**Smart Blood Donation & Donor Engagement System**

**Phase 1: Problem Understanding & Industry Analysis**

**Goal: Understand what we’re building and why**

**1. Requirement Gathering**

* **Core Need:** Blood banks and hospitals require a centralized system to manage donor records, blood inventory, and urgent requests.
* **Key Requirements:**
  + Maintain donor profiles (blood group, last donation date, eligibility).
  + Automate donation reminders and alerts for eligible donors.
  + Real-time matching of urgent hospital requests with available donors.
  + Track donation history and generate reports.
  + Ensure compliance with medical regulations and privacy standards.

**2. Stakeholder Analysis**

* **Blood Banks & Hospitals:** Primary users, need efficient donor management and quick access to blood in emergencies.
* **Donors:** End users who provide blood; require transparency, reminders, and recognition for their contributions.
* **Volunteers/NGOs:** Help coordinate donation drives and engage communities.
* **Government/Regulators:** Require reports to ensure compliance and transparency in blood donation.
* **System Administrators:** Maintain and manage CRM operations.

**3. Business Process Mapping**

**Current Process (Manual):**

* Hospitals manually contact donors during shortages.
* Donor records are often fragmented across spreadsheets or local databases.
* Delays in matching donors → shortage in critical situations.

**Proposed CRM-enabled Process:**

1. Donor registers in CRM (via hospital/NGO portal).
2. System validates donor eligibility (based on health and donation frequency).
3. CRM sends automated reminders when donor becomes eligible again.
4. Hospitals raise urgent blood requests → CRM matches with eligible nearby donors.
5. Dashboards provide real-time visibility of donor activity, requests, and donation drives.

**4. Industry-specific Use Case Analysis**

* **Healthcare Industry Need:** Reliable, transparent, and quick blood donation systems are critical to saving lives.
* **Challenges Today:** Lack of transparency, fragmented data, emergency delays.
* **Value of CRM in Healthcare Blood Donation:**
  + Centralizes donor data for quick access.
  + Improves trust among donors by ensuring proper tracking and communication.
  + Provides hospitals with efficiency and reduced response time.
  + Enhances community engagement through transparent reporting.

**5. AppExchange Exploration**

On Salesforce **AppExchange**, similar healthcare apps exist (for hospital management, patient tracking, and volunteer engagement). However, very few apps focus specifically on **blood donation management**.

* **Relevant Solutions on AppExchange:**
  + *Volunteers for Salesforce* (for volunteer scheduling).
  + *Health Cloud Apps* (for patient and healthcare data management).
* **Gap Identified:** No focused app that combines **blood donor engagement + hospital urgent request handling + automated reminders**.
* **Opportunity:** Build a unique solution that fills this gap and can later be extended or even published on AppExchange as a custom product.

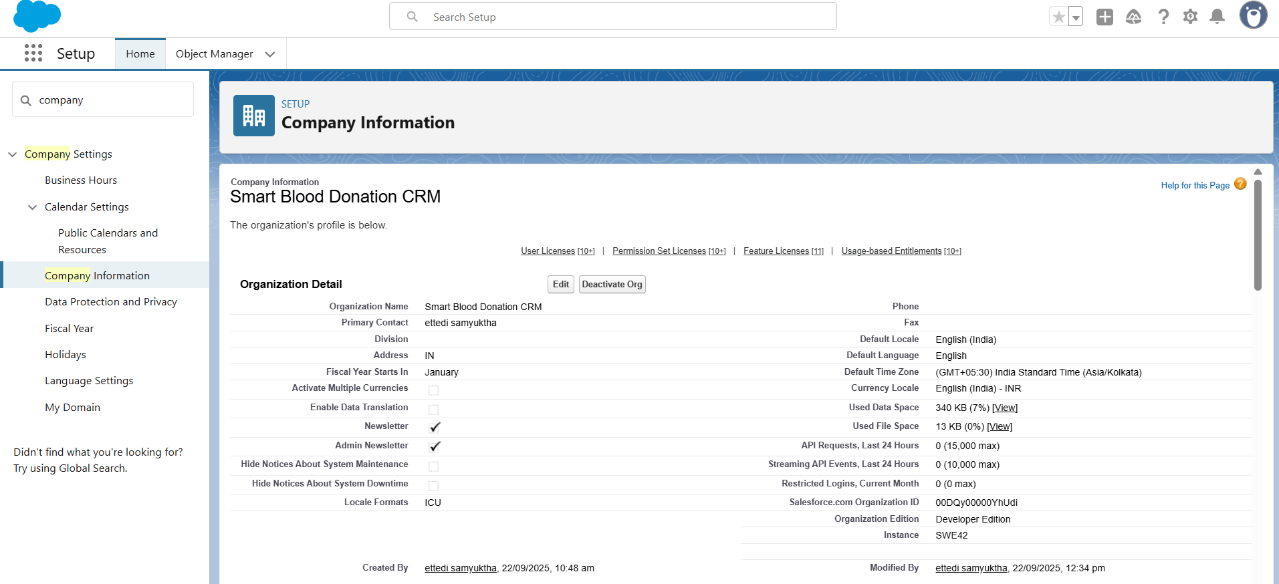
PHASE 2 - Org Setup & Configuration

Company Profile Setup

Path: Setup → Company Information → Edit

Name: Smart Blood Donation CRM  
Time Zone: GMT+05:30 Asia/Kolkata  
Locale: English (India)  
Language: English  
Currency: INR

Purpose: Establishes the foundational organizational settings for donor, hospital, and NGO data.

