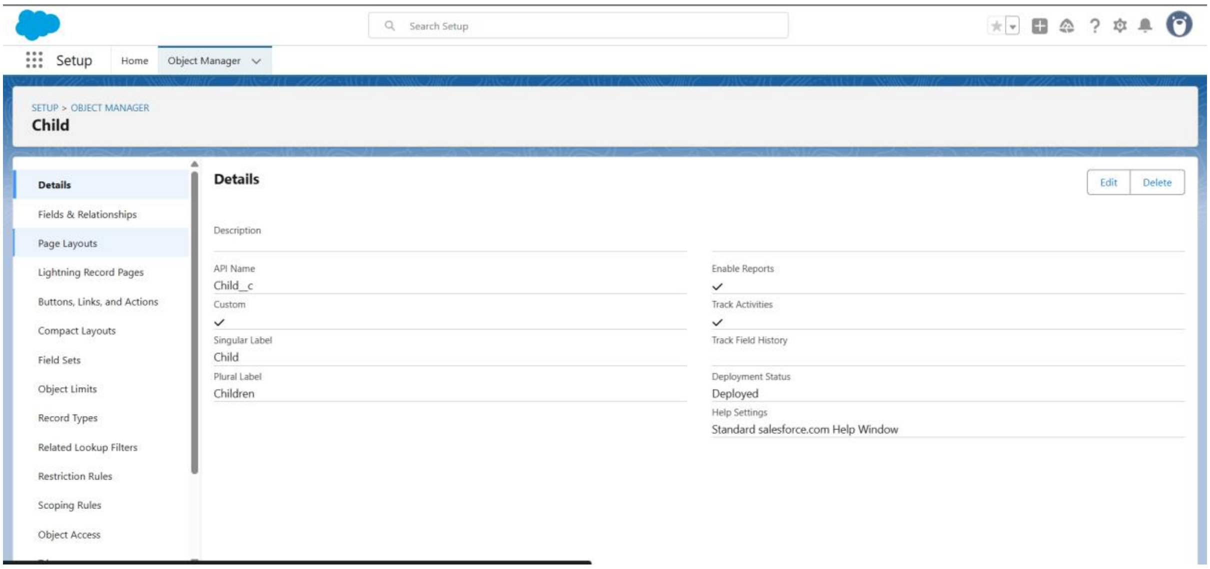


3.2 Child Object (Child__c)

This object stores the personal and developmental details of the child being sponsored.

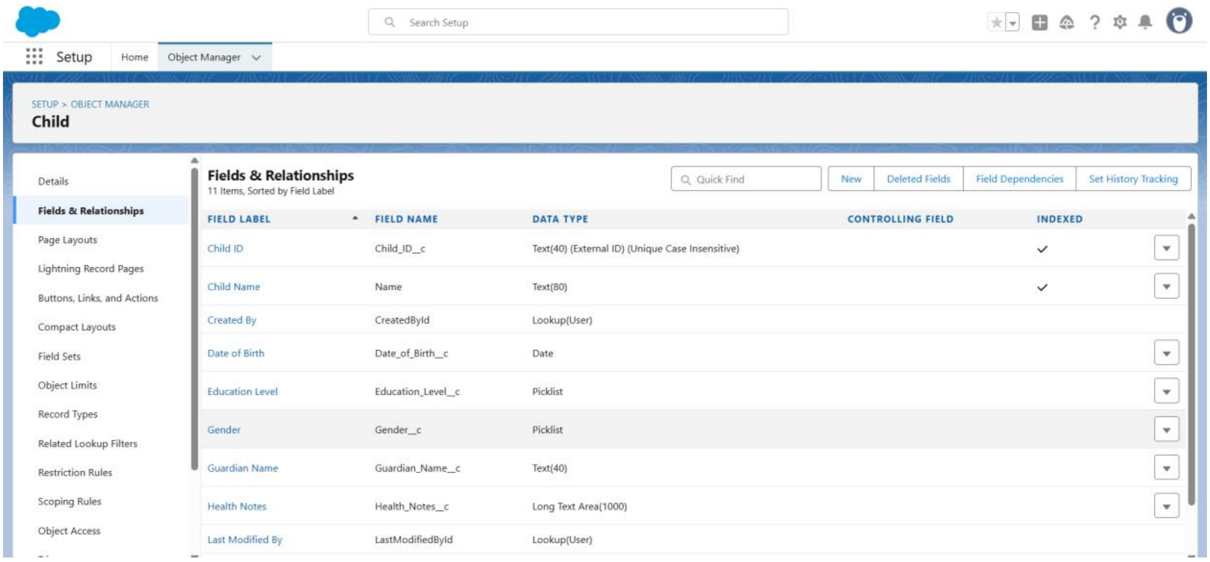
Property / Setting

- API Name: Child__c
- Record Name Field: Child Name (Text)
- Features: Reports, Activities, Field History Tracking



Fields

Field Label	API Name	Data Type	Description / Purpose
Child Name	Name (standard)	Text	Full name of the child
Date of Birth	Date_of_Birth__c	Date	Child's date of birth
Gender	Gender__c	Picklist	Male / Female / Other
Child ID	Child_ID__c	Text	Unique identifier / external ID
Education Level	Education_Level__c	Picklist	Pre-school / Primary / Secondary / Higher
Guardian Name	Guardian_Name__c	Text	Name of guardian or parent
Address	Address__c	Long Text Area	Residential address
Health Notes	Health_Status__c	Long Text Area	Medical or health-related info
Photo	Photo__c	URL (or Text)	Link to child's image (or use Files)
Age (Formula)	Age__c	Formula (Number)	e.g., FLOOR((TODAY() - Date_of_Birth__c)/365)



3.3 Sponsorship Object (Sponsorship__c)

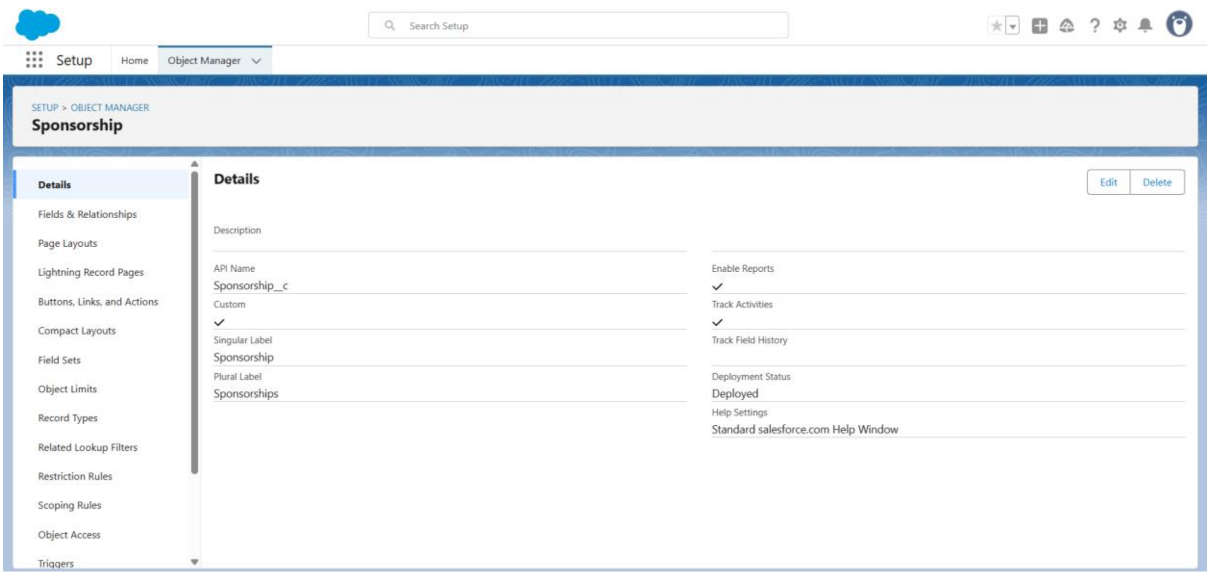
This is the junction object connecting Sponsor and Child. It holds metadata about the sponsorship relationship.

Property / Setting

- API Name: Sponsorship__c
- Record Name: Auto Number (e.g. SP-{0000})
- Features: Reports, Activities

Relationships

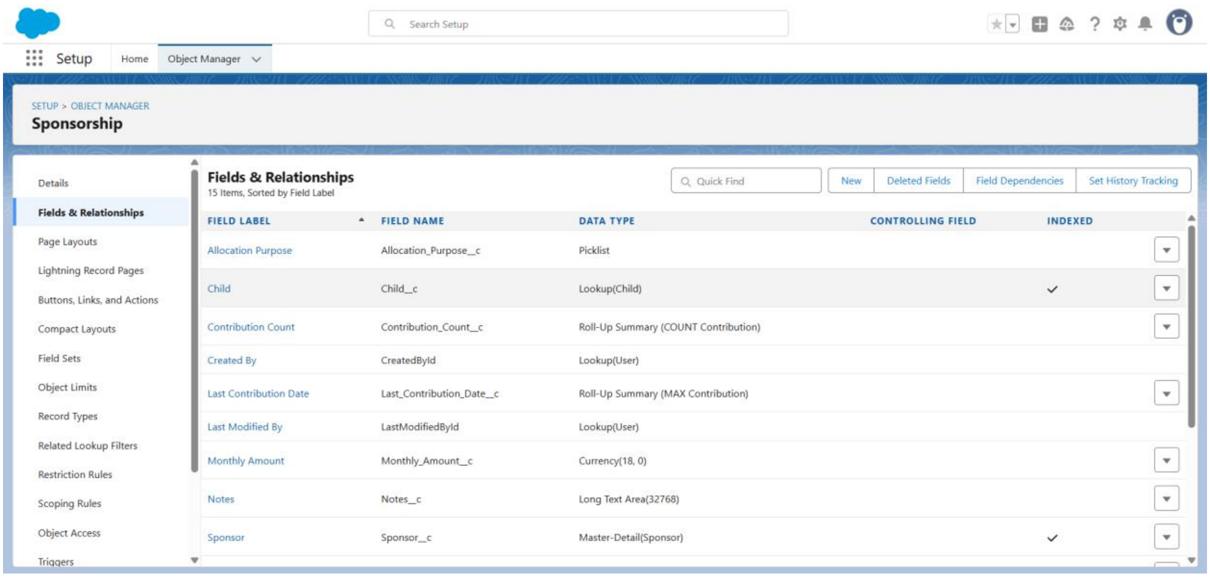
- Master-Detail → Sponsor__c
- Look Up→ Child__c



