**🩸 Project Title: Community Blood Donation & Donor Tracker**

* **Industry**: Healthcare / Social Welfare
* **Project Type**: Salesforce CRM Implementation (Community Service + Healthcare)
* **Target Users**: Donors, Hospitals, Blood Bank Administrators, Volunteers

**📝 Problem Statement**

Blood shortages are a recurring challenge in hospitals and blood banks. Manual tracking of donors, last donation dates, and hospital requests leads to delays and inefficiency. Often, hospitals cannot quickly find eligible donors, and donors do not receive timely reminders to donate blood again.

To solve this, we propose a **Salesforce-based Blood Donation & Donor Tracker System** that will:

* Centralize donor and hospital data.
* Automatically match donors to hospital requests based on eligibility.
* Send automated notifications to donors and hospitals.
* Provide dashboards to track blood availability, requests, and donation camps.

**✅ Use Cases**

1. **Donor Management**

* **Capture donor details:** name, age, blood group, contact info, location, last donation date.
* Mark donor as *Eligible/Not Eligible* based on last donation date (≥ 90 days).

1. **Hospital & Blood Request Management**

* Hospitals create blood requests specifying group & units required.
* System auto-matches eligible donors and notifies them.
* Track status: *Pending, In Progress, Fulfilled*.

1. **Donation Camp Management**

* Volunteers schedule and manage donation camps.
* Track participating donors and collected units.

1. **Notifications & Alerts**

* Automated reminders to donors when eligible to donate again.
* Email/SMS to donors when hospital requests match their blood group.

1. **Reporting & Dashboards**

* Donors by blood group & location.
* Pending vs fulfilled requests.
* Monthly/weekly donation reports.
* Upcoming donation camps.

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

**Requirement Gathering**

* Donors should be able to register with personal and blood-related details.
* Hospitals should be able to place blood requests.
* System should track donation history and eligibility.
* Automated notifications (email/SMS) required for reminders.
* Admins should have dashboards for insights.

**Stakeholder Analysis**

* **Donors**: Provide blood, receive reminders.
* **Hospitals**: Request blood, track status, fulfil demand.
* **Admins/Blood Bank Staff**: Manage data, monitor dashboards, approve requests.
* **Volunteers**: Organize donation camps, connect donors with hospitals.

**Business Process Mapping**

1. Donor Registration → Save donor details.
2. Hospital creates a Blood Request → System finds eligible donors.
3. Notifications → Donors alerted via email/SMS.
4. Donation Camp → Scheduled & tracked in Salesforce.
5. Admin monitors reports & dashboards.

**Industry-specific Use Case Analysis**

* Healthcare + NGO sector requires **real-time donor availability**.
* Blood banks need **compliance with donation cycle rules** (90 days gap).
* Hospitals require **fast donor matching**.

**AppExchange Exploration**

* Explore Salesforce **Health Cloud** add-ons.
* Explore **SMS/email apps** from AppExchange (e.g., Twilio for SMS, Mailchimp for campaigns).
* Use these if integration is allowed/required.

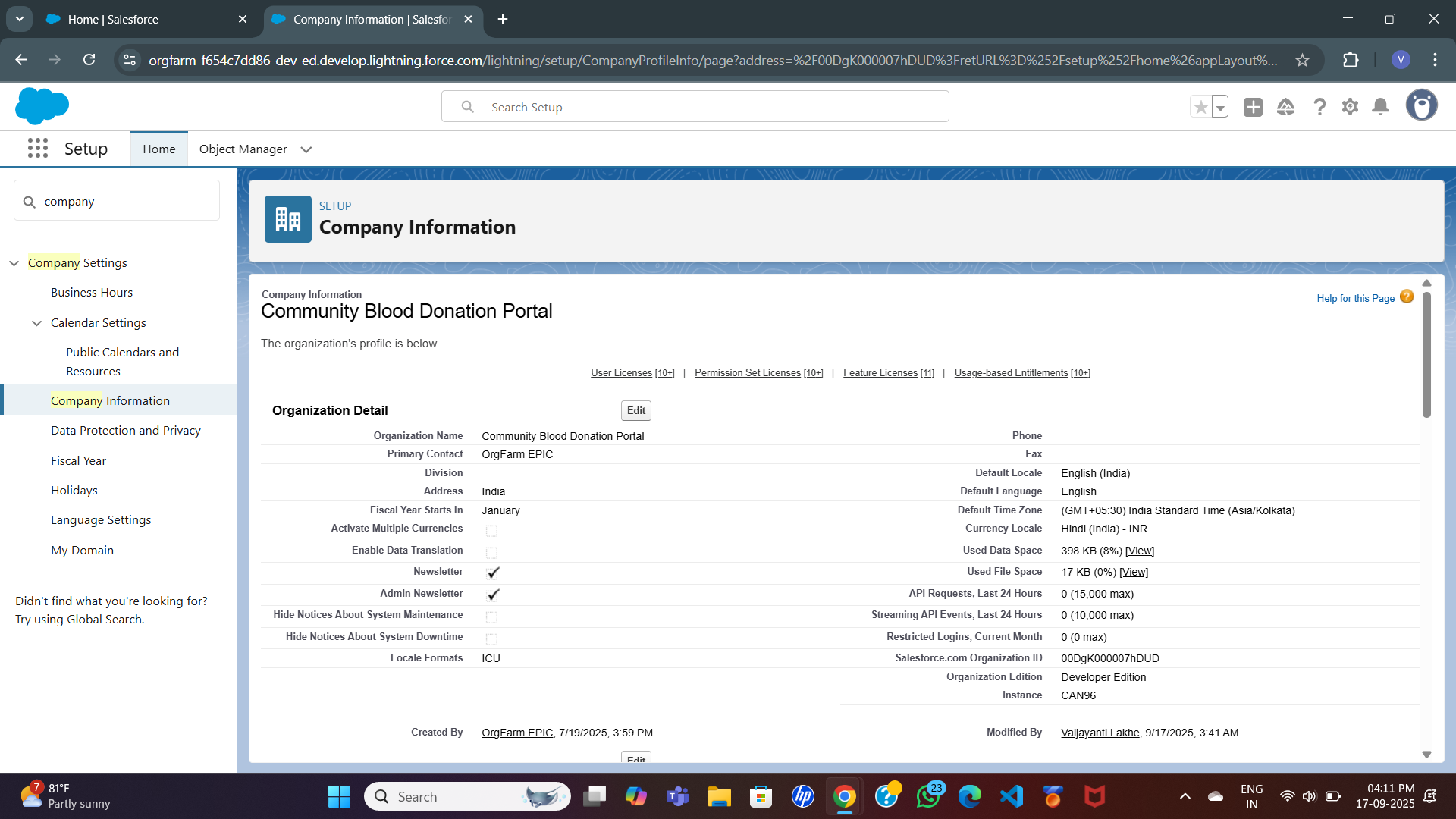
**🔹 Phase 2: Org Setup & Configuration**

**Salesforce Editions**

* Use **Salesforce Developer Edition** for project.