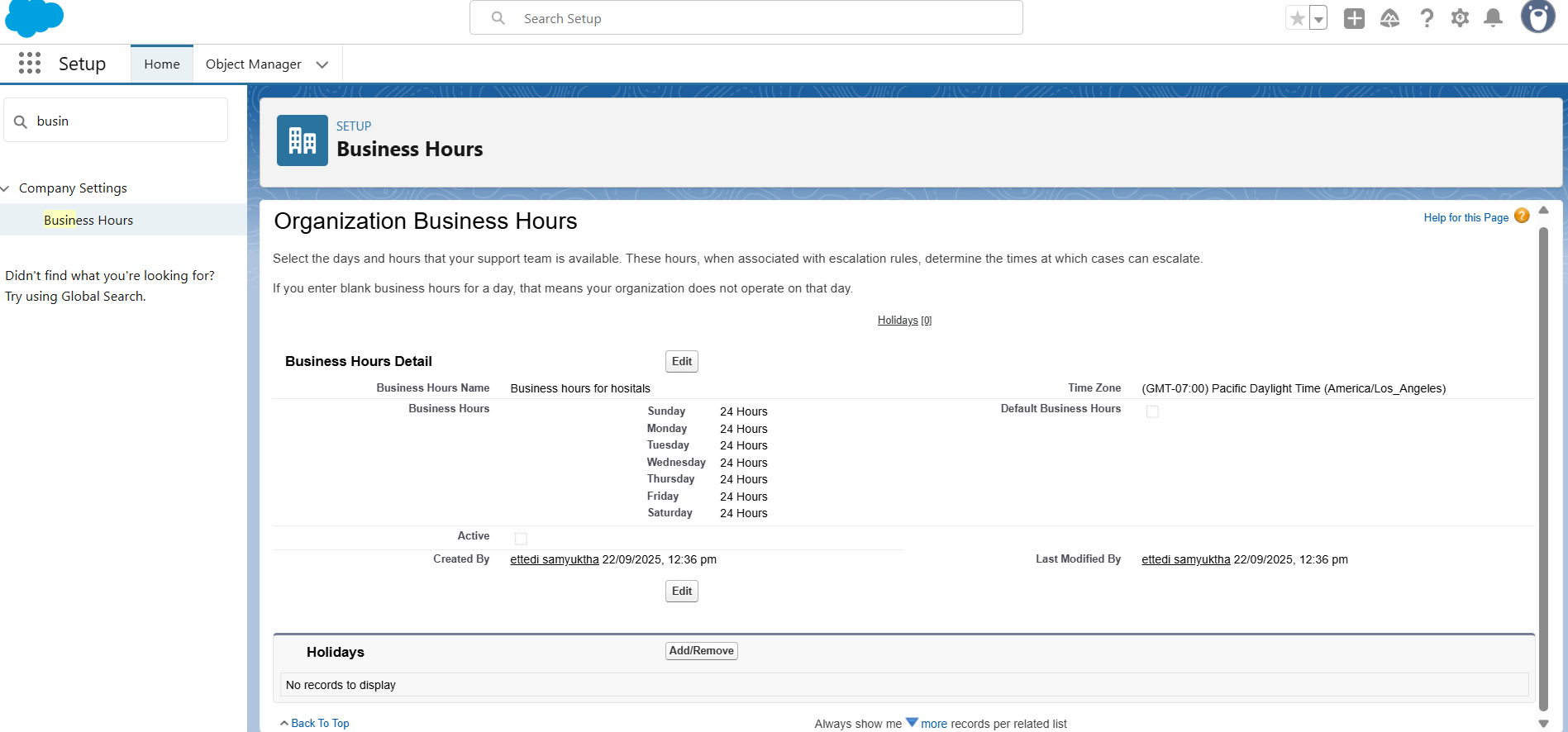


Business Hours Setup

Path: Setup → Business Hours → New

Name: 24x7 Hospital Emergency Hours  
Time Zone: GMT+05:30 Asia/Kolkata  
Working Hours: Mon-Sun 12:00 AM–11:59 PM

Purpose: Hospitals and blood banks operate round-the-clock; emergency requests need 24x7 coverage.