Fields

Field Label	API Name	Data Type	Description / Purpose
Sponsorship Start Date	Start_Date__c	Date	When the sponsorship begins
Sponsorship End Date	End_Date__c	Date	Optional end date
Sponsorship Status	Status__c	Picklist	Active / Paused / Ended / Terminated
Monthly Amount	Monthly_Amount__c	Currency	Amount expected per

Field Label	API Name	Data Type	Description / Purpose
			period
Total Contributions	Total_Contributions__c	Currency	Sum of contributions, via Apex or roll-up
Allocation Purpose	Allocation_Purpose__c	Picklist	Education / Health / Food / Other
Notes	Notes__c	Long Text Area	Additional notes



3.4 Contribution Object (Contribution__c)

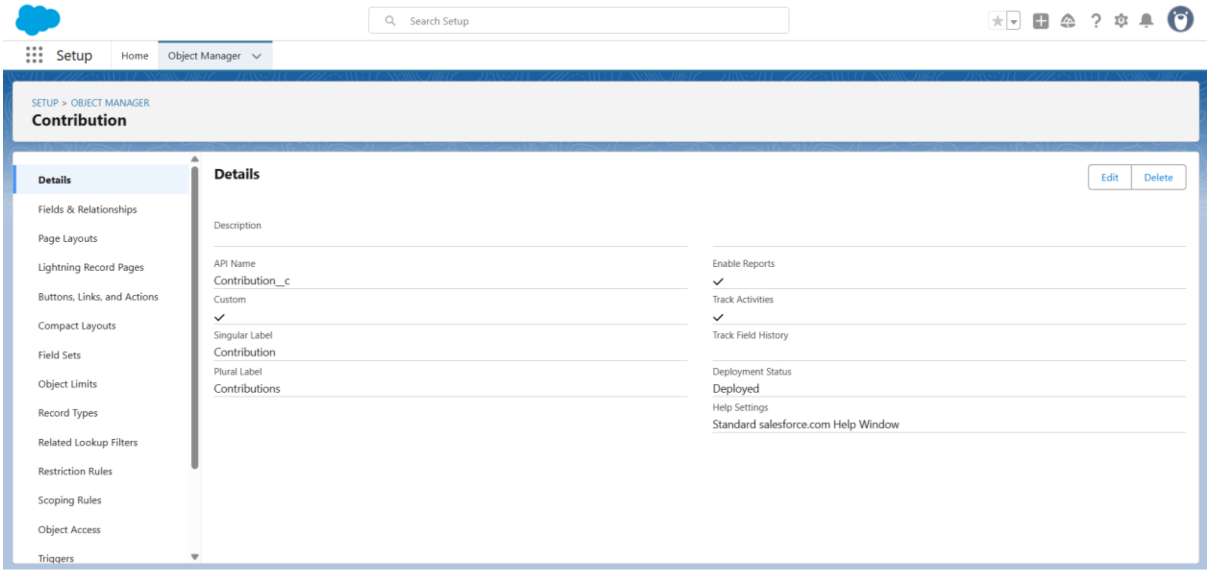
This object captures each payment made under a given sponsorship.

Property / Setting

- API Name: Contribution__c
- Record Name: Auto Number (e.g. CTR-{0000})
- Features: Reports, Activities

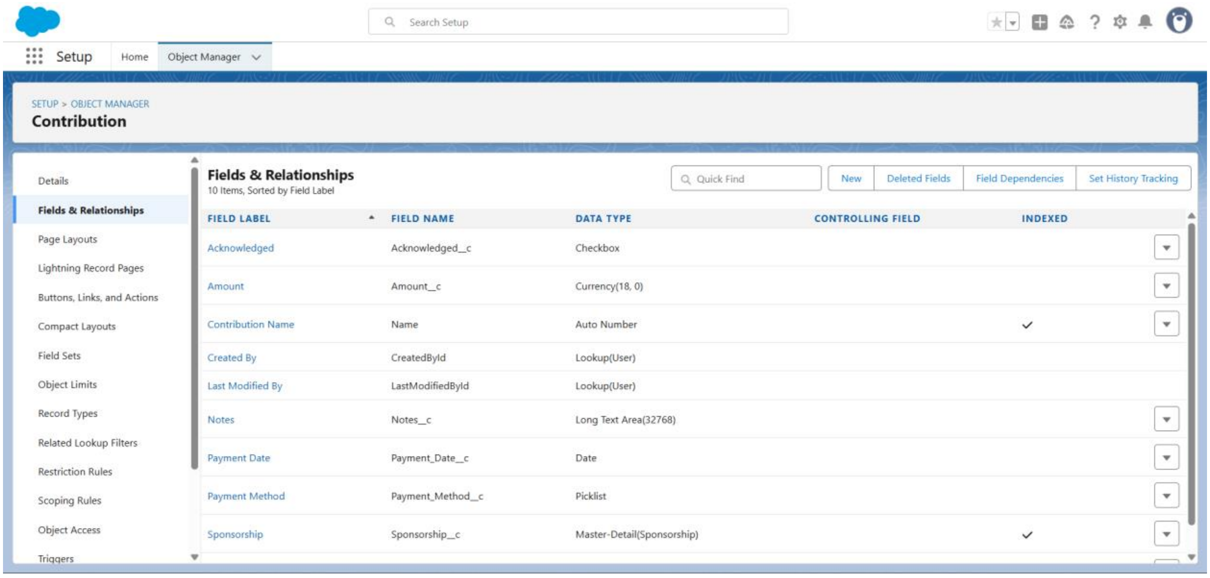
Relationship

- Master-Detail → Sponsorship__c



Fields

Field Label	API Name	Data Type	Description / Purpose
Amount	Amount__c	Currency	Payment amount
Payment Date	Payment_Date__c	Date	Date of payment
Payment Method	Payment_Method__c	Picklist	Online / Bank Transfer / Cheque
Transaction ID	Transaction_ID__c	Text	Reference ID
Acknowledged	Acknowledged__c	Checkbox	Mark when thank-you sent
Notes	Notes__c	Long Text Area	Additional notes



3.5 Development Update Object (Development_Update__c)

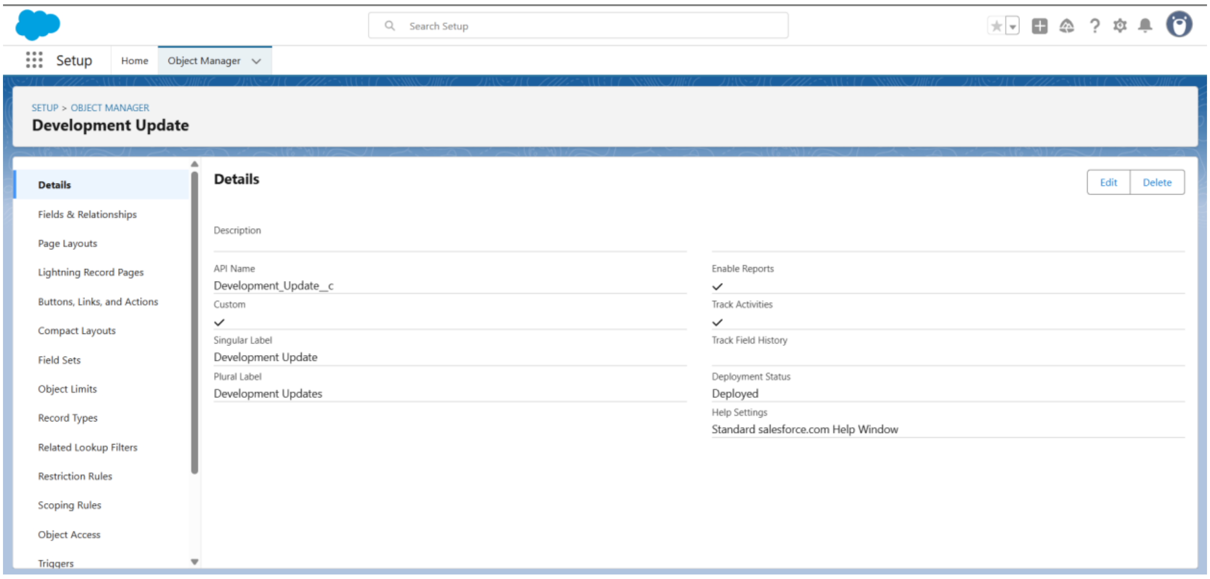
This object records periodic updates about a child's progress (education, health, behavior).

Property / Setting

- API Name: Development_Update__c
- Record Name: Auto Number or Text
- Features: Reports, Activities