**Company Profile Setup**

* **Company Name:** Community Blood Donation Portal
* **Default Currency:** INR (₹)
* **Locale & Language:** India (English, DD/MM/YYYY format)
* **Time Zone:** Asia/Kolkata (GMT+5:30)



**Business Hours & Holidays**

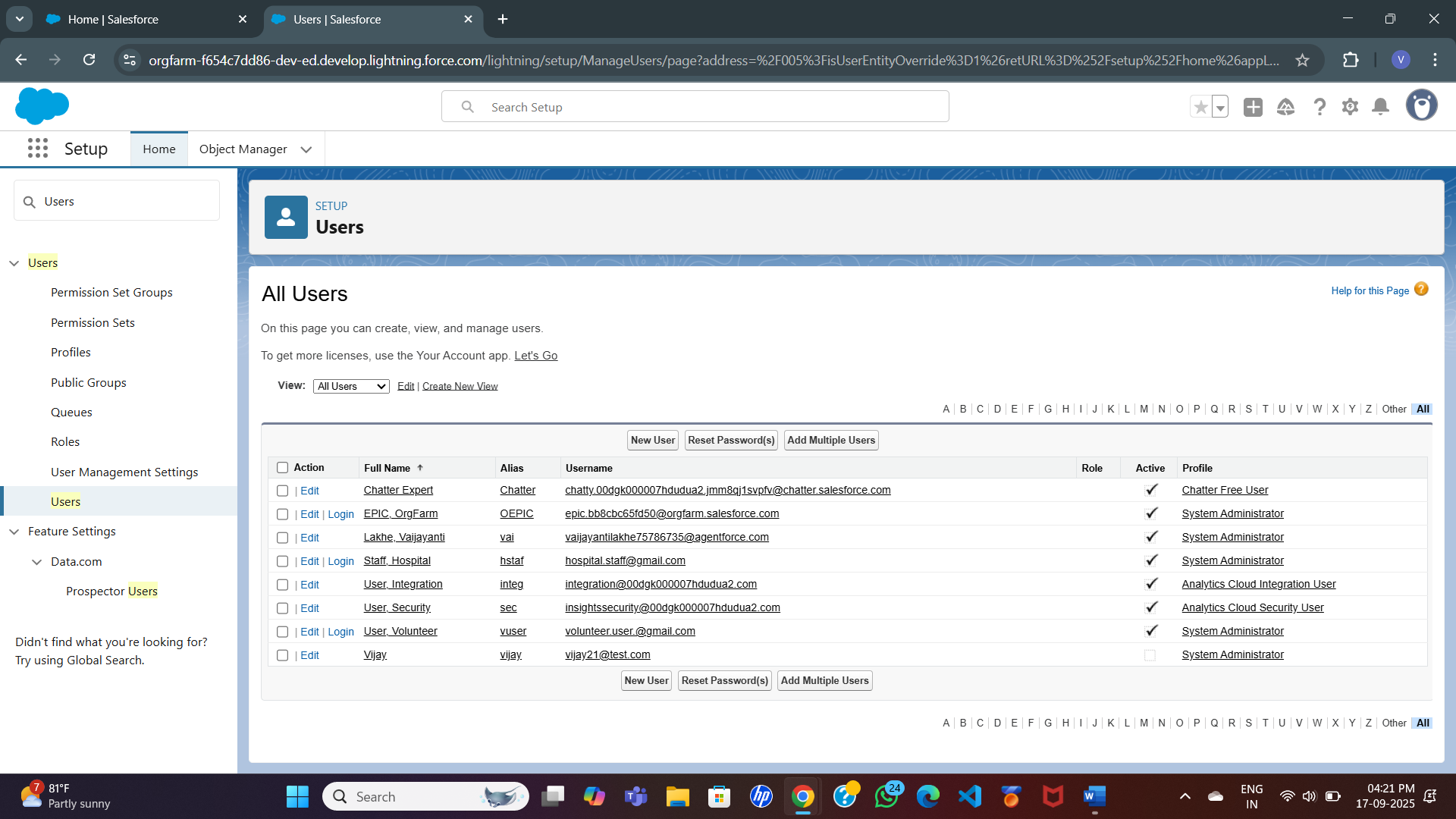
* **Business Hours:** 9:00 AM – 6:00 PM (Monday – Saturday).
* **Holidays:** National holidays (dummy setup).

**Fiscal Year Settings**

* Using the **Standard Fiscal Year (Jan–Dec)** for simplicity.
* Ensures financial tracking of donations and events can be reported on a yearly basis.

**User Setup & Licenses**

* **Admin User** – Full system access to manage configurations, security, and reporting.
* **Hospital User** – Access to create/view blood requests and monitor dashboards.
* **Volunteer User** – Access to manage donor records and schedule donation camps.

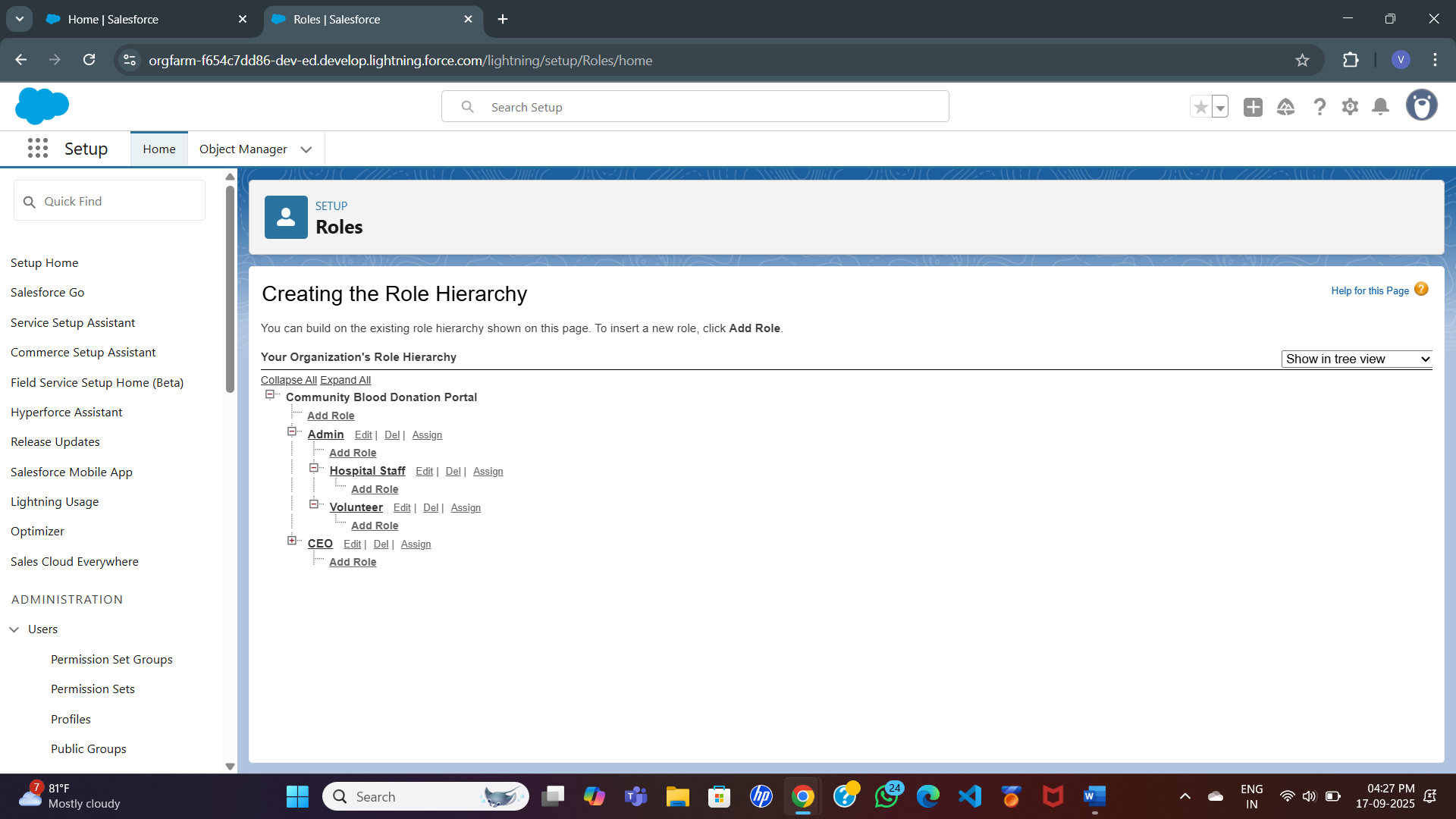


**Profiles**

* **Admin Profile** – Complete system access with CRUD (Create, Read, Update, Delete) permissions.
* **Hospital Profile** – Restricted to blood requests, dashboards, and reporting features.
* **Volunteer Profile** – Access to donor records, camps, and notifications.