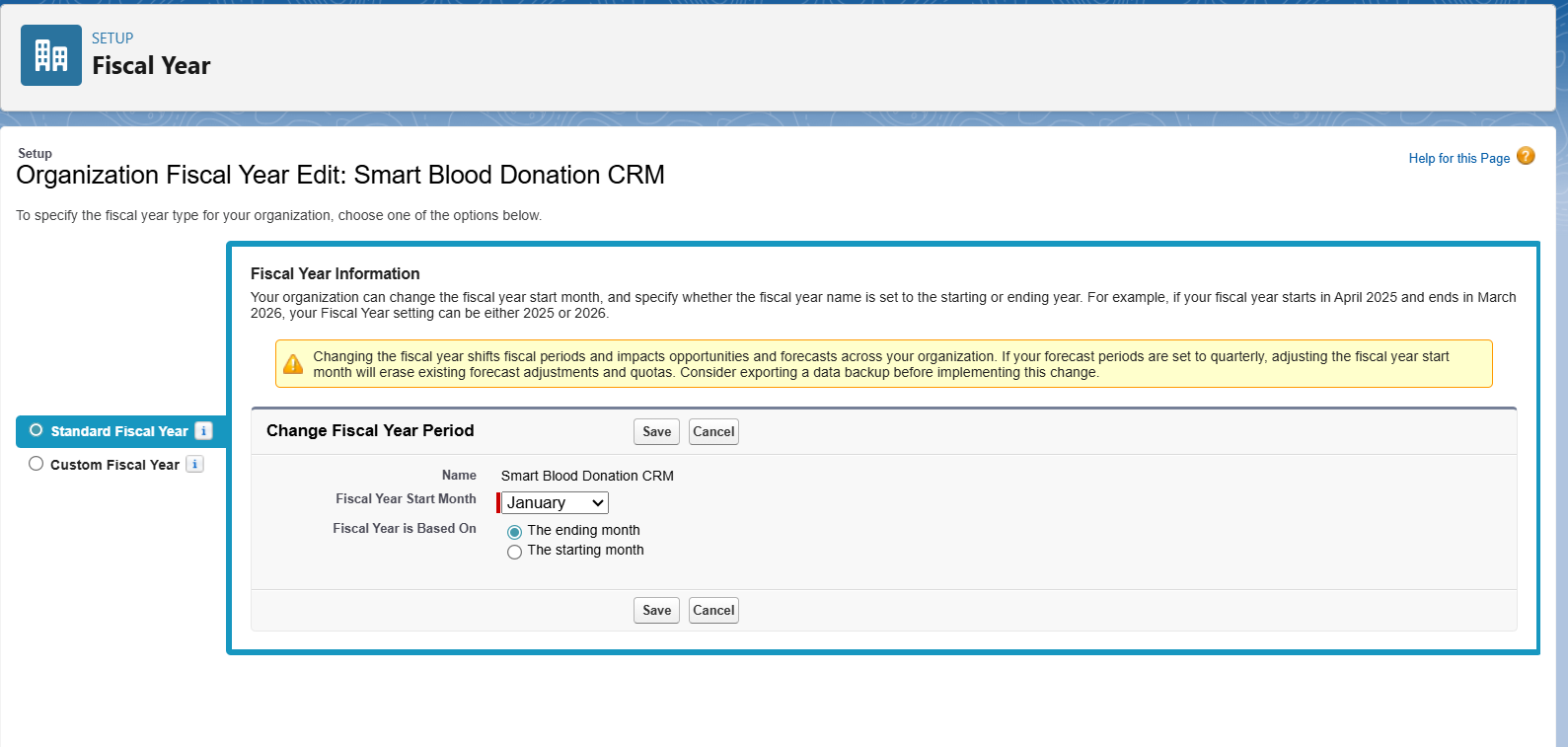


Fiscal Year Setup

Path: Setup → Fiscal Year

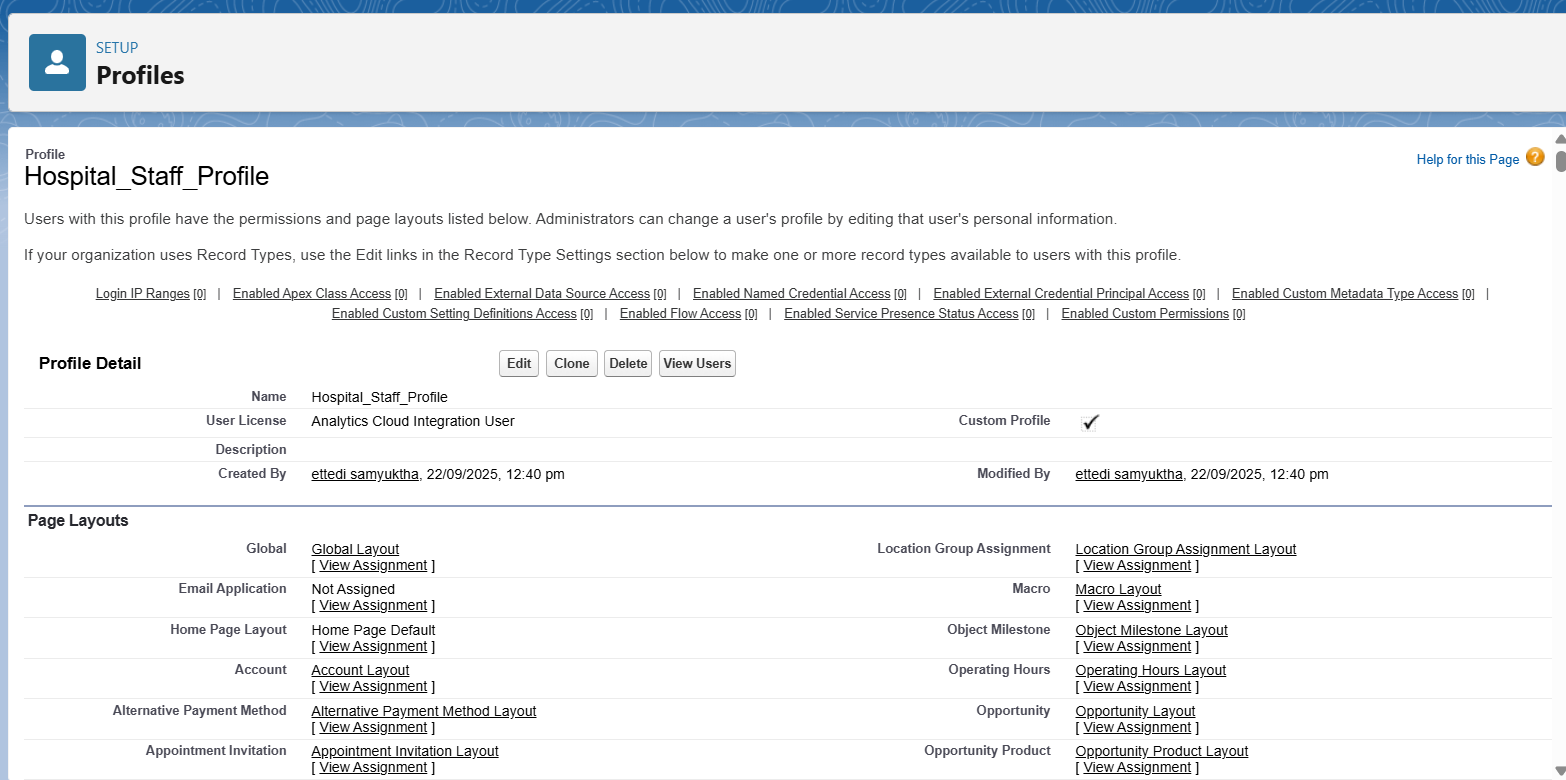
Type: Standard Fiscal Year  
Configuration: Starting month set to January

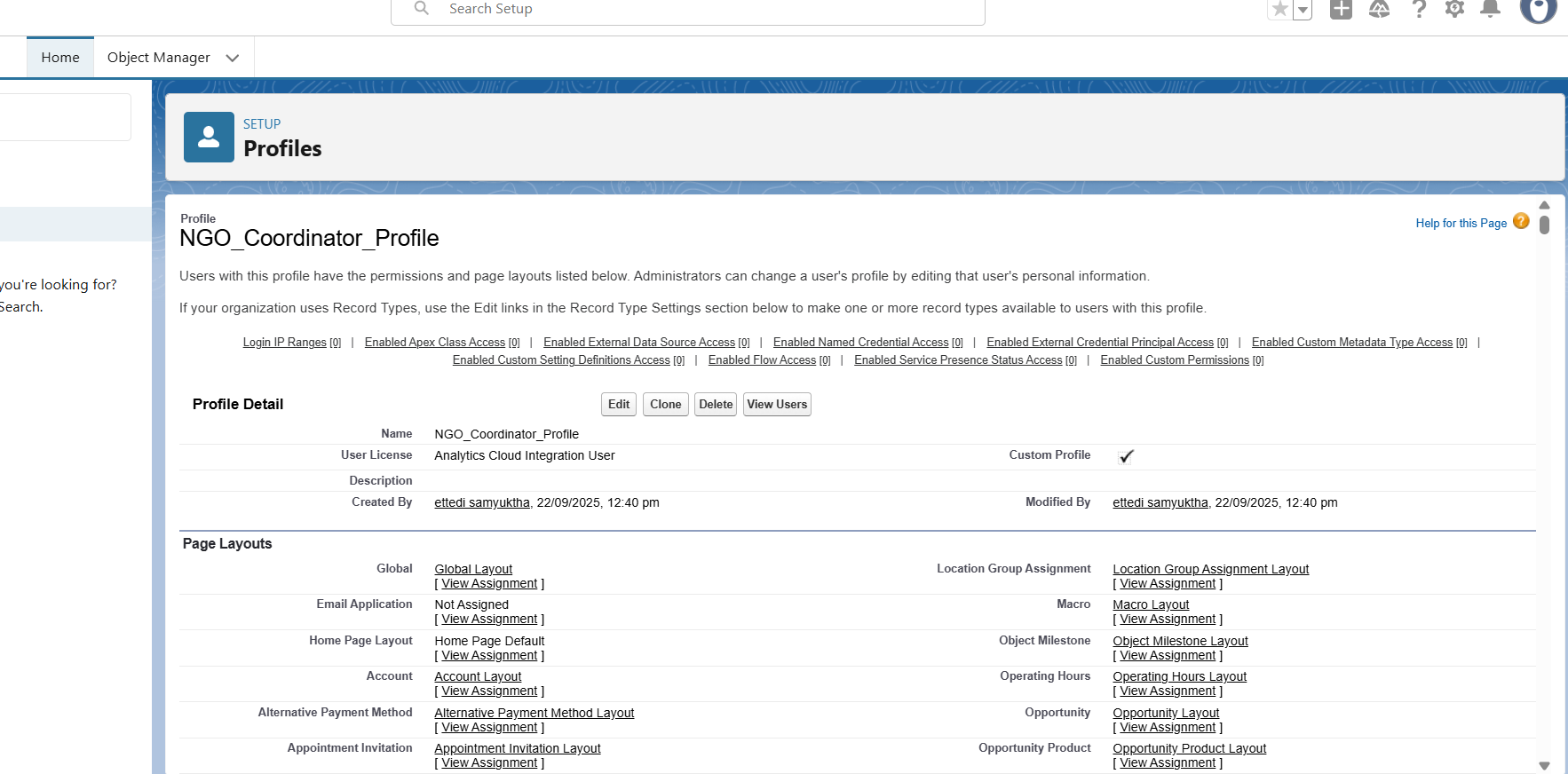
Purpose: Defines reporting periods for donor drives and hospital blood usage analysis.