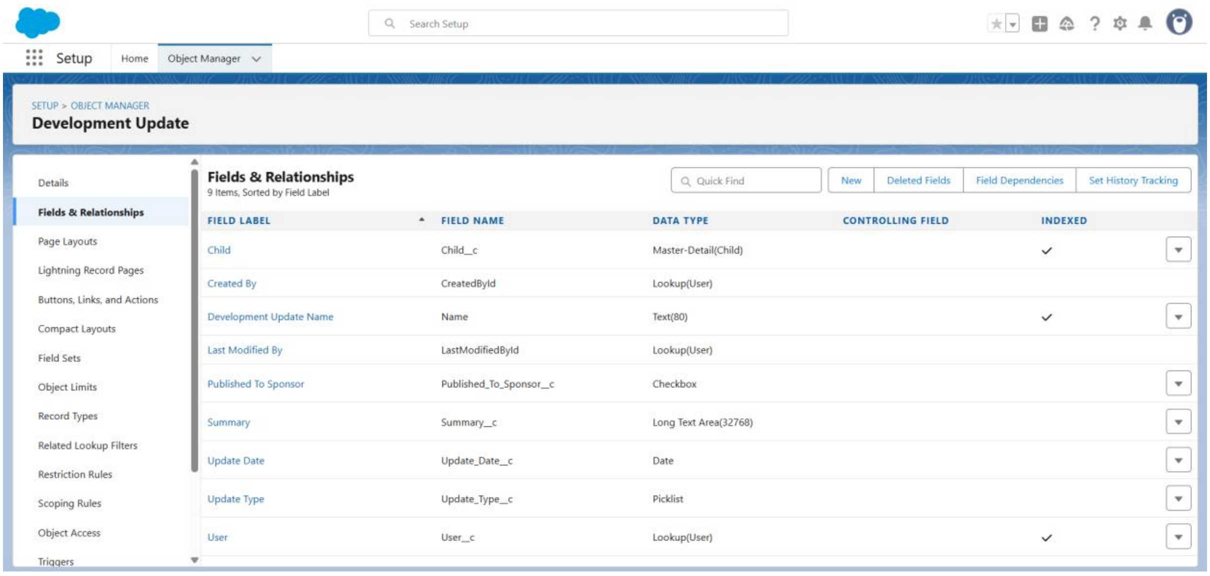
Relationship

- Master-Detail or Lookup → Child__c



Fields

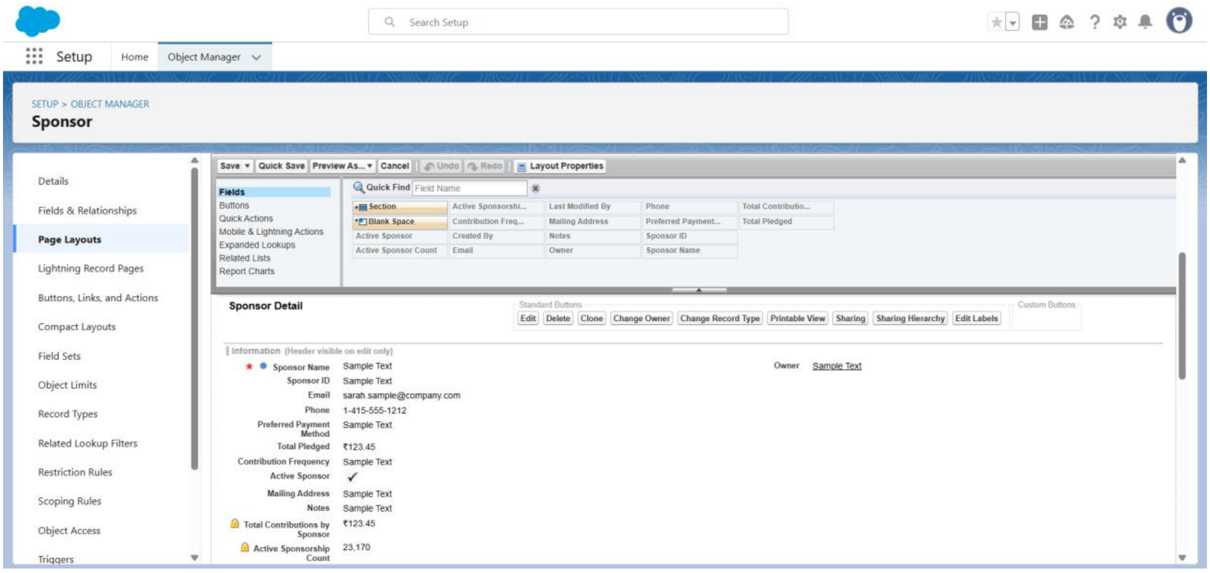
Field Label	API Name	Data Type	Description / Purpose
Update Date	Update_Date__c	Date	When the update was recorded
Update Type	Update_Type__c	Picklist	Education / Health / Behavior / Other
Summary	Summary__c	Long Text Area	Summary of the progress
Published To Sponsor	Published_To_Sponsor__c	Checkbox	Indicates if sponsor has been notified
Submitted By	Submitted_By__c	Lookup(User)	The user who recorded the update
Files (images/docs)	(Use Files related list)	—	Uploads such as photos or documents



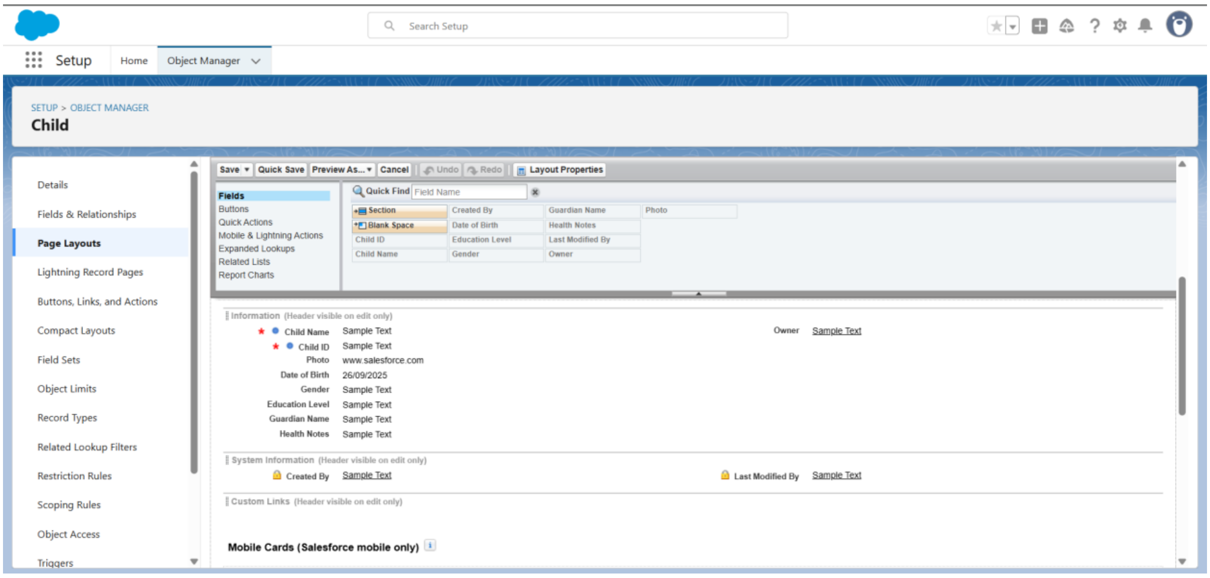
3.6 Page Layouts & Related Lists

I configured page layouts for each object to ensure clean UI and relationship visibility:

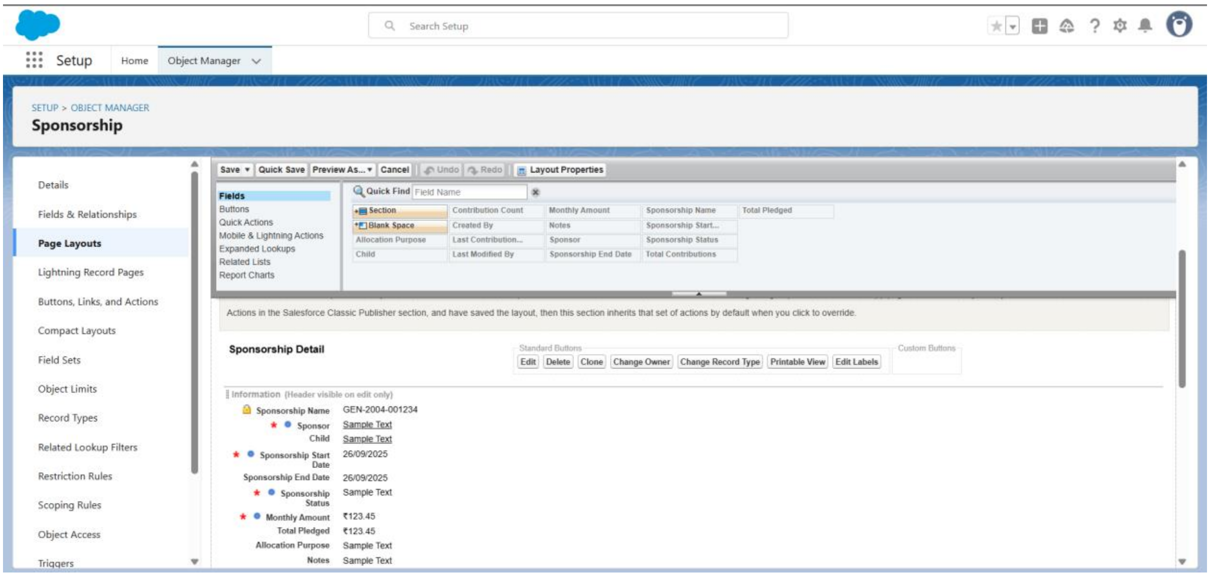
- **Sponsor Layout:** Fields like Email, Phone, Total Pledged; and related lists Sponsorships & Contributions.



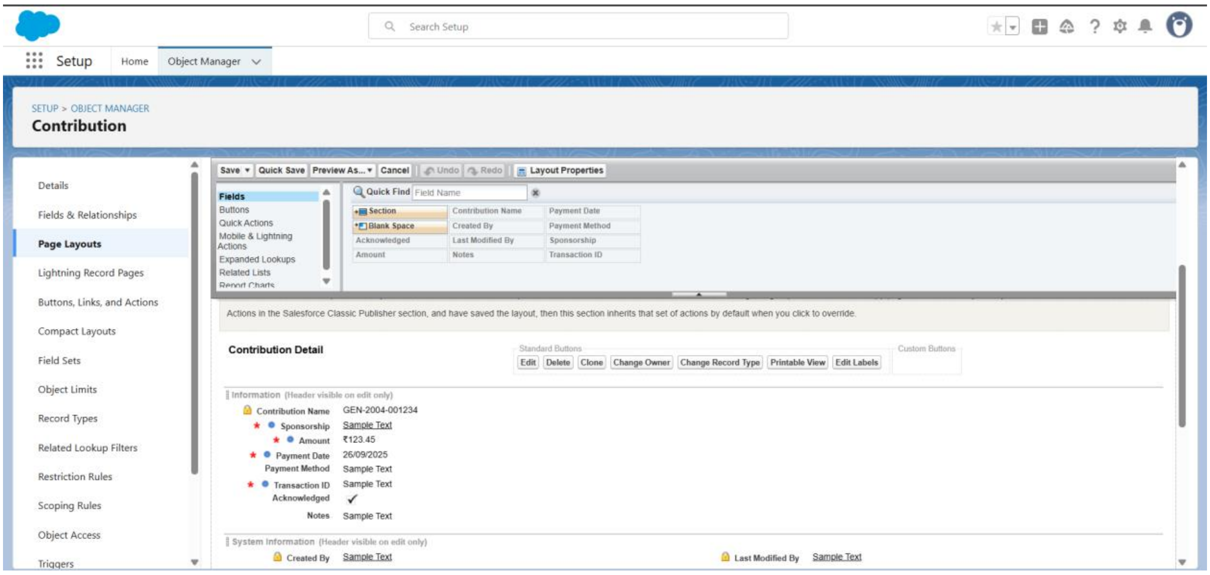
- **Child Layout:** Fields including DOB, Gender, Education Level, Photo; related lists Development Updates & Sponsorships.