**Roles**

* **Admin Role** – Top-level hierarchy, manages hospitals and volunteers.
* **Hospital Staff Role** – Handles donor requests and fulfilment workflows.
* **Volunteer Role** – Coordinates with donors and manages donation camp events.



Rationale: Role hierarchy ensures data visibility flows from Admin → Hospital Staff → Volunteers.

**Permission Sets**

* **Notification Manager** → Grants permission to send donor reminders and alerts.
* **Report Viewer** → Grants access to view reports/dashboards without editing data.

**OWD (Org-Wide Defaults)**

* **Donor Records** → Private (to protect sensitive health information).
* **Requests** → Public Read/Write (hospitals + admins).
* **Camps** → Public Read/Write (accessible to all users).

**Sharing Rules**

* Hospitals can share requests with other hospitals in case of emergencies.
* Volunteers can share donor lists with Admins for quick allocation.

**Login Access Policies**

* Enable login-as admin to troubleshoot hospital/volunteer accounts.

**Dev Org Setup**

* Salesforce **Developer Org** will be the primary workspace for configuration and implementation.
* All testing, validation, and demo preparation will be performed within this environment.

**Deployment Basics**

* Not applicable for this project, as all configurations and development will be done directly in the Salesforce Developer Org.

**🔹 Phase 3: Data Modelling & Relationships**

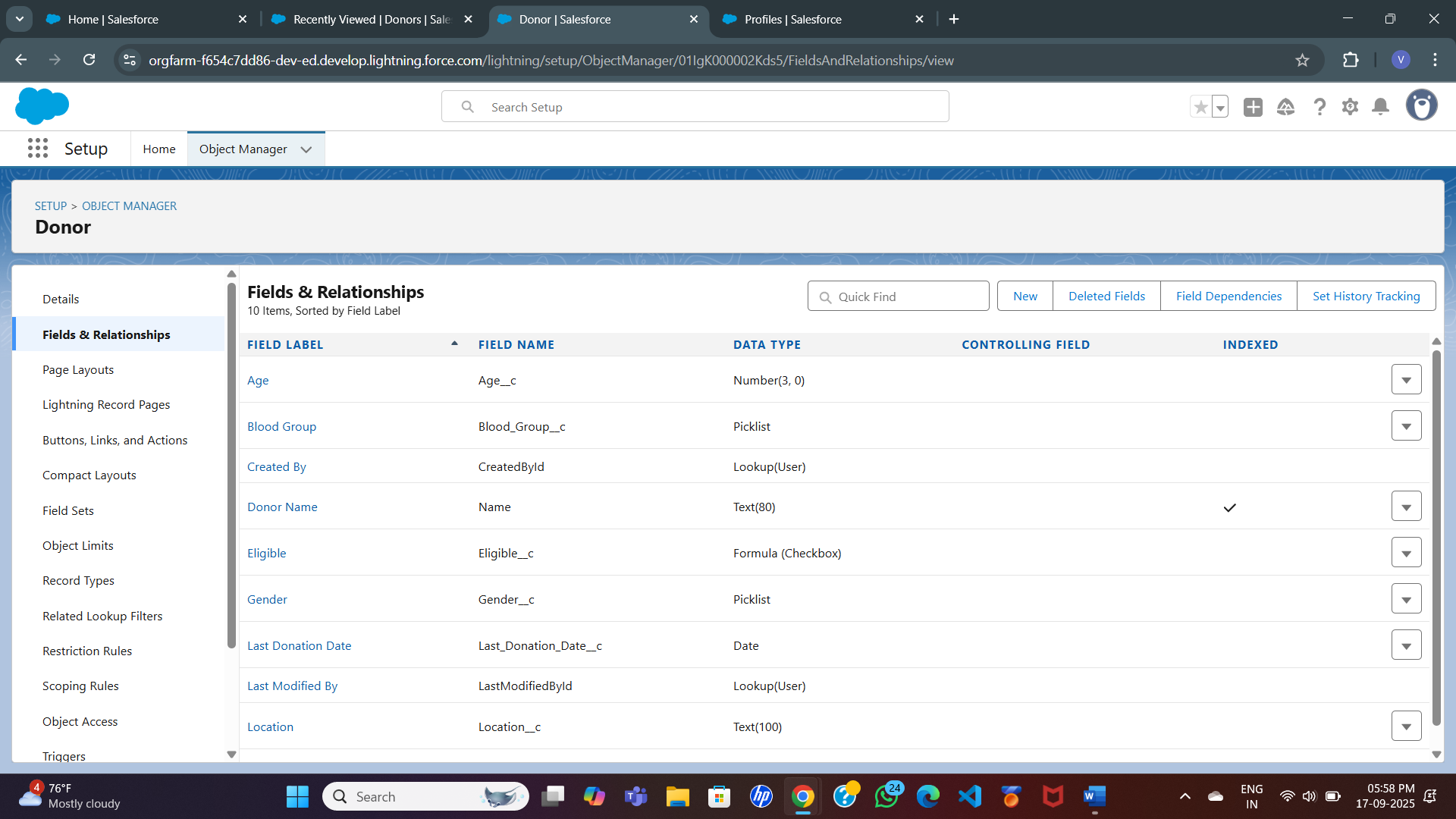
**Standard & Custom Objects**

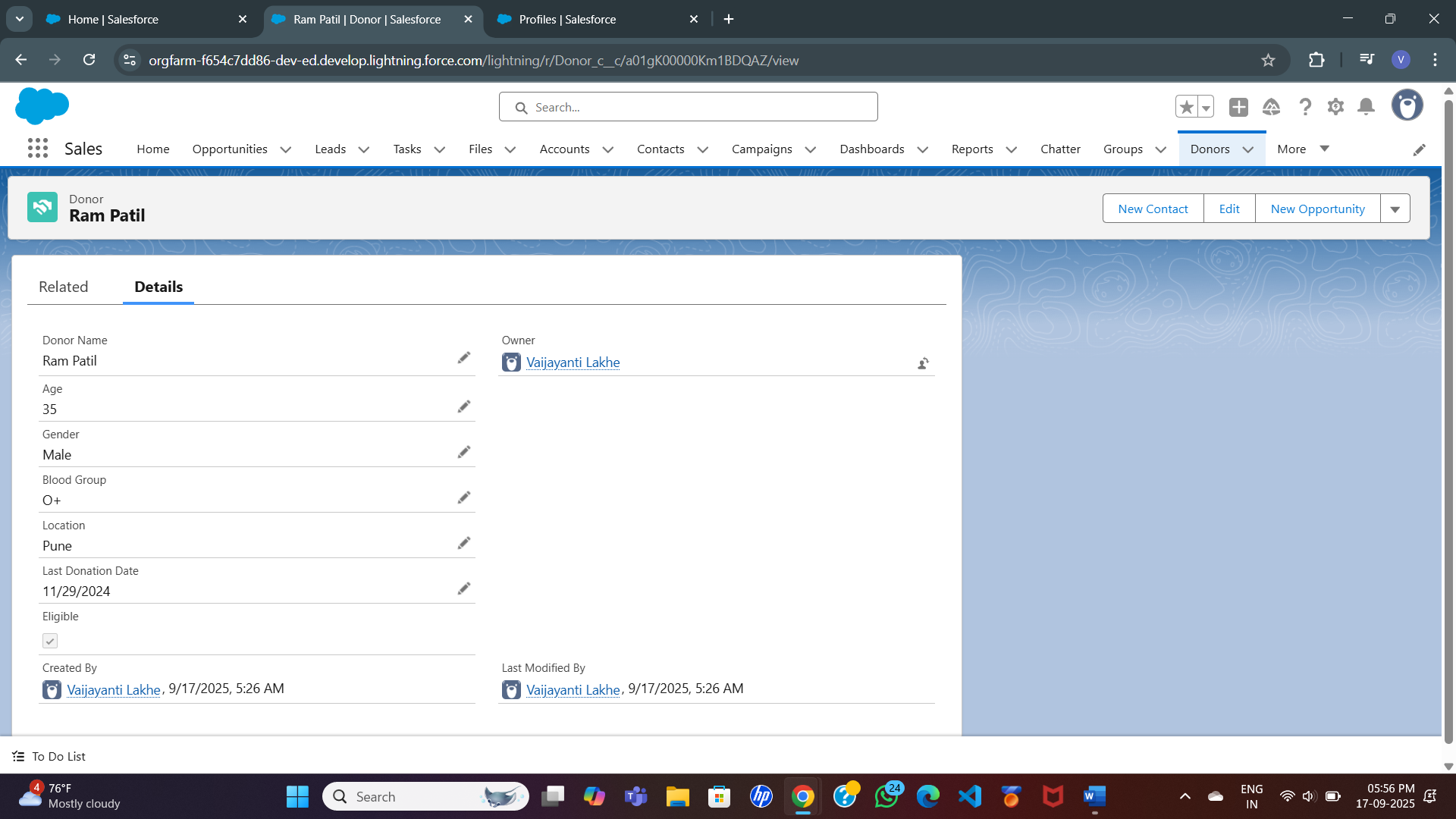
Salesforce provides some standard objects (like Users, Accounts, Contacts), but since our project is domain-specific, we will create custom objects to represent donors, hospitals, requests, and camps.

**Custom Objects & Fields**

**1. Donor**

* **Fields:**
  + Donor Name (Text)
  + Age (Number)
  + Gender (Picklist: Male/Female/Other)
  + Blood Group (Picklist: A+, A-, B+, B-, AB+, AB-, O+, O-)
  + Contact Number (Phone)
  + Email (Email)
  + Location (Text/Geolocation)
  + Last Donation Date (Date)
  + Eligible to Donate? (Formula: TRUE if Last Donation ≥ 90 days ago)





**2. Hospital**

* **Fields:**
  + Hospital Name (Text)
  + Address (Text Area)
  + Contact Person (Text)
  + Contact Number (Phone)
  + Email (Email)
  + City / Region (Text)

**3. Blood Request**

* **Fields:**
  + Request ID (Auto-Number)
  + Requested By (Lookup → Hospital)
  + Blood Group Needed (Picklist)
  + Units Required (Number)
  + Request Date (Date)
  + Status (Picklist: Pending, Approved, Fulfilled, Closed)
  + Assigned Donors (Lookup/Junction with Donor)

**4. Donation Camp**

* **Fields:**
  + Camp Name (Text)
  + Location (Text)
  + Date (Date)
  + Organized By (Lookup → Volunteer/User)
  + Number of Donors Participated (Roll-up Summary)

**5. Donation History (Junction Object)**

* Purpose: Track **many-to-many relationship** between Donors and Requests.
* **Fields:**
  + Donor (Lookup → Donor)
  + Blood Request (Lookup → Blood Request)
  + Units Donated (Number)
  + Donation Date (Date)