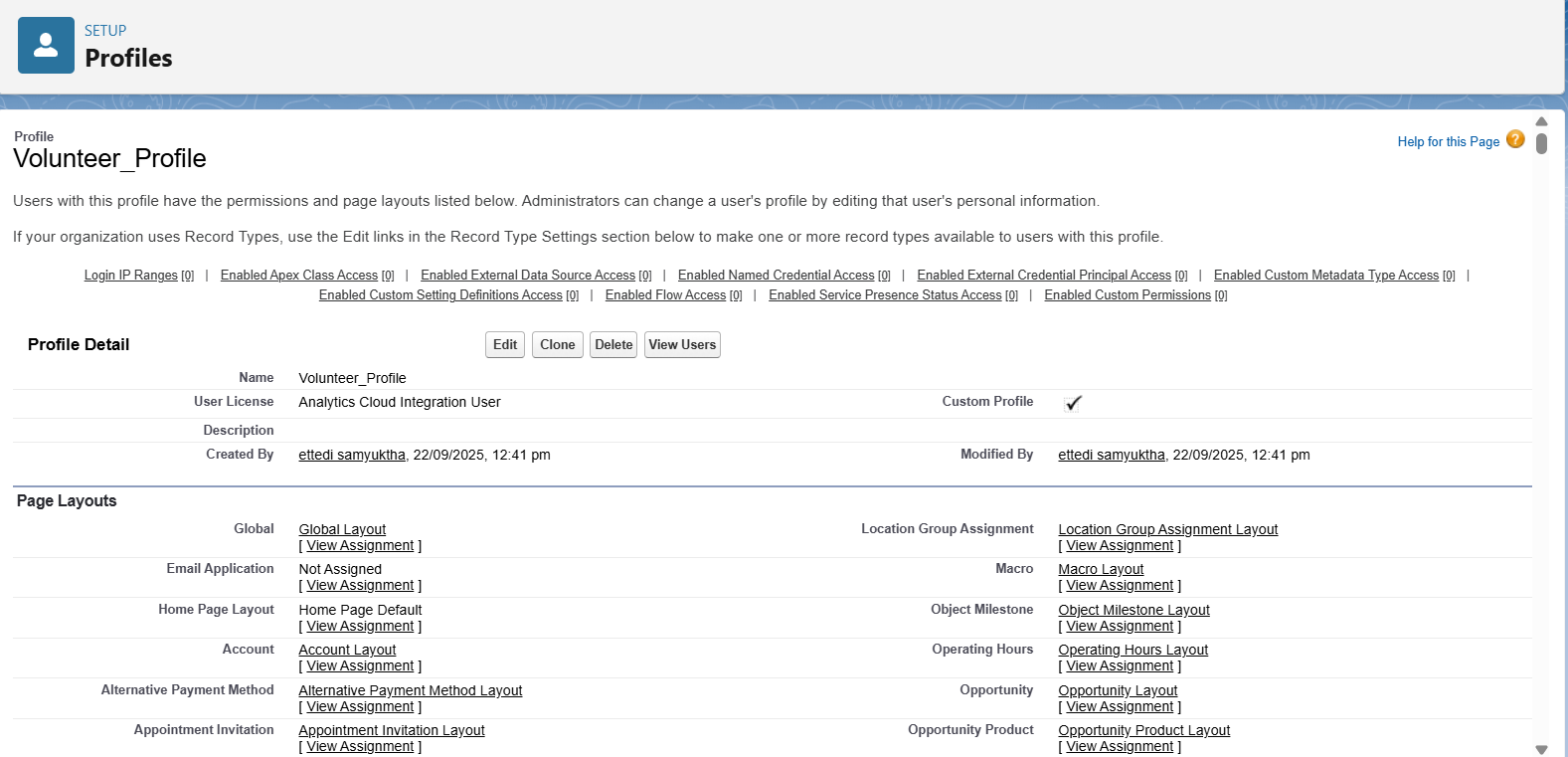


User Setup (Profiles, Roles, Permission Sets, Users)

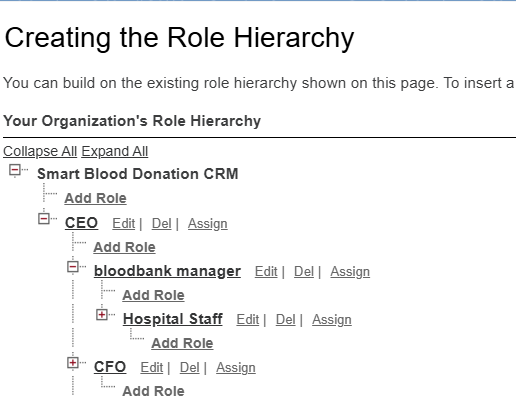
Profiles:  
- Hospital\_Staff\_Profile: For managing donor records and hospital requests.

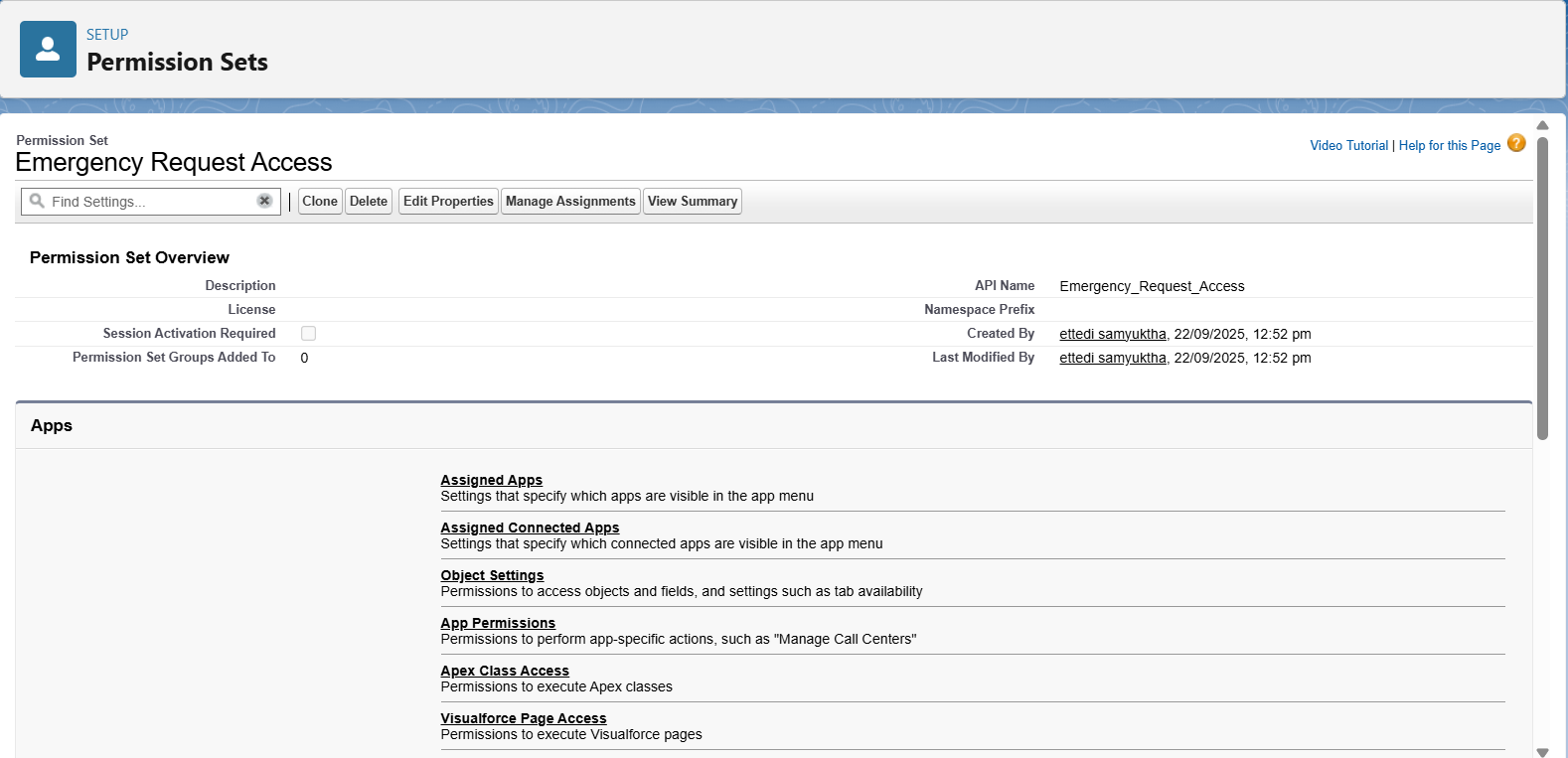
  
- NGO\_Coordinator\_Profile: For managing donation drives and volunteer data.

  
- Volunteer\_Profile: For limited access to participation in donation events.