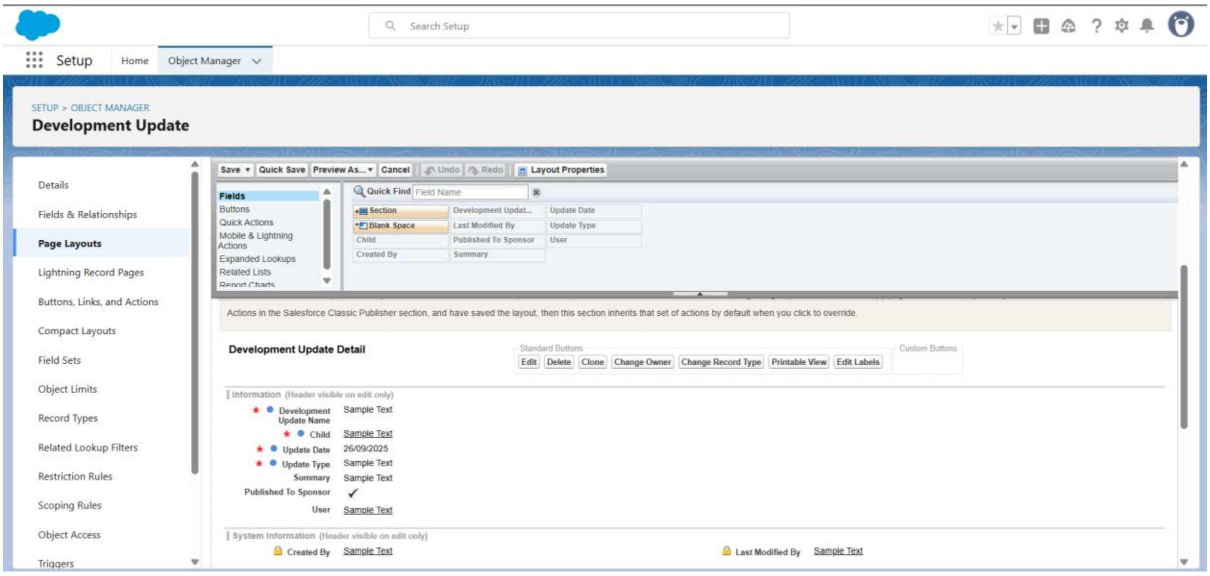
- **Sponsorship Layout:** Child, Sponsor, Start/End dates, Status, Total Contributions; related list Contributions.



- **Contribution Layout**: Amount, Payment Date, Method.



- **Development Update Layout:** Update Date, Type, Summary, Submitted By, and Files related list.



3.7 Relationship Summary & Design Rationale

- The design is one-to-many:
  - A Sponsor can support many Sponsorships
  - A Child can have multiple Sponsorships (at different times)
  - Each Sponsorship can have many Contribution records
  - Each Child can have many Development Updates
- Using Master-Detail relationships allows roll-up summaries (e.g. sum of contributions) and cascading behavior (deleting parent deletes child).
- The model is scalable: more objects (e.g. Payment, Feedback) can be added later without big changes.

Phase 4: Process Automation (Admin)

4.1 Validation Rules

Purpose:

To maintain data quality and integrity by enforcing business rules when users input or edit data.

Implementation:

- Validation rule added on Sponsorship to ensure the End Date is always after the Start Date to prevent logical errors.
- Additional validation to require Sponsor email and Contribution amount to be non-empty and valid.

4.2 Workflow Rules

Purpose:

Automate actions based on record criteria changes to improve responsiveness and user efficiency.

Implementation:

- Not heavily used — flows preferred for greater flexibility in this project.

4.3 Process Builder

Purpose:

To automate multi-step business processes declaratively without code.

Implementation:

- Initially considered but replaced by Flow Builder for enhanced flexibility.

4.4 Flow Builder (Screen, Record-Triggered, Scheduled, Auto-launched)

Purpose:

Automate business processes with rich automation capabilities including screen flows for guided interactions, record-triggered and scheduled flows for backend automations.

Implementation:

- Thank-You Email Flow: A record-triggered flow sends personalized thank-you emails automatically when a Contribution record is created.
- Sponsorship Reminder Email Flow: A scheduled flow runs to remind sponsors before their sponsorship ends.
- Record Updates: Flow updates Sponsorship statuses based on Contribution activity.

4.5 Email Alerts

Purpose:

Communicate with sponsors and internal users automatically to improve engagement and operational efficiency.

Implementation:

- Email templates created and linked to flows for sending thank-you and reminder emails.
- Alerts sent to designated NGO staff on new Development Updates submitted.

4.6 Field Updates

Purpose:

Automatically maintain calculated or status fields to reflect current record states.

Implementation:

- Field updates in flows to set Sponsorship Status as "Active," "Completed," or "Pending" based on dates and contributions.

4.7 Tasks

Purpose:

Automatically generate follow-up tasks to ensure timely completion of key actions.

Implementation:

- Flow assigns tasks to staff for follow-up on pending Development Updates or sponsor reminders.

4.8 Custom Notifications

Purpose:

Enhance user awareness with in-app notifications for critical events.

Implementation:

- Notifications configured for NGO staff when new contributions or updates require review.

Phase 5: Apex Programming (Developer)

Objective:

In this phase we introduced a foundational level of programmatic logic using Apex, to complement the declarative features (flows, validation rules, etc.). The aim was not to replace point-and-click tools, but to showcase an example of custom logic for use cases that go beyond what Flows or Roll-up summaries can cover reliably.

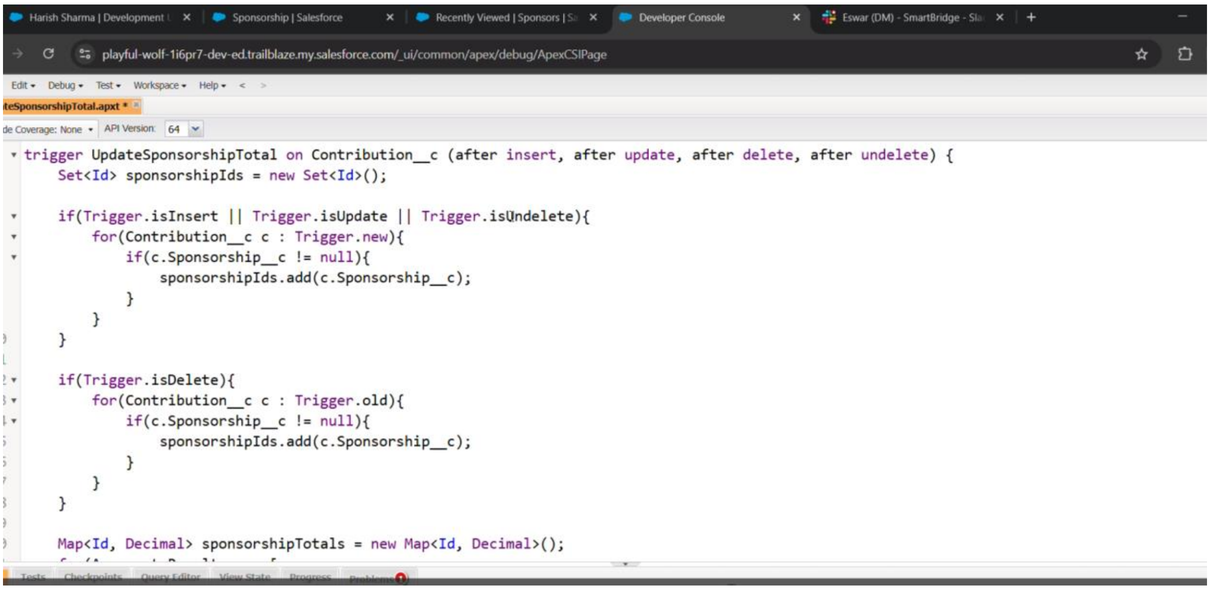
5.1 Apex Trigger: UpdateSponsorshipTotal

Purpose:

The trigger ensures that whenever a Contribution record is created, updated, or deleted, the Total Contributions field on the related Sponsorship record is always up-to-date. This avoids data discrepancies and reduces manual updating.

Implementation Details:

- Trigger Type / Events: after insert, after update, after delete, after undelete on Contribution__c
- Logic:
  - Collect the set of Sponsorship Ids affected by the contribution changes
  - Aggregate (SUM) the Amount__c field grouped by Sponsorship__c
  - Update corresponding Sponsorship records with the new totals
- Using after context ensures you can query the Contribution object for existing values and perform updates safely without mutating the record being inserted.



Trigger Code (UpdateSponsorshipTotal.trigger):

```
trigger UpdateSponsorshipTotal on Contribution__c (after insert, after update, after delete, after undelete) {
    Set<Id> sponsorshipIds = new Set<Id>();

    if(Trigger.isInsert || Trigger.isUpdate || Trigger.isUndelete) {
        for(Contribution__c c : Trigger.new) {
            if(c.Sponsorship__c != null) {
                sponsorshipIds.add(c.Sponsorship__c);
            }
        }
    }
    if(Trigger.isDelete) {
        for(Contribution__c c : Trigger.old) {
            if(c.Sponsorship__c != null) {
                sponsorshipIds.add(c.Sponsorship__c);
            }
        }
    }
```

```
    // Query aggregated sums
    Map<Id, Decimal> totalsMap = new Map<Id, Decimal>();
    for (AggregateResult ar : [
        SELECT Sponsorship__c, SUM(Amount__c) totalAmt
        FROM Contribution__c
        WHERE Sponsorship__c IN :sponsorshipIds
        GROUP BY Sponsorship__c
    ]) {
        totalsMap.put((Id)ar.get('Sponsorship__c'), (Decimal) ar.get('totalAmt'));
    }

    List<Sponsorship__c> updates = new List<Sponsorship__c>();
    for (Id sid : sponsorshipIds) {
        Sponsorship__c s = new Sponsorship__c(Id = sid);
        s.Total_Contributions__c = totalsMap.containsKey(sid) ? totalsMap.get(sid) : 0;
        updates.add(s);
    }

    if (!updates.isEmpty()) {
        update updates;
    }
}
```

5.2 Test Class: UpdateSponsorshipTotalTest

To adhere to Salesforce requirements (minimum 75% code coverage), I created a test class. The test ensures that the trigger works across insert, update, and delete operations.