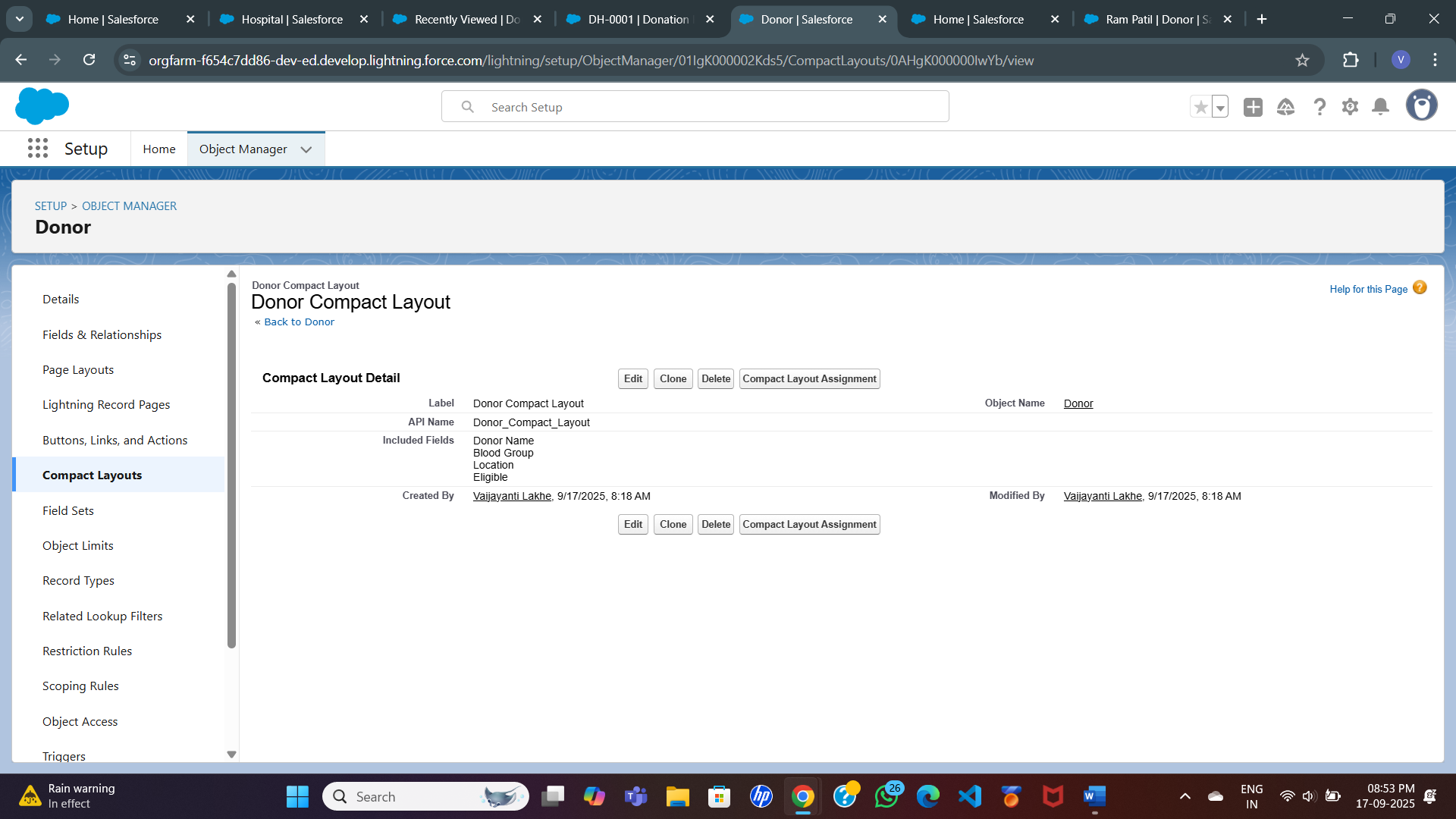
**Record Types**

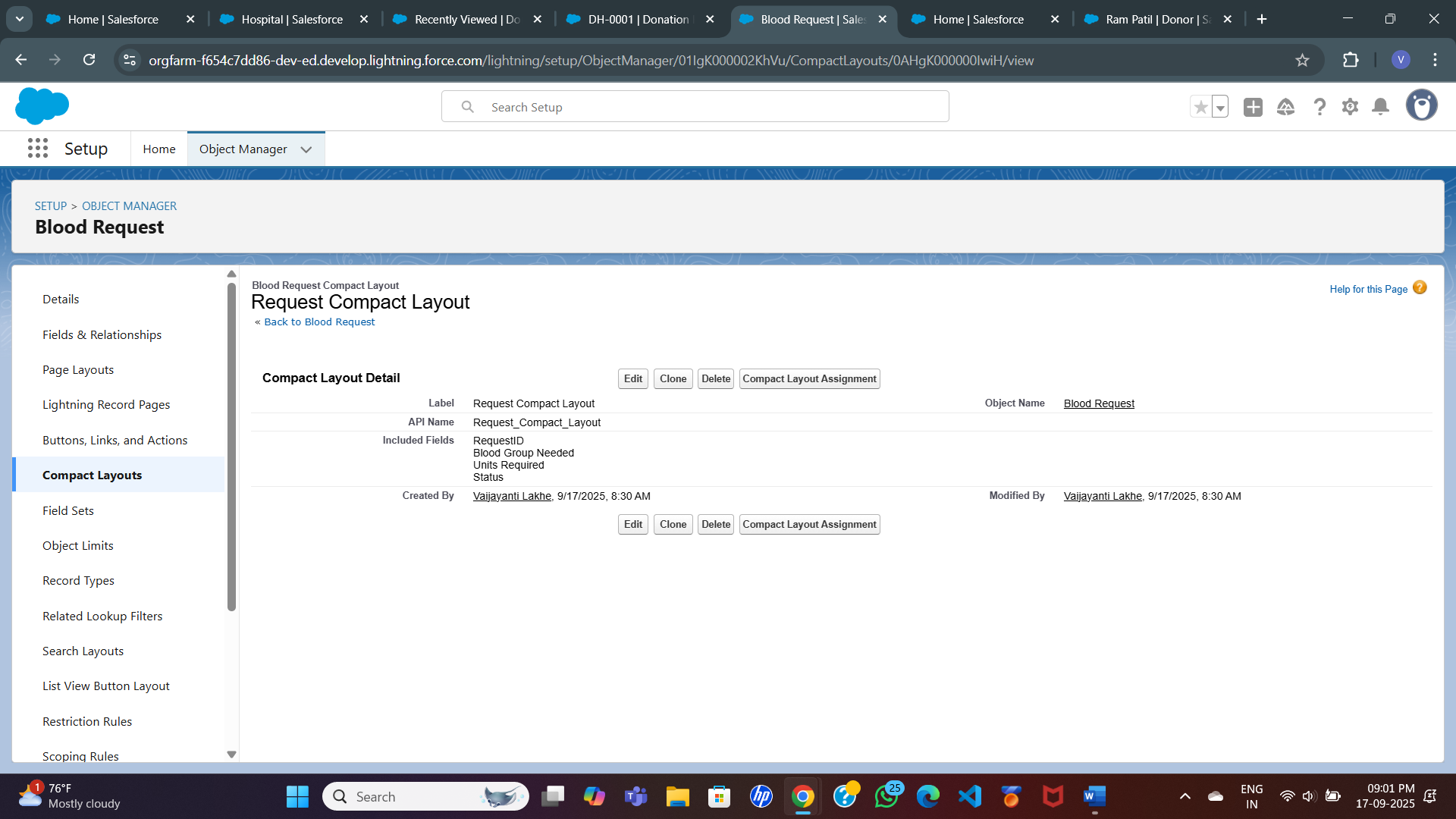
**Do you need it? 👉 Optional.**

* Create **different record types** for Hospital Requests (e.g., Emergency Request, Planned Request).

**Page Layouts**

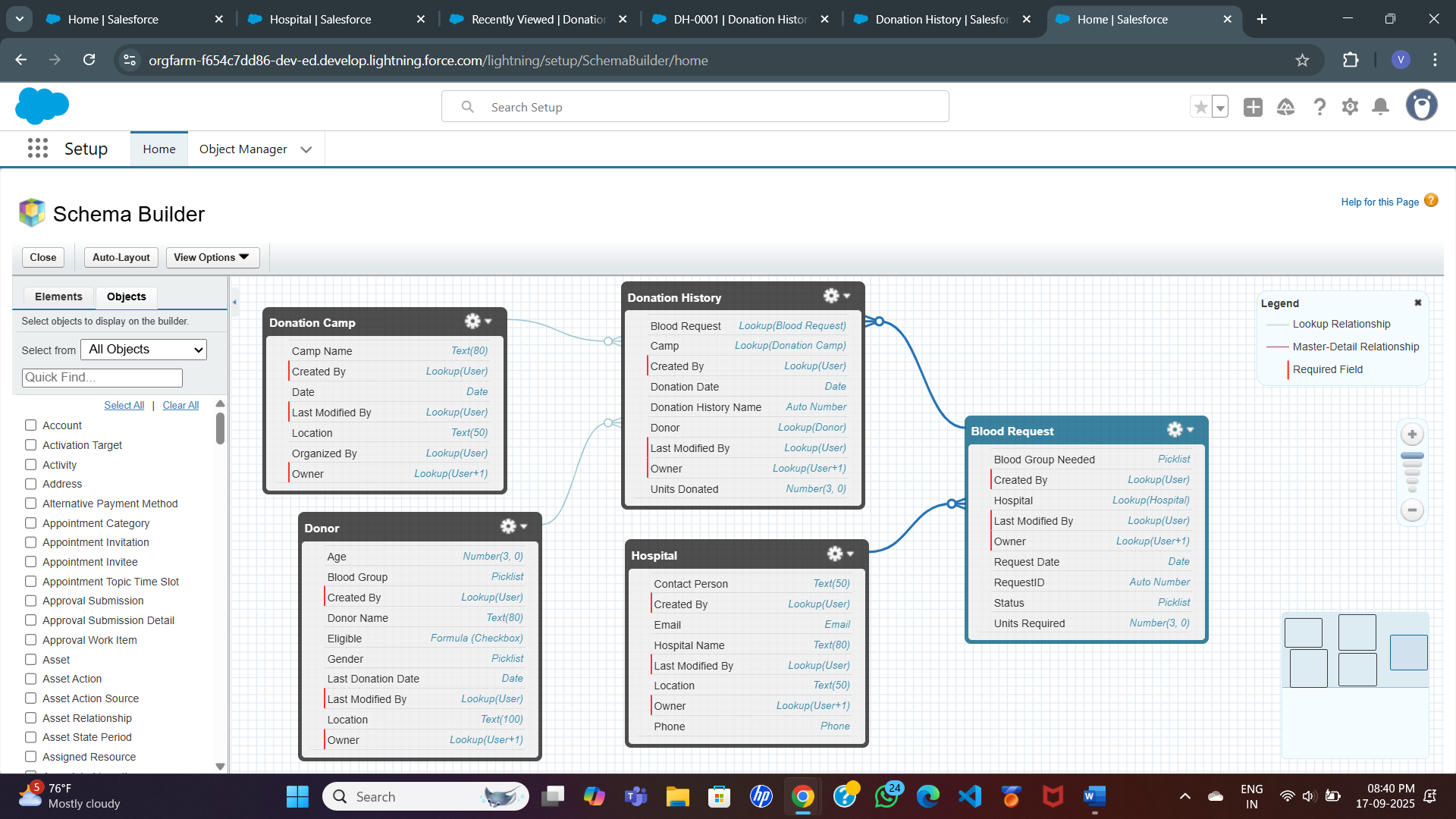
* **Donor Page Layout** → Show blood group, last donation date, eligibility flag.
* **Request Page Layout** → Show hospital, required units, assigned donors, and status.
* **Camp Page Layout** → Show event details + participating donors.