Roles:  
- CEO → Blood Bank Manager → Hospital Staff → Volunteers



Permission Sets:  
- Emergency\_Request\_Access\_PS: Grants ability to handle urgent hospital requests.  


Users:  
- staff1 → Hospital\_Staff\_Profile, Role: Hospital Staff  
- ngo1 → NGO\_Coordinator\_Profile, Role: NGO Coordinator  
- volunteer1 → Volunteer\_Profile, Role: Volunteer

Role Hierarchy Setup

Path: Setup → Roles → Set Up Roles

Top-Level Role: CEO  
Child Role: Blood Bank Manager  
Child Role: Hospital Staff  
Child Role: Volunteer

Resulting Hierarchy: CEO → Blood Bank Manager → Hospital Staff → Volunteer

**OWD & Sharing Rules**

* **OWD (Org-Wide Default)**: This will be configured in Phase 3, as the settings need to be applied to the custom objects that will be created then.
* **Sharing Rule**: This will also be implemented in Phase 3 after the custom objects and OWD settings are in place.

**Dev Org Setup**

The development environment was prepared to support the project implementation.

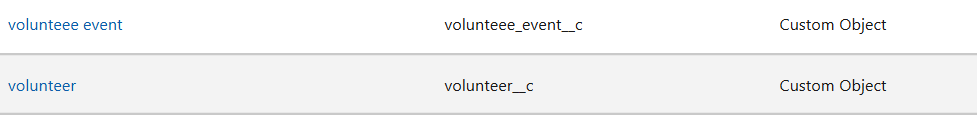
* A Salesforce **Developer Edition org** was set up to build the project.
* A **GitHub Repository** was created for version control of the source code.
* **VS Code and SFDX** were configured for developing the Apex backend and future Lightning Web Components.

**Phase 3 – Data Modeling & Relationships**

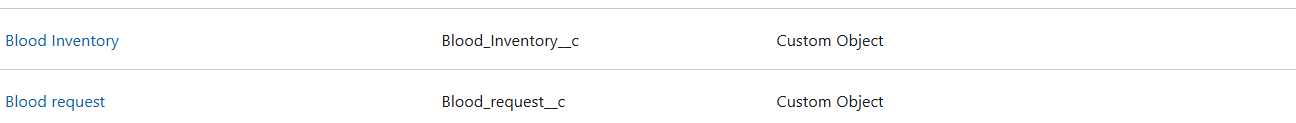
**👉 Goal: Build data structure**

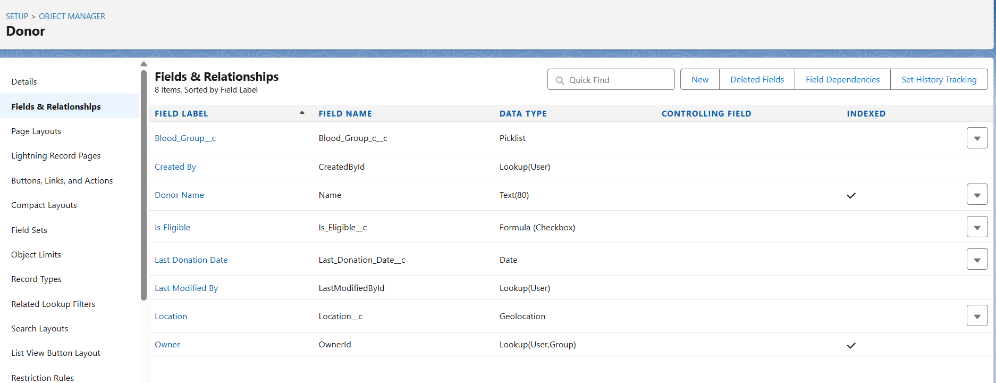
**1. Standard & Custom Objects**

* **Standard Objects:**
  + *User* (for staff, NGO coordinators, volunteers)
  + *Campaign* (for blood donation drives)
  + *Contact* (donors and recipients)
  + *Account* (hospitals, NGOs, organizations)
* **Custom Objects:**
  + *Donor* (blood group, eligibility, donation history)
  + *Blood\_Request* (urgent hospital request, request date, priority)
  + *Volunteer* (NGO/individual volunteers for campaigns)
  + *Donation\_Event* (blood drives and camps, linked to campaigns)

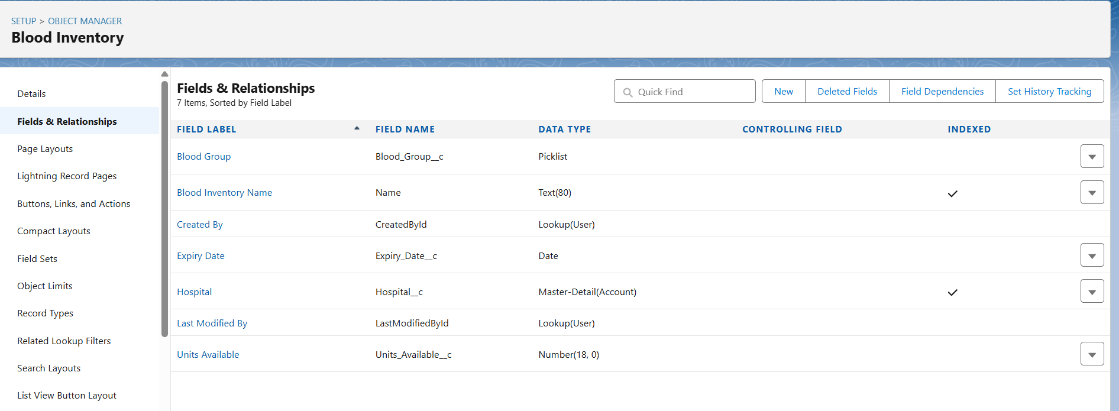
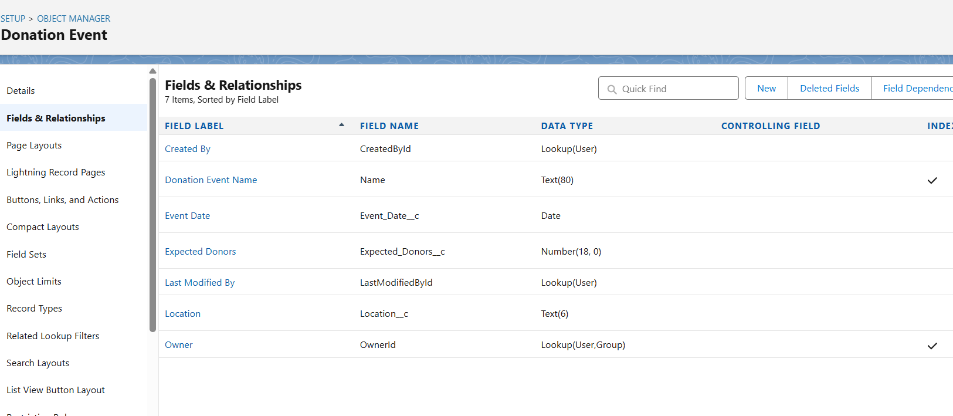


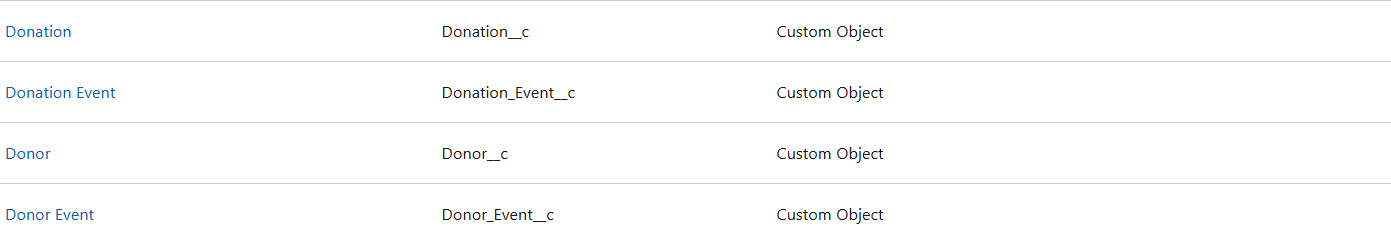
* + *Blood\_Inventory* (available blood units, expiry dates)