Code:

```
@isTest
private class UpdateSponsorshipTotalTest {
    @isTest static void testContributionInsertUpdateDelete() {
        // Setup: Create Sponsor, Child, Sponsorship
        Sponsor__c sp = new Sponsor__c(Name='Test Sponsor',
Email__c='sponsor@example.com');
        insert sp;
        Child__c ch = new Child__c(Name='Child Test', Date_of_Birth__c =
Date.today().addYears(-10));
        insert ch;
        Sponsorship__c s = new Sponsorship__c(Sponsor__c = sp.Id, Child__c = ch.Id,
                                              Start_Date__c = Date.today(), Status__c = 'Active');
        insert s;

        // 1. Insert Contribution
        Contribution__c c1 = new Contribution__c(Sponsorship__c = s.Id, Amount__c =
100, Payment_Date__c = Date.today());
        insert c1;

        Sponsorship__c sAfter1 = [SELECT Total_Contributions__c FROM
Sponsorship__c WHERE Id = :s.Id];
        System.assertEquals(100, sAfter1.Total_Contributions__c, 'After first
contribution, total should be 100');

        // 2. Insert another Contribution
        Contribution__c c2 = new Contribution__c(Sponsorship__c = s.Id, Amount__c =
50, Payment_Date__c = Date.today());
        insert c2;
```

Sponsorship__c sAfter2 = [SELECT Total_Contributions__c FROM Sponsorship__c WHERE Id = :s.Id];

System.assertEquals(150, sAfter2.Total_Contributions__c, 'After second contribution, total should be 150');

// 3. Delete one contribution

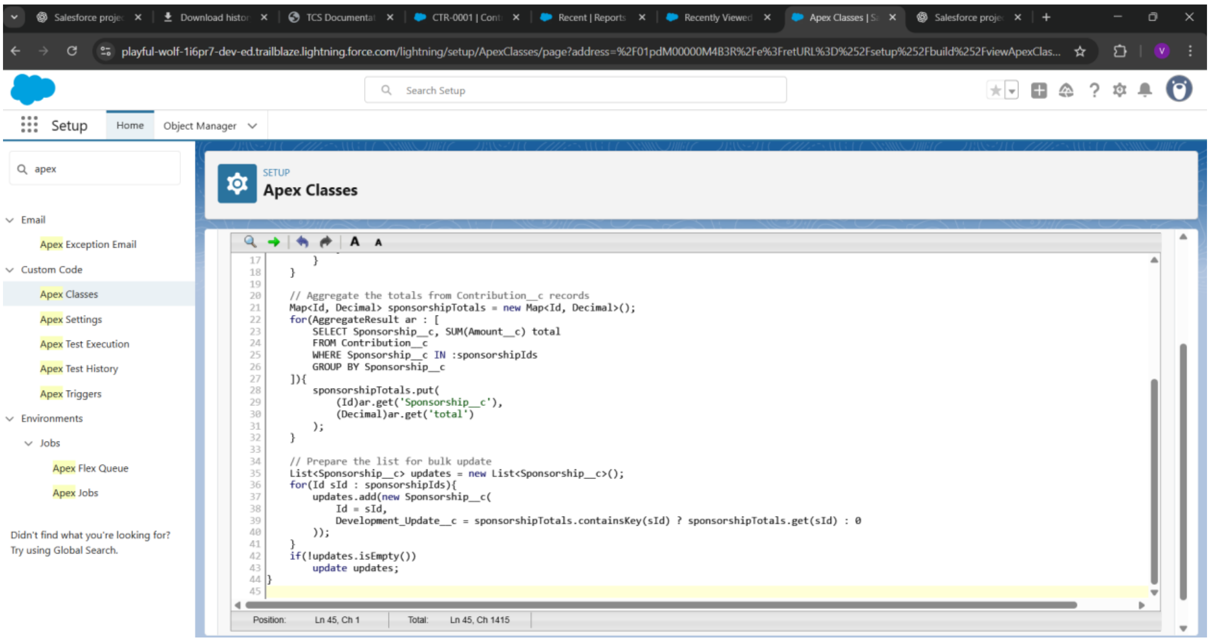
delete c1;

Sponsorship__c sAfterDel = [SELECT Total_Contributions__c FROM Sponsorship__c WHERE Id = :s.Id];

System.assertEquals(50, sAfterDel.Total_Contributions__c, 'After deletion, total should reflect remaining contributions');

}

}



5.3 Future Apex Enhancements (Conceptual Use Cases)

Below are potential advanced features you might implement with Apex in the future:

- **Batch Apex for Data Cleanup:**

  E.g. periodically find Contribution records older than a certain date or

Sponsorships without recent contributions, and flag them or archive them.

- **Queueable Apex for External APIs / Callouts:**
  If sponsors pay via external payment gateways or you have a donor portal, you might queue asynchronous calls from Salesforce to external systems upon contribution creation.
- **Scheduled Apex:**
  Automate periodic jobs, such as resetting statuses, expiring sponsorships, sending scheduled summary emails, or refreshing external data.
- **Trigger Framework and Handler Layer:**
  Refactor consistently using a trigger-handler pattern, so your logic scales, avoids recursion, and is modular.

Phase 6: User Interface Development

Objective

The objective of this phase was to design a clean, intuitive, and role-appropriate user interface in Salesforce. The goal was to make navigation simple, present records in a meaningful way, and help users interact with data without confusion. This phase includes creating custom tabs, a dedicated Lightning App, and customizing record pages via the Lightning App Builder.

6.1 Custom Tabs

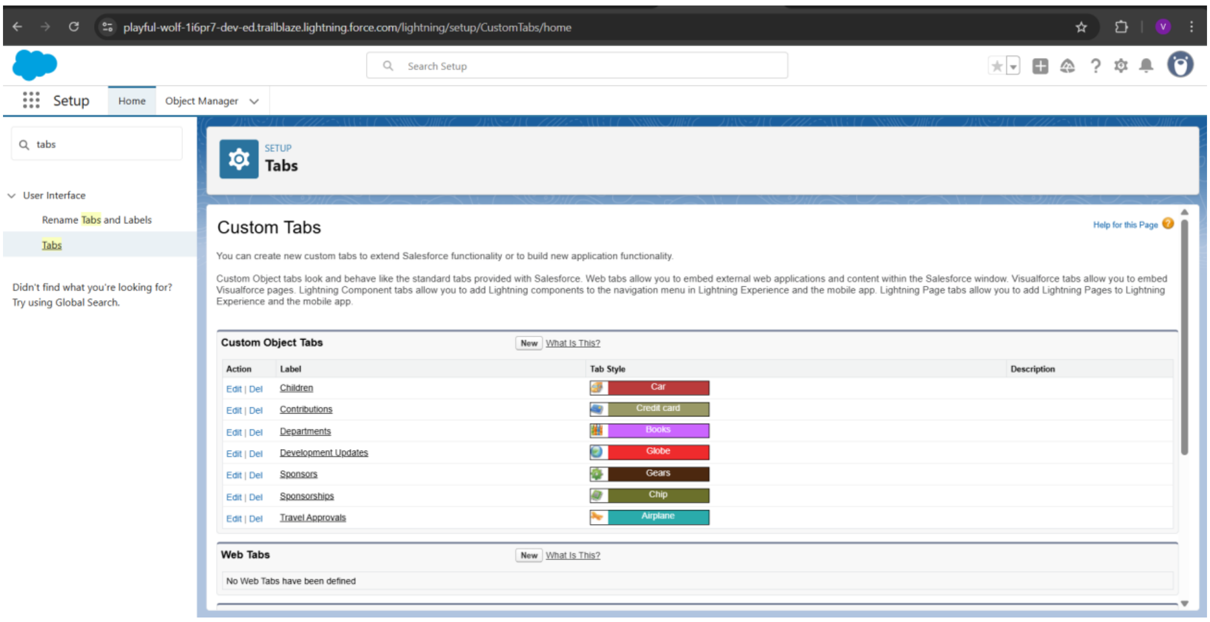
Purpose

Custom tabs allow users direct access to each object's records from the navigation bar. They make it easy to access Sponsors, Children, Sponsorships, Contributions, and Development Updates.

Implementation

I created the following custom object tabs:

- **Sponsors**: Lists all sponsor records.
- **Children**: Shows child records and their details.
- **Sponsorships**: Allows users to view and manage sponsor–child relationships.
- **Contributions**: Displays payment records.
- **Development Updates**: Shows progress reports for children.



6.2 Lightning App: Child Sponsorship CRM

Purpose

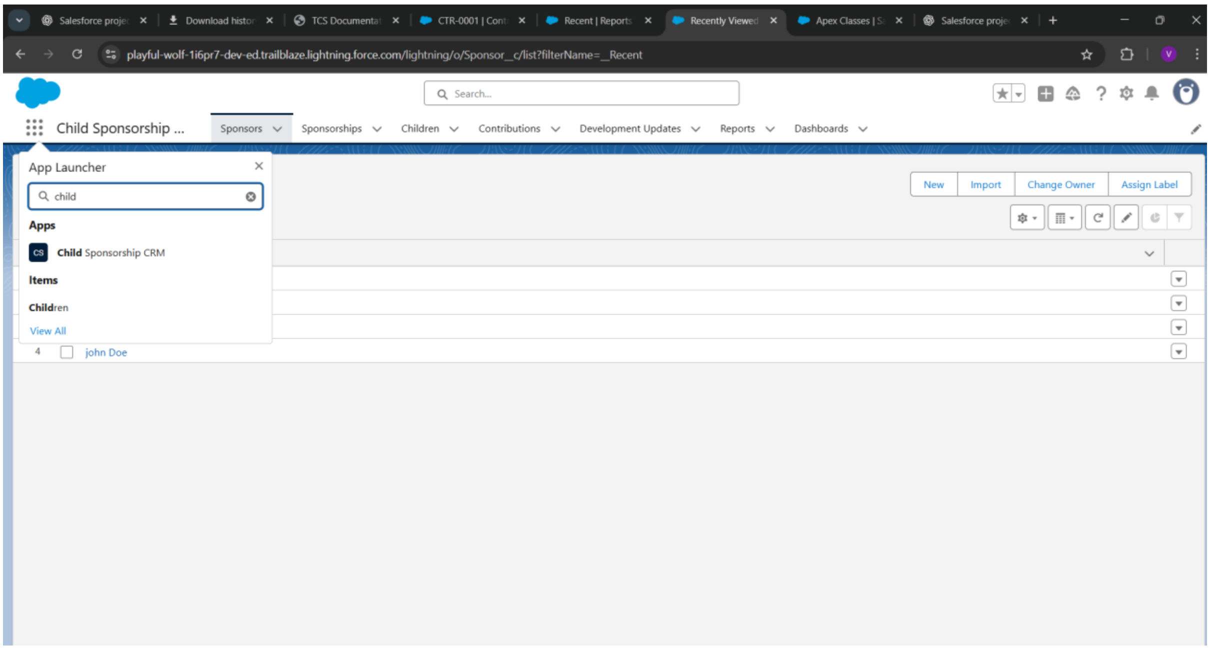
A dedicated Lightning App ensures users access the CRM in a focused environment, with all relevant tabs and tools in one place — no distractions, clean workspace.