**Schema Builder**

* Use Salesforce **Schema Builder** to design the relationships visually.
* Helps stakeholders understand data flow:
  + Hospital → Request → Donation History → Donor
  + Volunteer → Camp → Donor Participation



**Relationships**

* **Hospital → Blood Request**: One hospital can create many requests. (Lookup)
* **Donor → Donation History**: One donor can donate multiple times. (Lookup)
* **Blood Request → Donation History**: One request can be fulfilled by multiple donors. (Lookup)
* **Volunteer → Donation Camp**: One volunteer organizes multiple camps. (Lookup)

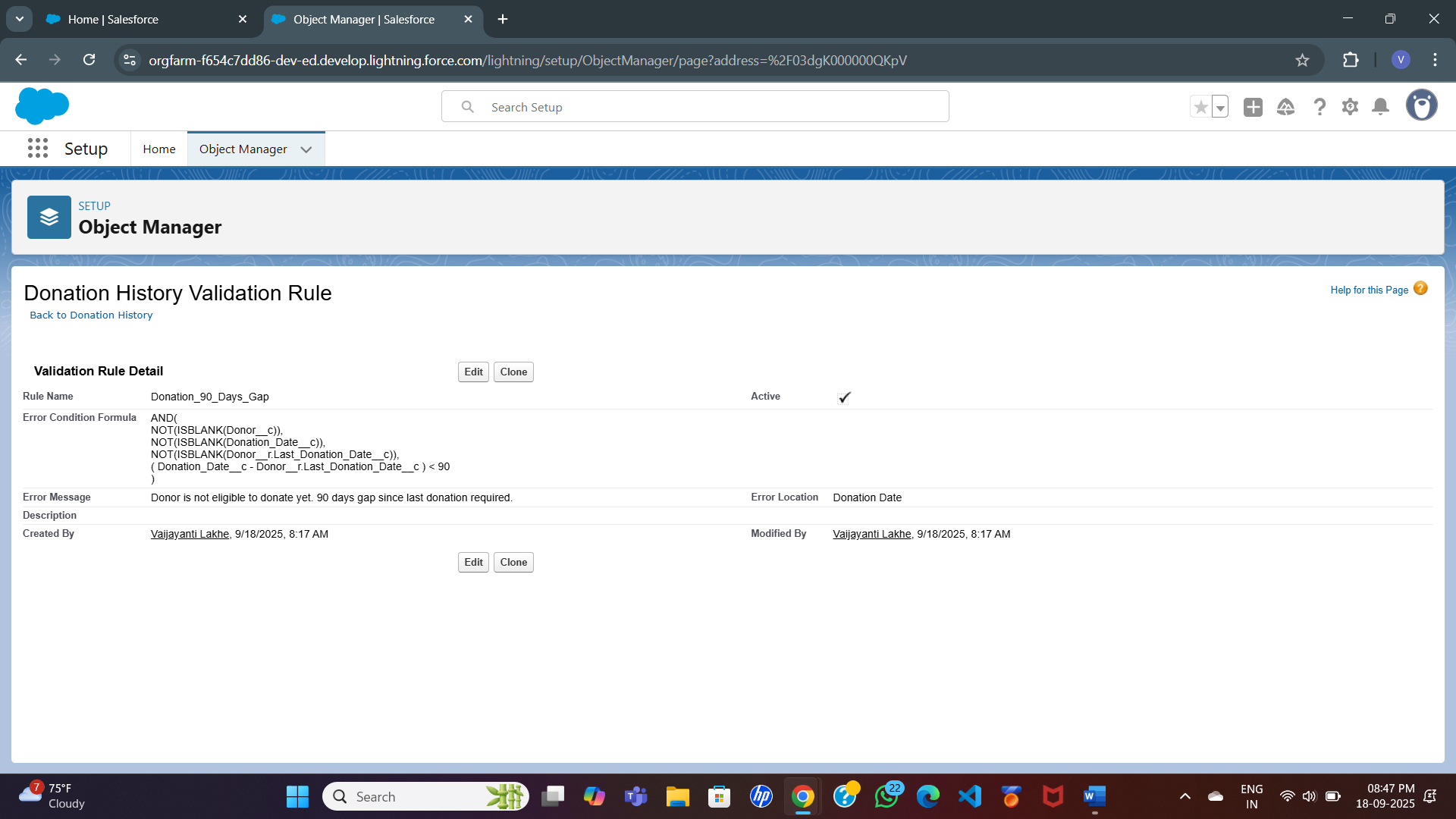
**Rationale**

* The **Donor–Request many-to-many relationship** ensures flexibility (one donor can fulfil multiple requests, and one request can be served by multiple donors).
* The **Eligibility formula field** ensures compliance with the 90-day donation rule.
* Separation of **Hospital, Donor, and Camp objects** keeps data organized and realistic to actual blood bank operations.

**🔹Phase 4: Process Automation (Admin)**

**Validation Rules**

* Ensure minimum **age for donor = 18 years.**
* Prevent donation if **Last Donation Date < 90 days ago.**
* Ensure **Units Required > 0** for Blood Requests.



**Workflow Rules (if used)**

* Auto-send a **“Thank You” email** after donation record is created.
* Send **notification to Hospital** when request status = Fulfilled.