**2. Fields**

* **Donor:** Blood Group (Picklist), Last Donation Date (Date), Eligibility Status (Formula), Location (Geolocation).
* **Blood\_Request:** Request Type (Emergency/Planned), Quantity Required (Number), Request Status (Picklist).
* 
* 
* **Volunteer:** Skills (Text), Availability (Date/Time), Hours Contributed (Number).
* **Donation\_Event:** Event Location (Text), Date (Date), Expected Donors (Number).
* **Blood\_Inventory:** Blood Group (Picklist), Units Available (Number), Expiry Date (Date).



**3. Record Types**

* *Donor Record Types:* Regular Donor, Rare Donor (for rare blood groups).
* *Blood\_Request Record Types:* Emergency Request, Planned Request.
* *Volunteer Record Types:* NGO Volunteer, Independent Volunteer.

**4. Page Layouts**

* Customize layouts for usability:
  + Donor Page → Show donation history, eligibility, and contact info.
  + Blood Request Page → Show request status, hospital details, and donor matches.
  + Volunteer Page → Show event participation and contribution hours.

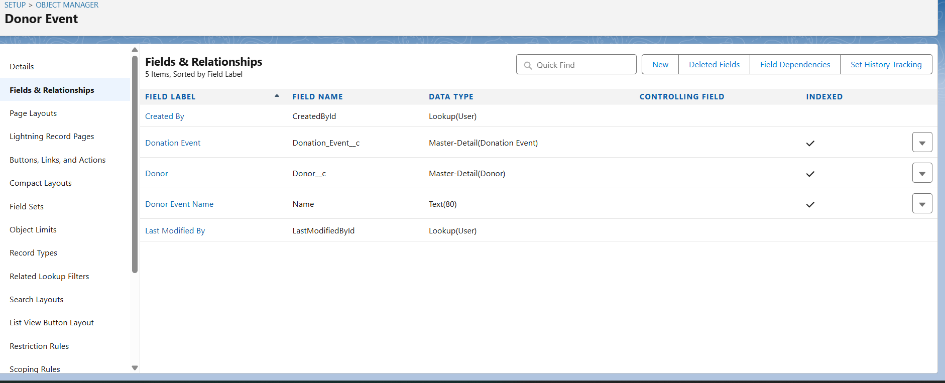
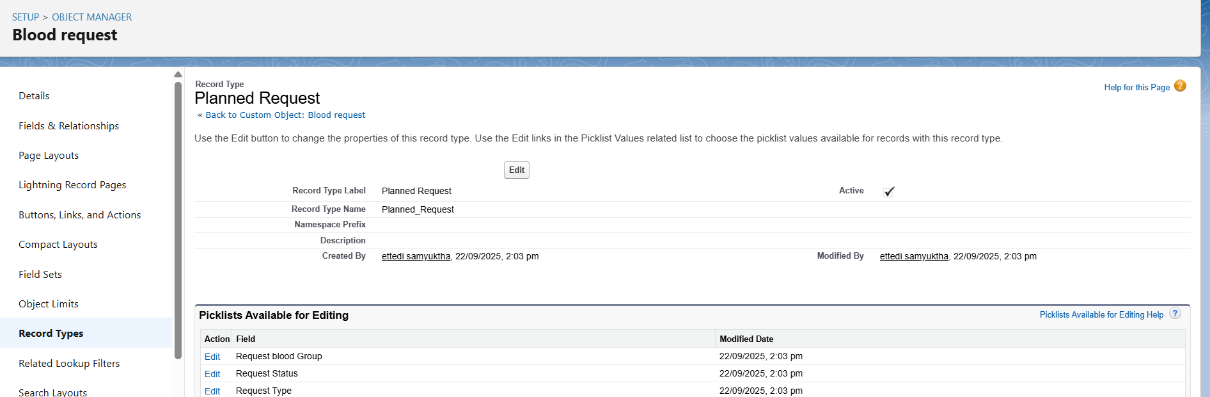
**5. Compact Layouts**

* **Donor Compact Layout:** Blood Group, Last Donation Date, Eligibility.
* **Blood Request Compact Layout:** Request Type, Quantity, Status.
* **Volunteer Compact Layout:** Name, Availability, Skills.

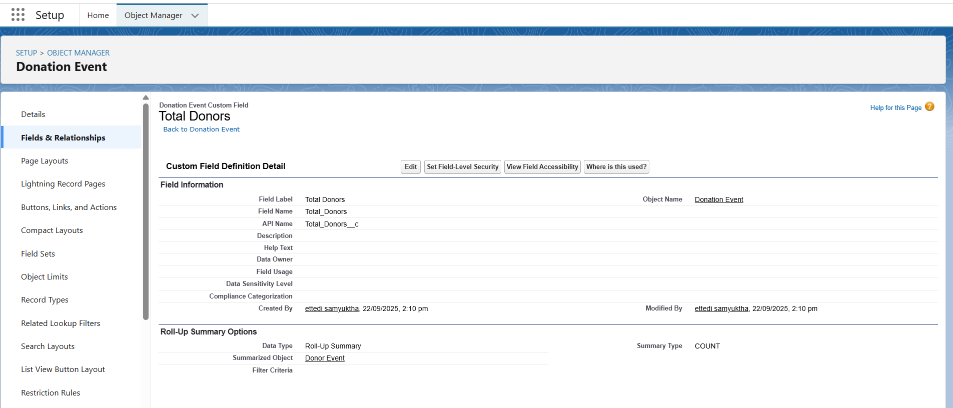
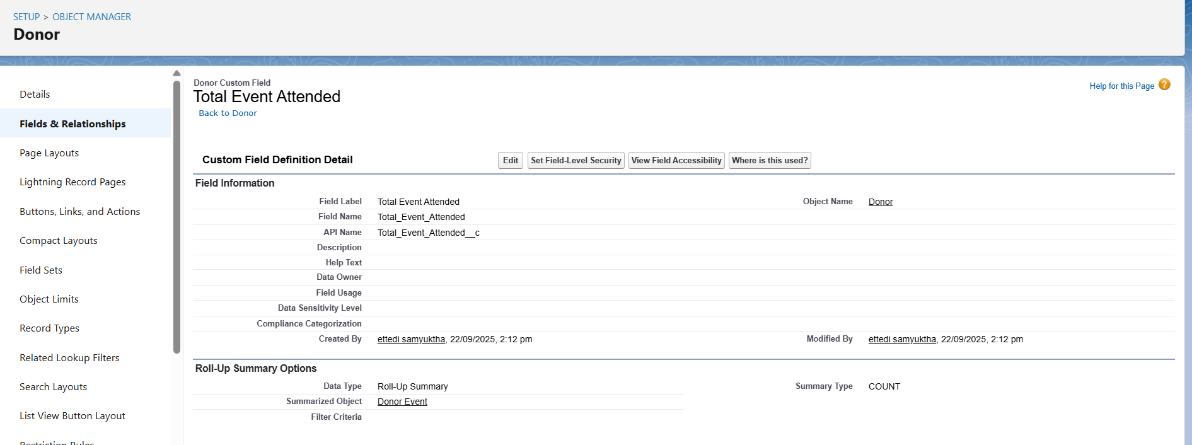
**6. Schema Builder**

* Use **Schema Builder** to visualize object relationships:
  + Donor → Donation\_Event (Many-to-Many via Junction Object).
  + Blood\_Request → Donor (Lookup).
  + Volunteer → Donation\_Event (Junction Object).
  + Blood\_Inventory → Hospital (Master-Detail).