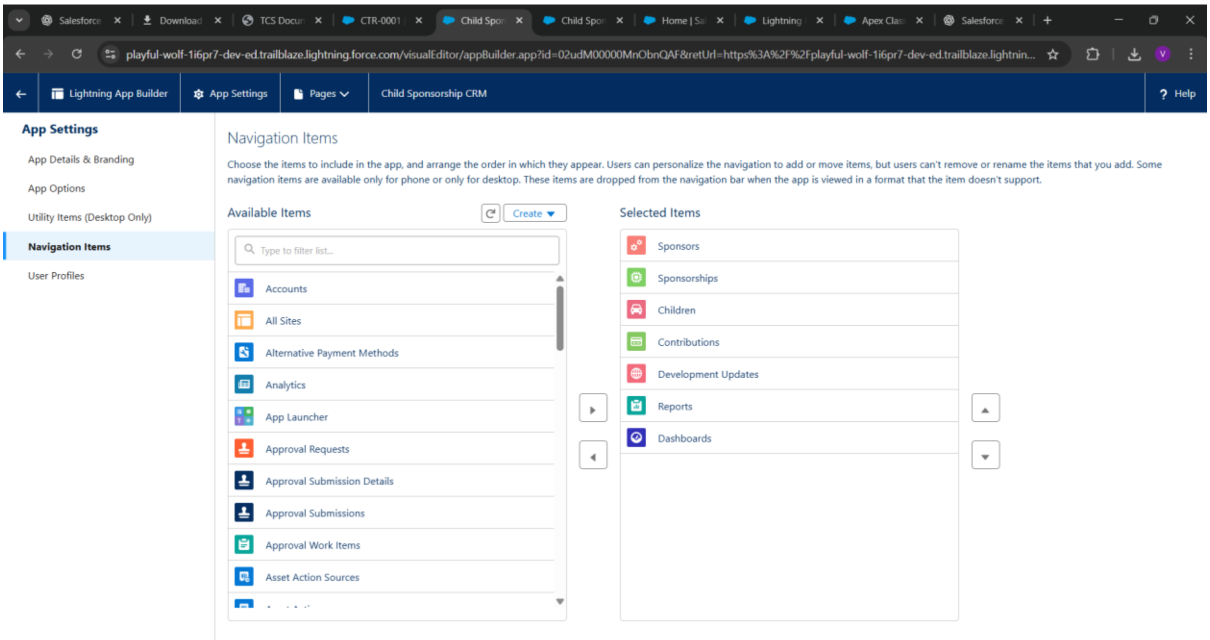
Implementation Details

- **App Details & Branding**:

  • App Name: Child Sponsorship CRM

  • Assigned a theme color (e.g. a calm blue/green) and an appropriate logo (such as a child and sponsor icon) so it's easily identifiable in the App Launcher.





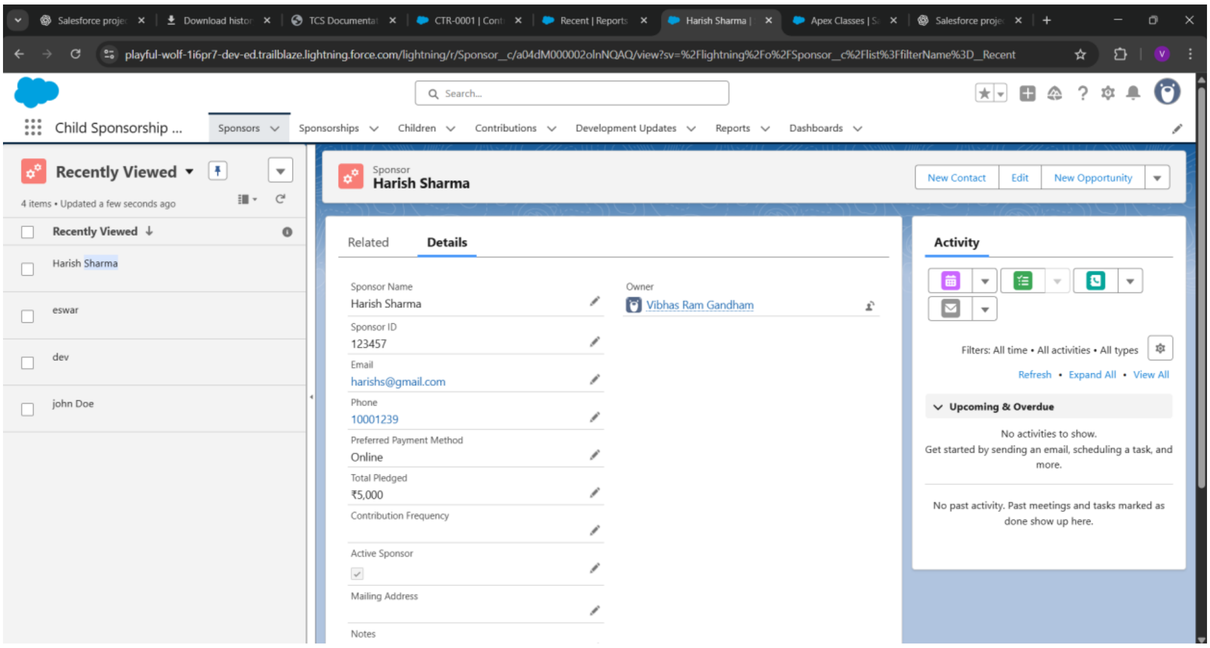
- **Navigation Items**:

  Configured the app's navigation bar to include the key tabs in logical order, such as:

  1. Home
  2. Reports
  3. Dashboards
  4. Sponsors
  5. Children
  6. Sponsorships
  7. Contributions
  8. Development Updates

- **Profile Assignment / Access Control**:

  The app is assigned so that only authorized profiles (System Administrator, NGO Staff, possibly Volunteer) can see and use it. This keeps the workspace secure and relevant to user roles.



6.3 Lightning Record Page: Sponsorship Record Page (example)

You might create custom record pages for one or more objects (Sponsor, Child, Sponsorship) using the Lightning App Builder. Here is one example — you can adapt for your objects.

Purpose

Standard page layouts are static; custom Lightning Record Pages allow richer layouts, components, charts, related lists, and conditional visibility—all improving user experience.

Implementation Details

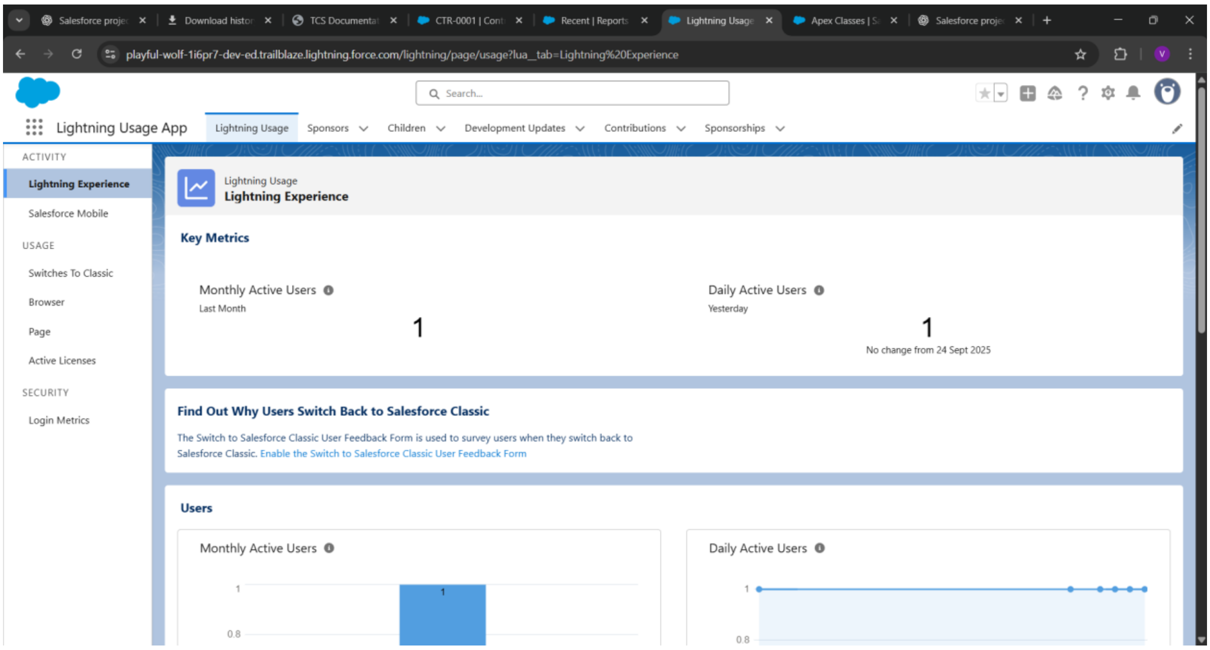
- **Page Template**:
  I selected a layout with header + two columns or header + right sidebar (based on what fits best) so content can be organized visually.
- **Components Placed**:
  • **Header / Highlights Panel**: Placed fields like Sponsorship Status, Start Date, Total Contributions for quick glance.
  • **Path Component** (if used): Shows the stages of a sponsorship (e.g. Proposed → Active → Ended).
  • **Left / Main Region**:
  – Record Details component: shows all fields from the page layout (Sponsor, Child, Dates, Amount, etc.).
  – Related Lists component: shows child records like Contributions.

• **Right Sidebar (if used)**:

– Activities / Related Actions: to log calls, create tasks, etc.

– A small Report Chart component (if you added a mini chart like “Recent Contributions”).

You can also use conditional visibility / dynamic components (if you implemented) — for example, show additional fields only when status = Ended.



6.4 Summary of Benefits

- Users easily navigate between Sponsors, Children, Contributions, etc.
- The app is purpose-built, reducing clutter.
- Lightning record pages are intuitive, with important information prioritized.
- Consistent layout across records increases efficiency and reduces user errors.
- UI enhancements complement the back-end logic and data model to create a polished CRM experience.