**Process Builder (or Flow replacement)**

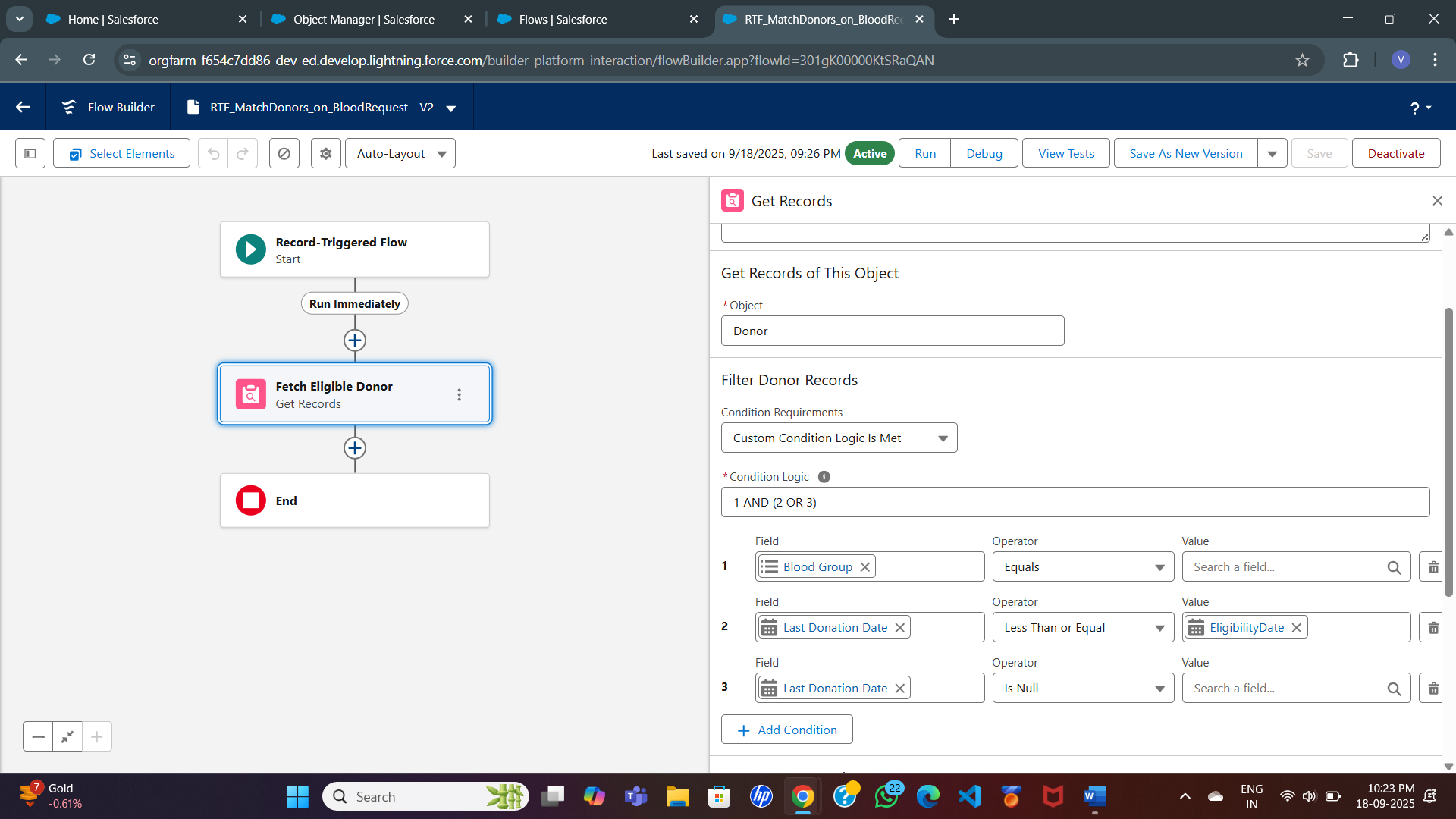
* Automatically update **Request Status** to “In Progress” when at least one donor is matched.
* Automatically update **Request Status** to “Fulfilled” when required units are met.