**7. Lookup vs Master-Detail vs Hierarchical**

* **Lookup:** Donor ↔ Blood\_Request (optional relationship, donor may or may not respond).
* 
* **Master-Detail:** Hospital → Blood\_Inventory (ownership & roll-up summaries).
* 
* **Hierarchical:** Used in *User* object (e.g., Staff reporting to Manager).

**8. Junction Objects**

* **Donor\_Event\_\_c:** Junction between Donor and Donation\_Event.
* **Volunteer\_Event\_\_c:** Junction between Volunteer and Donation\_Event.
* 
* Purpose: Allow many-to-many relationships for tracking multiple donors/volunteers in multiple events.
* 

**9. External Objects**

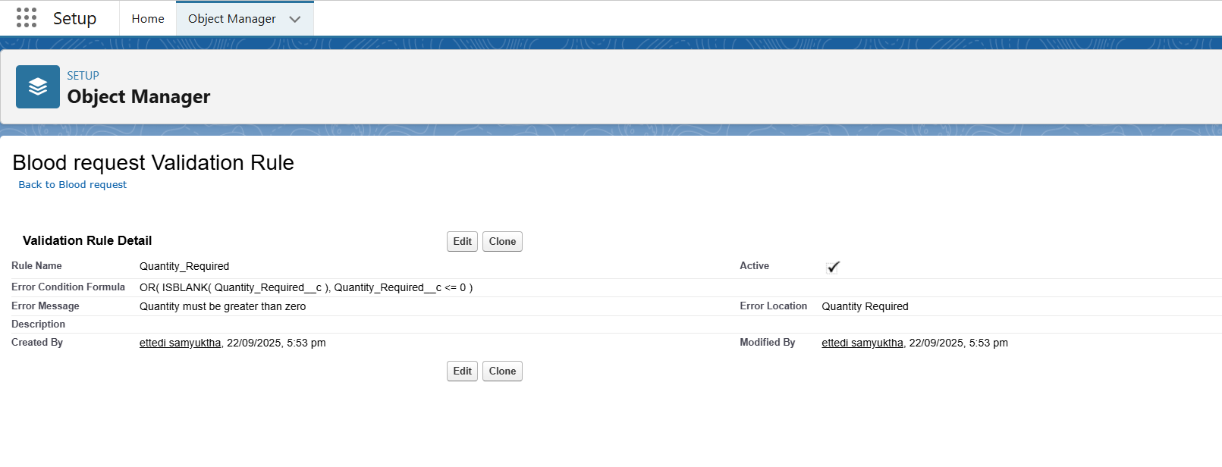
* Integrate with **external hospital systems** or **government blood bank APIs** for:
  + Real-time blood inventory sync.
  + Automatic update of donor eligibility from medical labs.

**Phase 4 – Process Automation (Admin)**

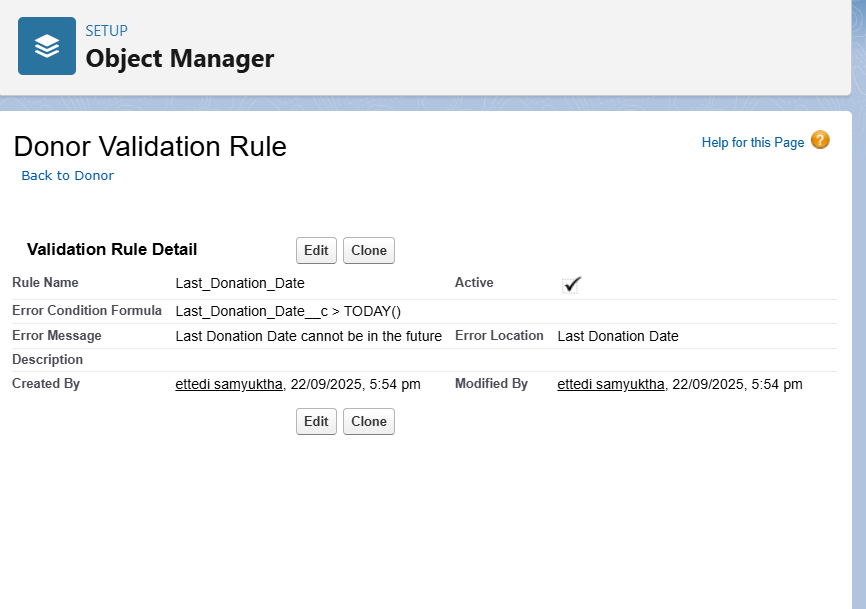
**👉 Goal: Automate tasks**

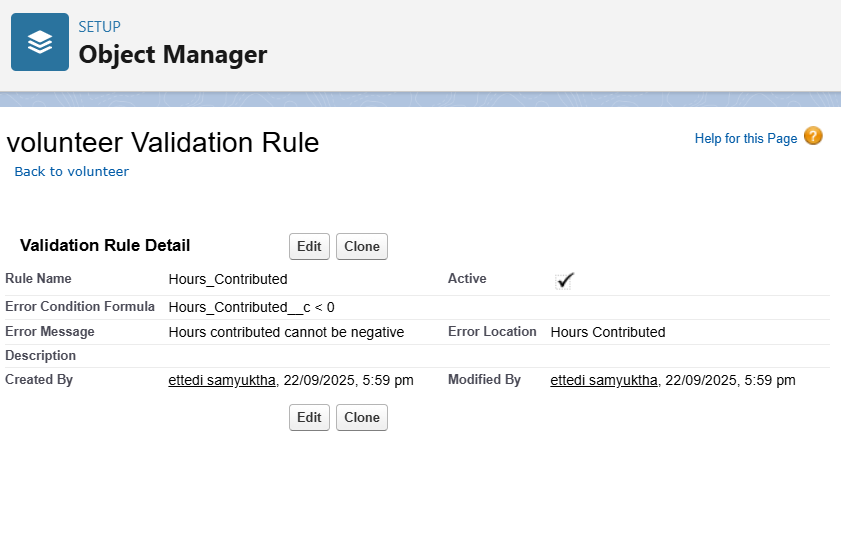
**1. Validation Rules**

* Ensure **data quality and accuracy** before saving records.
* Examples:
  + On Blood\_Request\_\_c: Prevent requests with Quantity\_Required\_\_c <= 0.



* + On Donor\_\_c: Ensure Last\_Donation\_Date\_\_c cannot be a future date.



* + On Volunteer\_\_c: Ensure Hours\_Contributed\_\_c ≥ 0. 