Phase 7: Integration & External Access

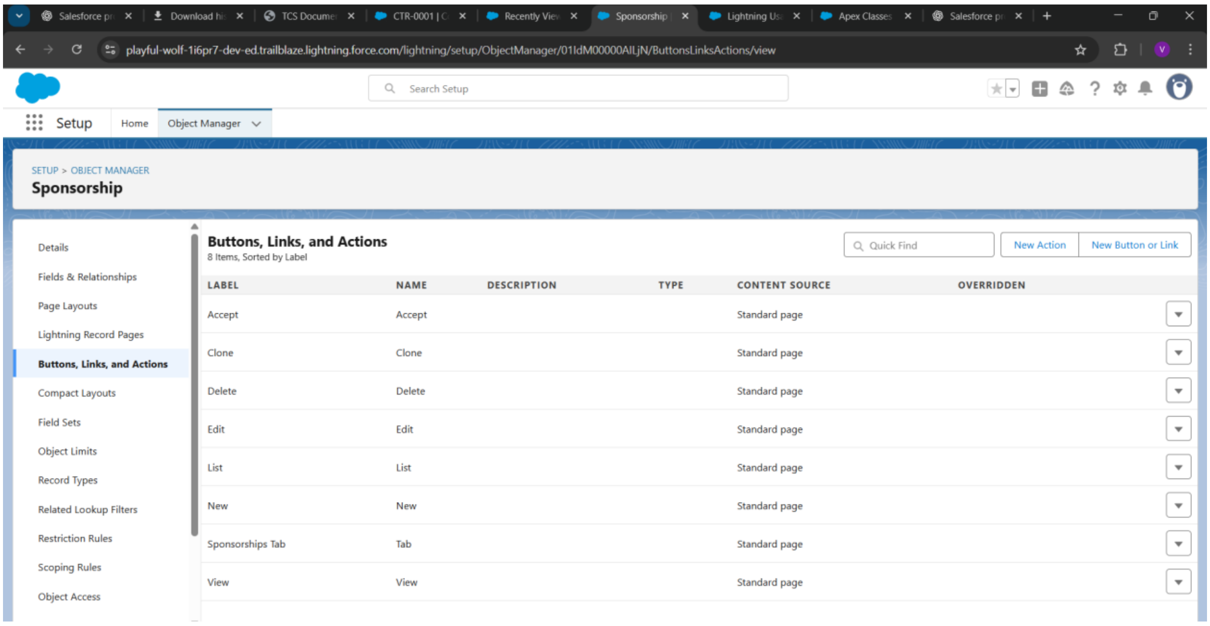
Objective

The objective of this phase was to enable basic and future-ready integration capabilities for the Child Sponsorship CRM to connect with external systems and services. Integrations enhance the CRM’s reach and interoperability, allowing nonprofit organizations to synchronize data, automate external workflows, and scale their sponsorship program.

7.1 Implemented Feature: Custom Link for External Sponsor Lookup

To demonstrate simple, user-initiated external integration, a custom button was added to the Sponsor record page.

- **Component:** Custom Button – Sponsor External Lookup
- **Purpose:** Provide nonprofit staff a quick way to perform external internet searches or verification on a sponsor’s name or details, streamlining sponsor background checks or information validation.
- **Implementation Details:**
    - **Object:** Sponsor__c
    - **Display Type:** Detail Page Button
    - **Behavior:** Opens link in a new browser window
    - **Content Source:** URL formula using a merge field
- **URL Formula Example:**
  https://www.google.com/search?q={!Sponsor__c.Name}+donor+profile
- This formula dynamically inserts the current Sponsor’s name into the URL, making the search context-aware.



7.2 Future Enhancements

These advanced patterns are planned to achieve robust enterprise-grade integration:

- **REST API Callouts to Donor Management Systems**
  - Use Apex HTTP callouts to send sponsorship and contribution data to external nonprofit donor platforms for synchronized reporting.
  - Named Credentials will secure the system authentication.
- **Platform Events for Real-Time Cross-System Notifications**
  - Publish events after key milestones, like sponsorship renewal or child progress report submission.
  - Subscribe external systems (financial, communication platforms) to respond asynchronously, enabling scalable decoupled workflows.
- **Salesforce Connect for External Data Access**
  - Integrate external child data or financial records directly into the CRM UI without storing duplicate data in Salesforce.
- **OAuth & Authentication Setup**
  - Configure security for external API access enabling seamless, secure integration.

Phase 8: Data Management & Deployment

Objective

The objective of this phase was to manage the application's business data and metadata efficiently, using Salesforce's standard data management tools alongside modern source control and deployment technologies to ensure data integrity and development best practices.

8.1 Data Management

Data Import Wizard

Purpose:
The Data Import Wizard allows non-technical users to upload and update records for standard and custom objects easily through a web interface, supporting up to 50,000 records per operation.

Implementation:

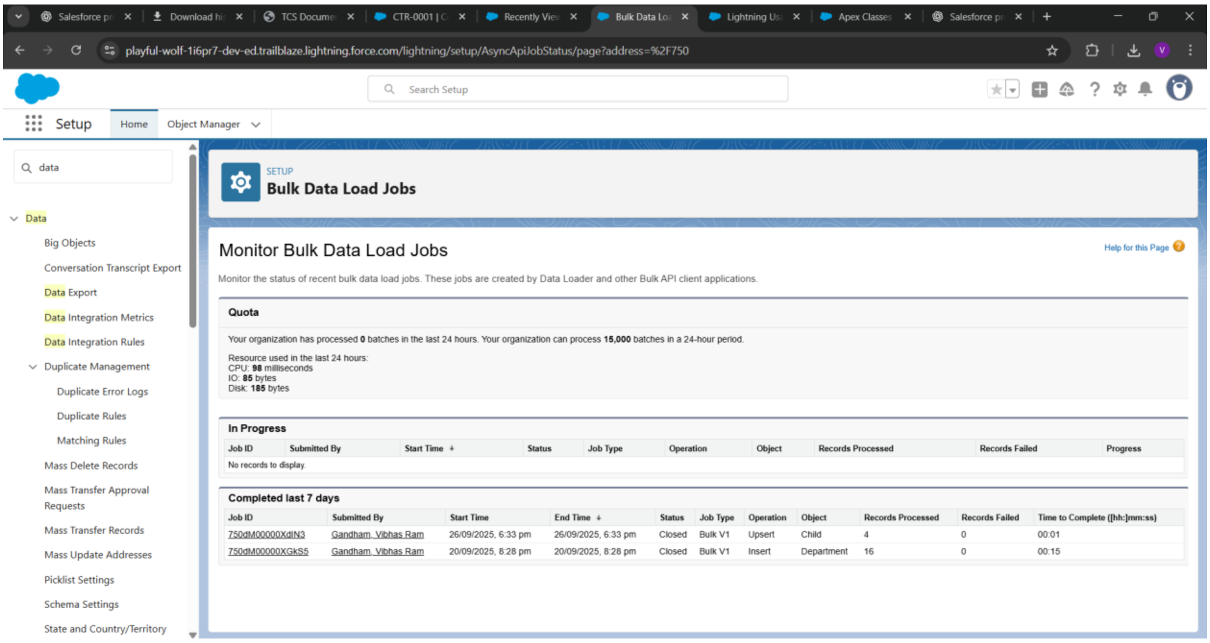
- Used to import bulk Sponsor, Child, and Contribution records from prepared CSV files.
- For example, a spreadsheet containing sponsor details and contributions is mapped to Salesforce fields via the wizard.
- The wizard handles field mapping, validation, and provides feedback on success or errors.

Data Loader

Purpose:
For scenarios requiring higher volume data loading, automated scheduled imports, or complex data actions, Data Loader offers command-line automation and supports millions of records.

Implementation:

- Utilized for initial data uploads involving complex Contribution transaction histories or for large batch data cleansing.
- Data Loader exported backup copies of existing data for offline storage.



8.2 Deployment & Metadata Management

Visual Studio Code & Salesforce CLI (SFDX)

Purpose:

VS Code paired with SFDX offers a modern development environment allowing source-driven metadata management, version control, and streamlined deployments.

Implementation:

- Configured the Child Sponsorship CRM as an SFDX project.
- A manifest file (package.xml) defined all custom objects, fields, Apex classes, and automation flows.
- Metadata was retrieved from Salesforce org into local files for development.
- Source maintained and versioned in GitHub repositories.
- Deployments to sandbox and production environments performed via Salesforce CLI commands.