**2. Workflow Rules (Legacy)**

* Though Salesforce recommends **Flows** now, Workflow Rules can still handle simple tasks.
* Example:
* Record-Triggered Flow: Donor Matching for Emergency Requests

**3. Process Builder (Legacy)**

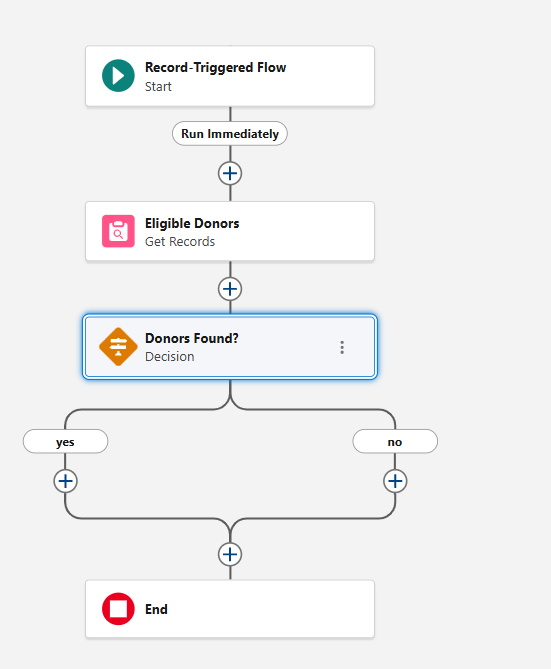
* Used for **if-then automation logic** (also being replaced by Flows).
* Example:
  + When a new **Donation\_Event\_\_c** is created, automatically create related **Tasks** for staff to prepare venue and equipment.

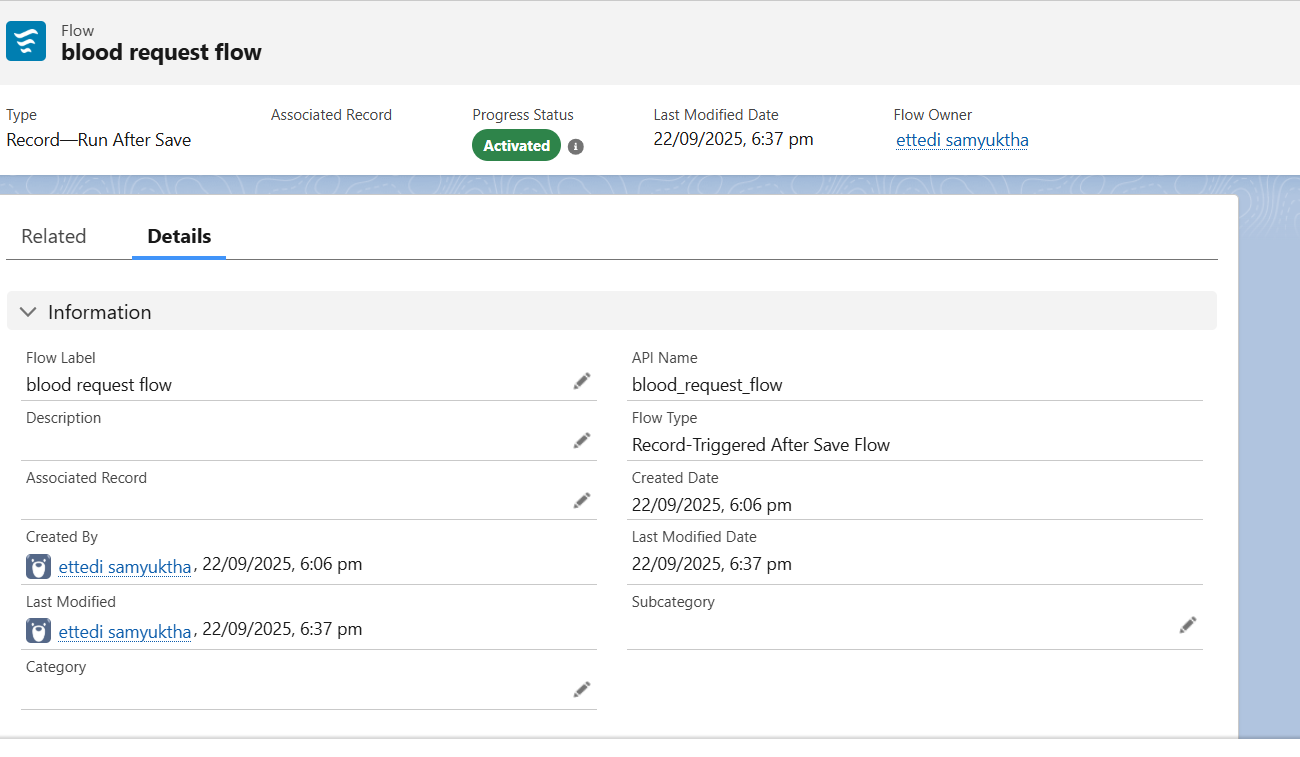
**4. Approval Process**

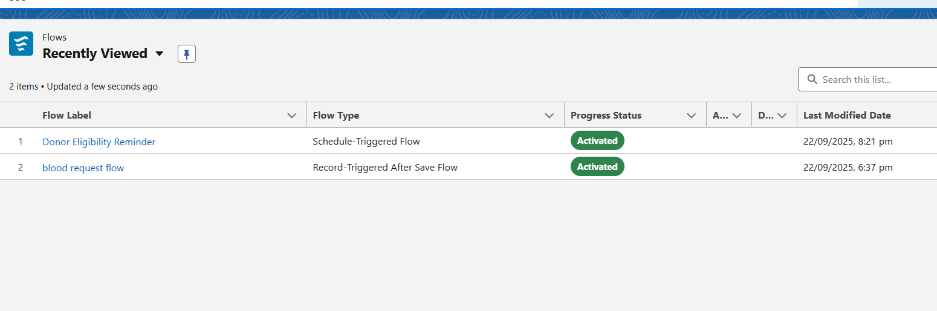
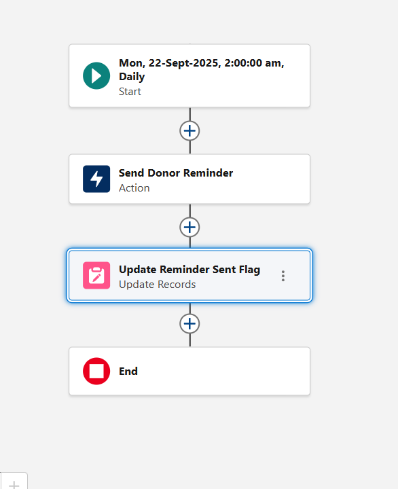
* Manage **multi-step approvals**.
* Example:
  + Large **fund donation requests** by NGOs require approval from CSR Manager → Director.
  + Steps: Submit → Pending → Approved/Rejected.

**5. Flow Builder (Recommended)**

* Main tool for automation going forward.
* Examples for Blood Donation CRM:
  + **Record-Triggered Flow:** When a new **Blood\_Request\_\_c** is logged, auto-match with eligible donors (Donor\_\_c.Is\_Eligible = TRUE AND Blood\_Group = Request\_Blood\_Group).



* + **Scheduled Flow:** Send reminders to donors whose eligibility date has just become valid (90+ days since last donation). Run daily at 2 AM → find donors who are eligible to donate again after 90+ days → send reminder → mark as “reminded”.
  + 



**6. Email Alerts**

* Automatically send personalized emails.
* Examples:
  + **Donors** → Thank-you email after successful donation.
  + **Volunteers** → Confirmation email after event sign-up.
  + **Hospitals** → Notification when requests are fulfilled.

**7. Field Updates**

* Automatically update fields based on conditions.
* Examples:
  + When Blood\_Request\_\_c.Status = Fulfilled, update Fulfilled\_Date\_\_c = TODAY().
  + When Donor\_\_c makes a donation, update Last\_Donation\_Date\_\_c.

**9. Custom Notifications**

* Push notifications to Salesforce users (in web/mobile).
* Examples:
  + Notify hospital staff immediately when an **Emergency Request** is created.
  + Notify NGO coordinators when **volunteer capacity is below threshold** for an event.