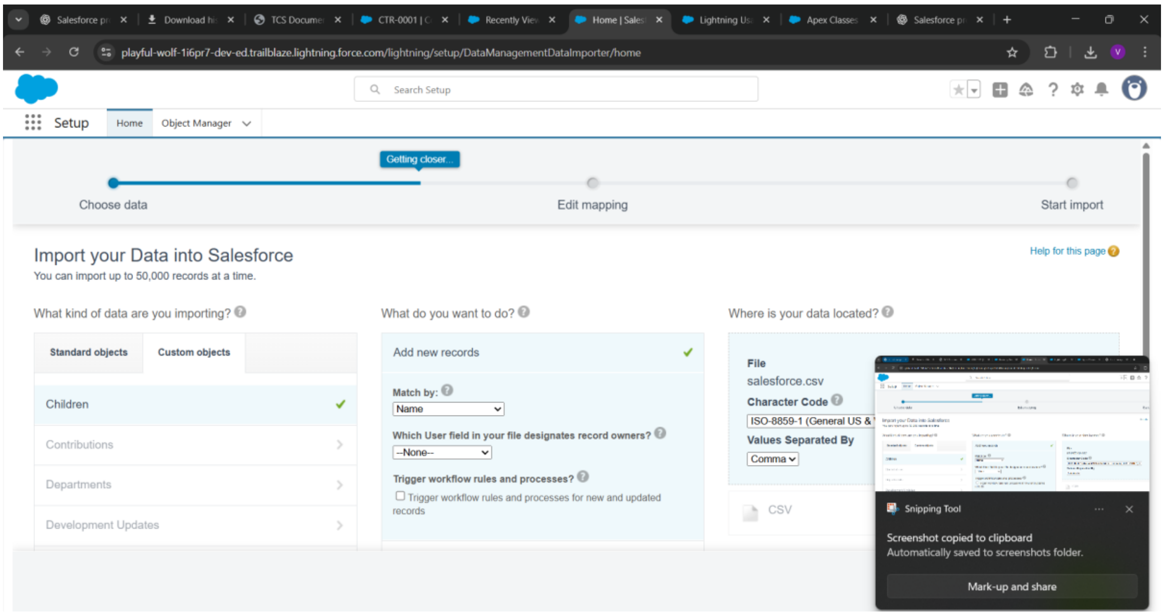
Change Sets

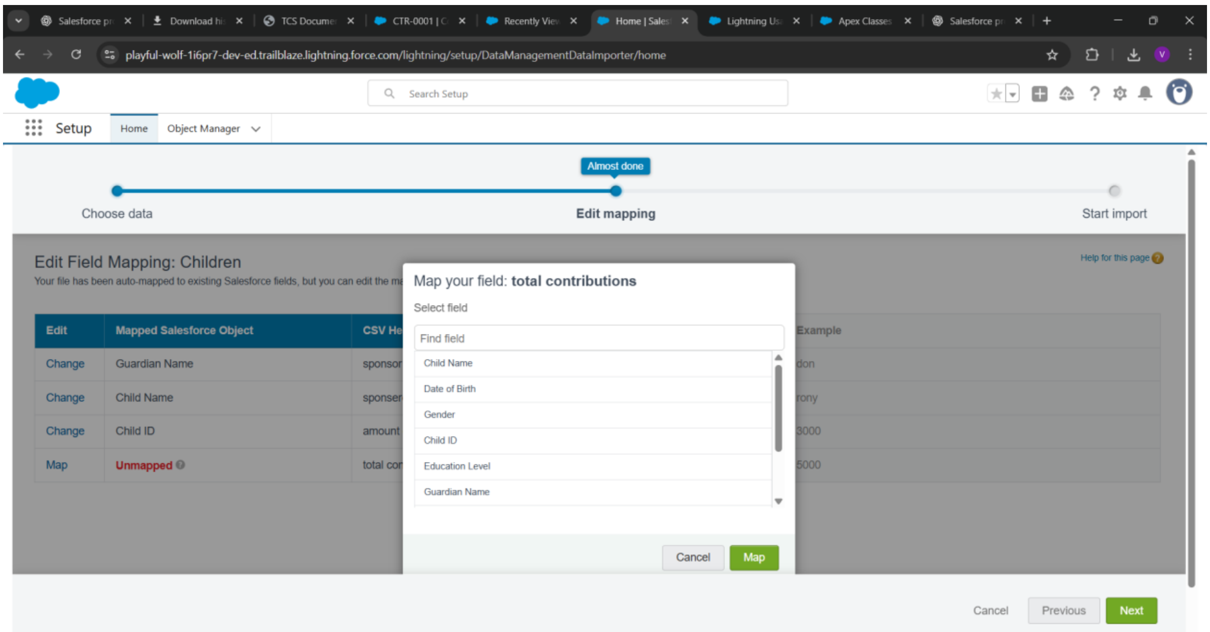
Purpose:

Change Sets provide a point-and-click method for deploying configuration and code between Salesforce orgs, commonly used for sandbox-to-production migration.

Implementation:

- Not used in this project as all development occurred within a single Developer Edition org.
- In a production setting, Change Sets would be the deployment mechanism for moving metadata to live orgs.





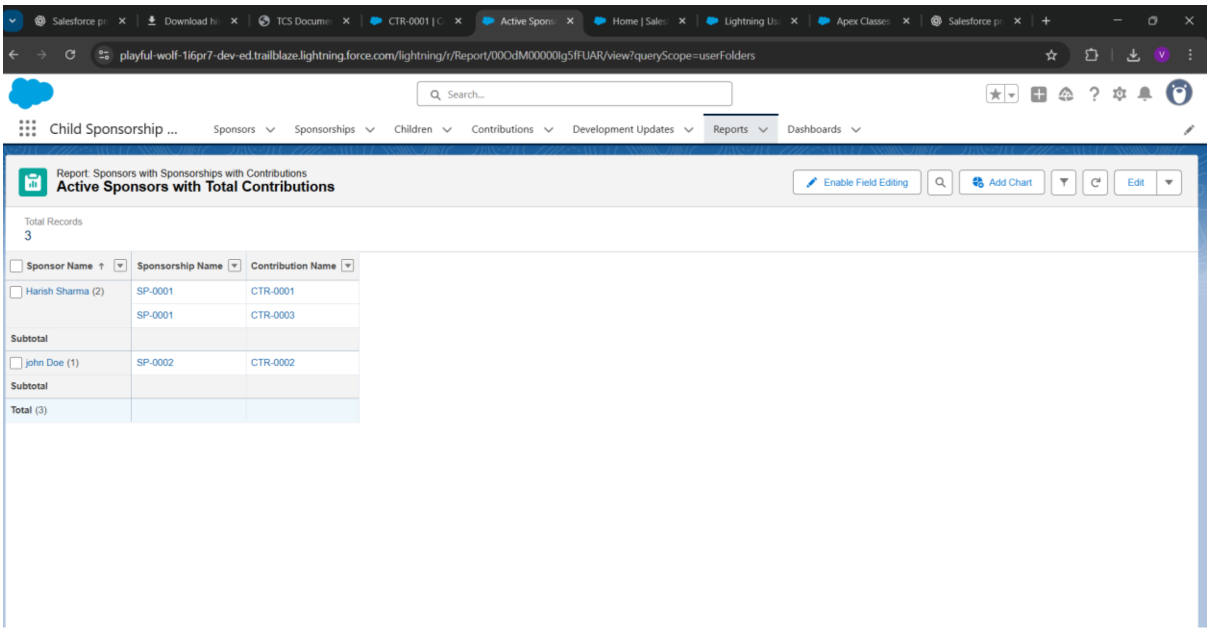
Phase 9: Reporting, Dashboards & Security Review

Objective

The objective of this phase was twofold: to convert raw CRM data into actionable insights through tailored reports and dashboards, and to perform a comprehensive security review ensuring the integrity, confidentiality, and appropriate accessibility of all system data.

9.1 Reports

Reports form the backbone of data analysis in Salesforce, providing data filtering, grouping, and summary features to derive meaningful insights. Key reports developed include:



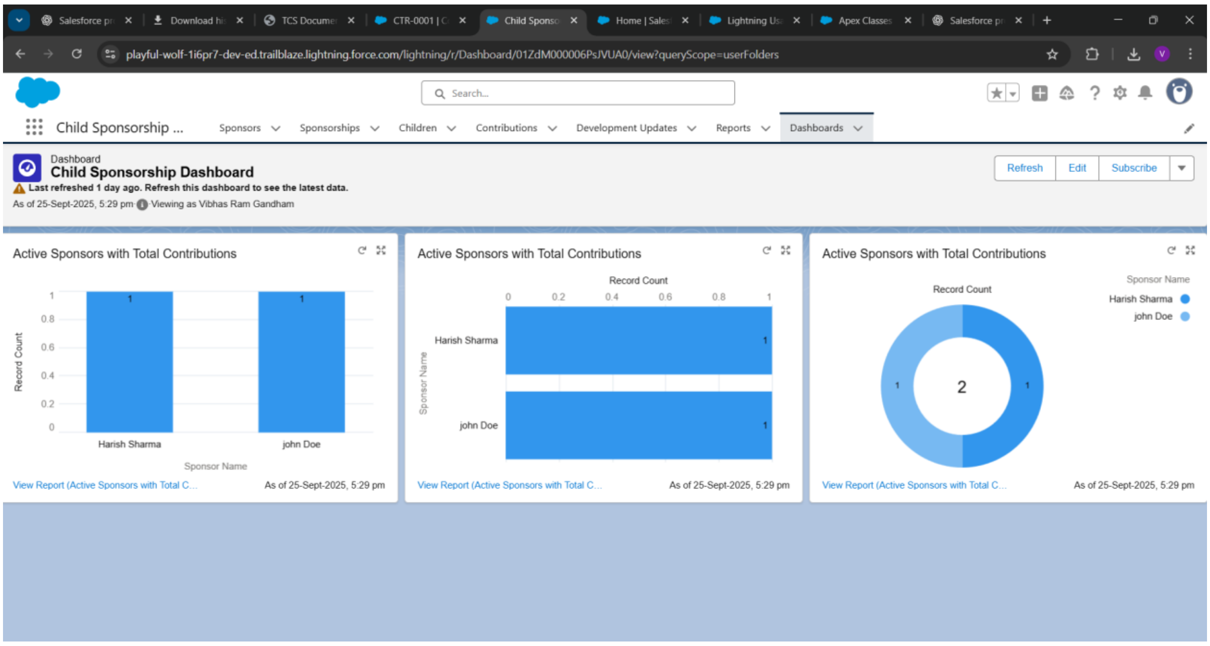
- **Report 1: Active Sponsorships by Sponsor**
    - **Purpose:** Summarizes the number of active sponsorships attributed to each sponsor.
    - **Type:** Summary report based on the Sponsorship object, grouped by Sponsor.

    - **Visualization:** Donut Chart highlighting proportions of sponsorship activity per sponsor.
- **Report 2: Contributions by Month**
    - **Purpose:** Tracks contributions received over time to identify donation trends and campaign effectiveness.
    - **Type:** Summary report on Contribution object, grouped by payment date month.
    - **Visualization:** Bar Chart illustrating monthly contribution totals.
- **Report 3: Child Development Updates Status**
    - **Purpose:** Monitors submission and review status of development updates for sponsored children.
    - **Type:** Summary report on Development Update object, grouped by status or update type.
    - **Visualization:** Bar Chart showing volume of updates per category.

9.2 Dashboards

A consolidated dashboard was created for nonprofit staff and management to monitor critical program metrics at a glance:

- **Dashboard: Child Sponsorship Overview**
    - Combines charts from the reports above for holistic visibility.
    - Includes charts for active sponsorship distributions, monthly contributions, and child development update statuses.
    - Configured as a dynamic dashboard to reflect data personalized to the logged-in user's role.



9.3 Security Review

Ensuring the security of sensitive donor and child data was critical. Security measures implemented:

- **Profiles:** Custom profiles created for Admin, NGO Staff, and Volunteers with tailored object and field permissions following the least privilege principle.
- **Roles:** Defined a role hierarchy reflecting the NGO structure to manage data visibility for reporting and administration.
- **Organization-Wide Defaults (OWD):** Set to Private for key objects like Sponsorship, Contributions, and Development Updates to restrict record access unless explicitly shared.
- **Production-Level Security:**
  - **Field-Level Security (FLS):** Sensitive fields (such as financial data) restricted from unauthorized profiles.
  - **Login IP Ranges:** Configured to restrict user login to trusted networks enforcing organizational policies.

- **Setup Audit Trail:** Enabled and monitored to track administrative changes supporting compliance and troubleshooting.

Phase 10: Final Presentation & Project Assets

Project Links & Showcase

- **Live Demo Video:** https://drive.google.com/file/d/1D-DlZphbnwDo7MJWbLNBBekQQHSPx9Mj/view?usp=drive_linkss
- **Source Code Repository:** https://github.com/VibhasRamG/Student-Enrollment-Management-System

Conclusion

The Child Sponsorship CRM project successfully delivered a comprehensive, scalable solution on the Salesforce platform addressing the core needs of nonprofits managing sponsor contributions and beneficiary progress. With a solid data model, integrated automation using Flows and Apex, and impactful reporting dashboards, this CRM streamlines operational transparency and donor engagement. The project provided valuable hands-on experience across Salesforce declarative and programmatic development, deployment strategies, and security best practices.

Acknowledgments

Sincere gratitude is extended to the mentor and the program organizing team for their invaluable guidance and support throughout this project journey.

G Vibhas Ram