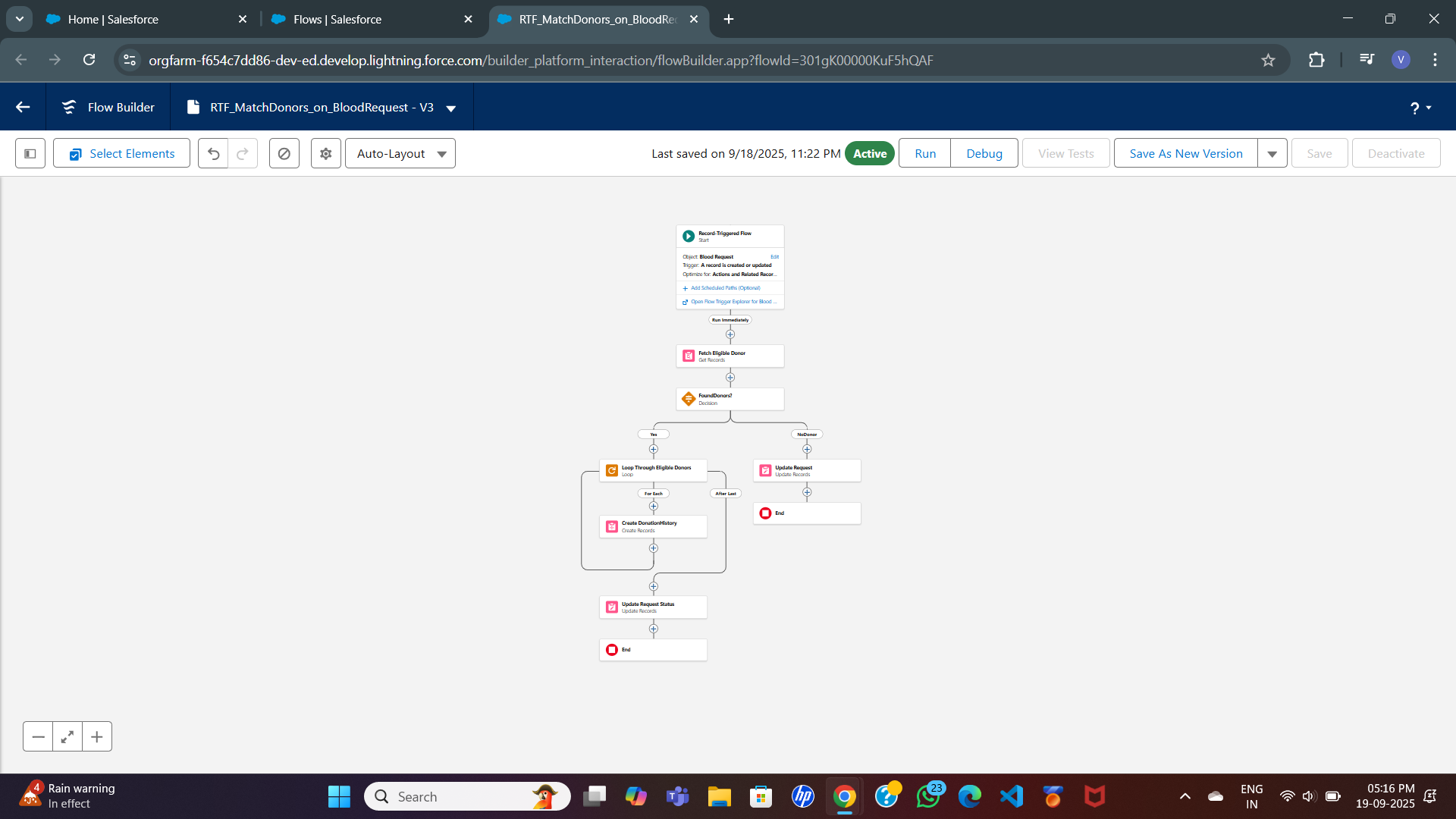
**Approval Process (optional)**

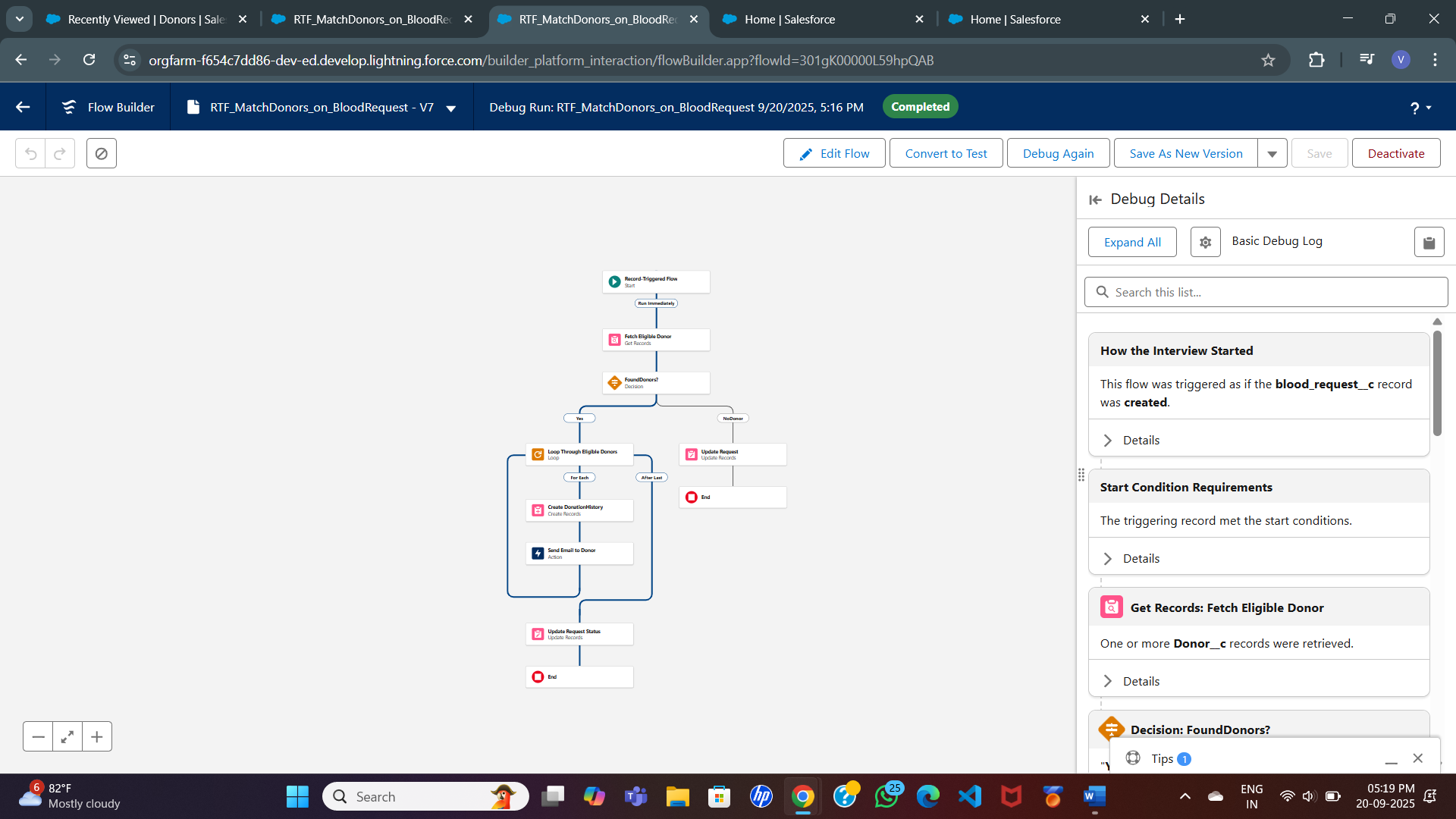
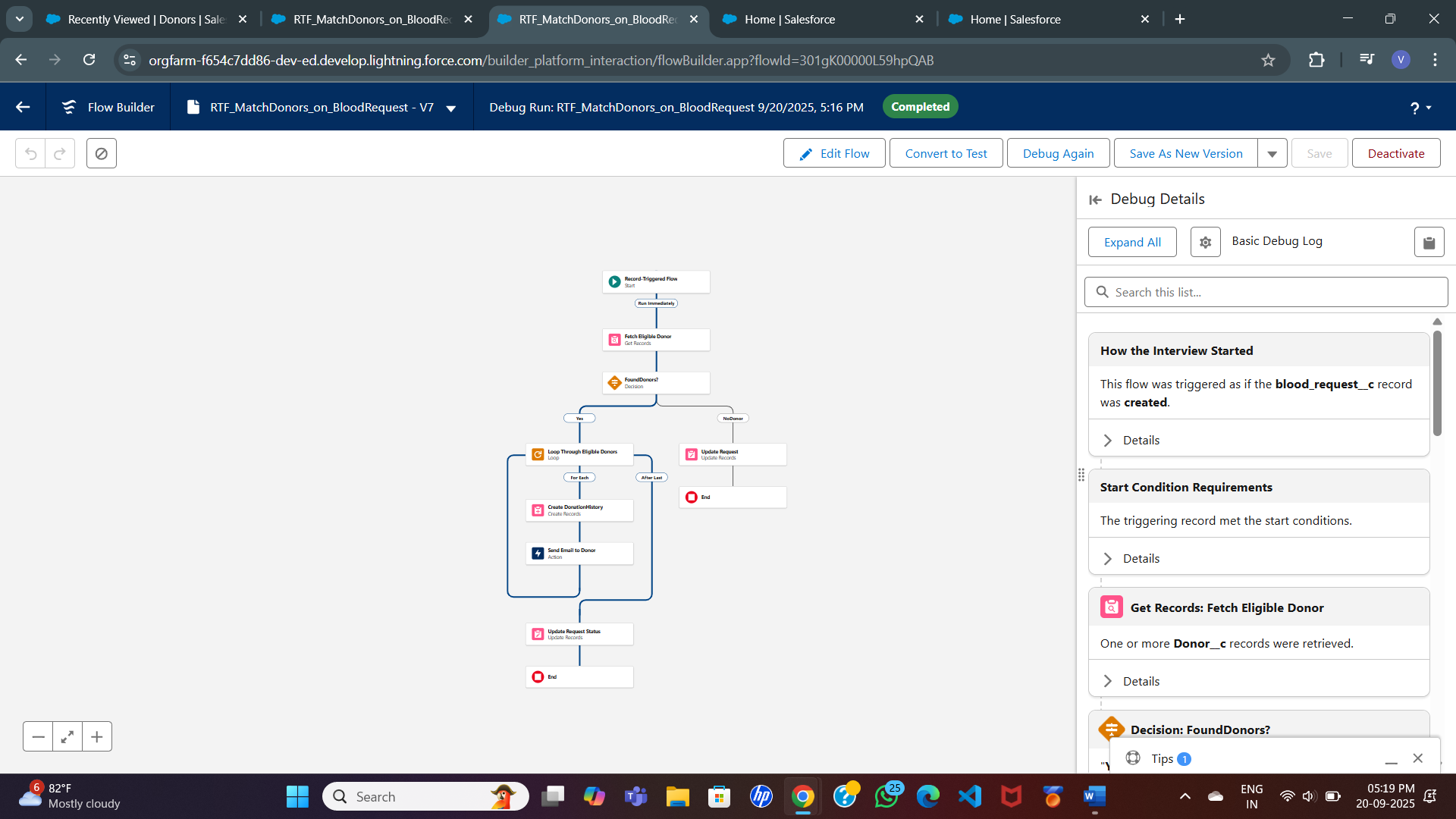
* Hospitals submit a request → Admin approves → then notifications are sent to eligible donors.
* Adds realism to critical request management.

**Flow Builder (Recommended)**

* **Record-Triggered Flow:**  
  When a **Blood Request** is created → find eligible donors (same Blood Group, Last Donation ≥ 90 days).  
  → Update “Eligible Donors” field in Request OR create related Donation History records.
* **Scheduled Flow:**  
  Runs every day → checks donors whose **90-day eligibility window just completed** → sends them reminder email/SMS.
* **Screen Flow:**  
  Volunteers use it during Camps → select donor list → auto-create Donation History records.

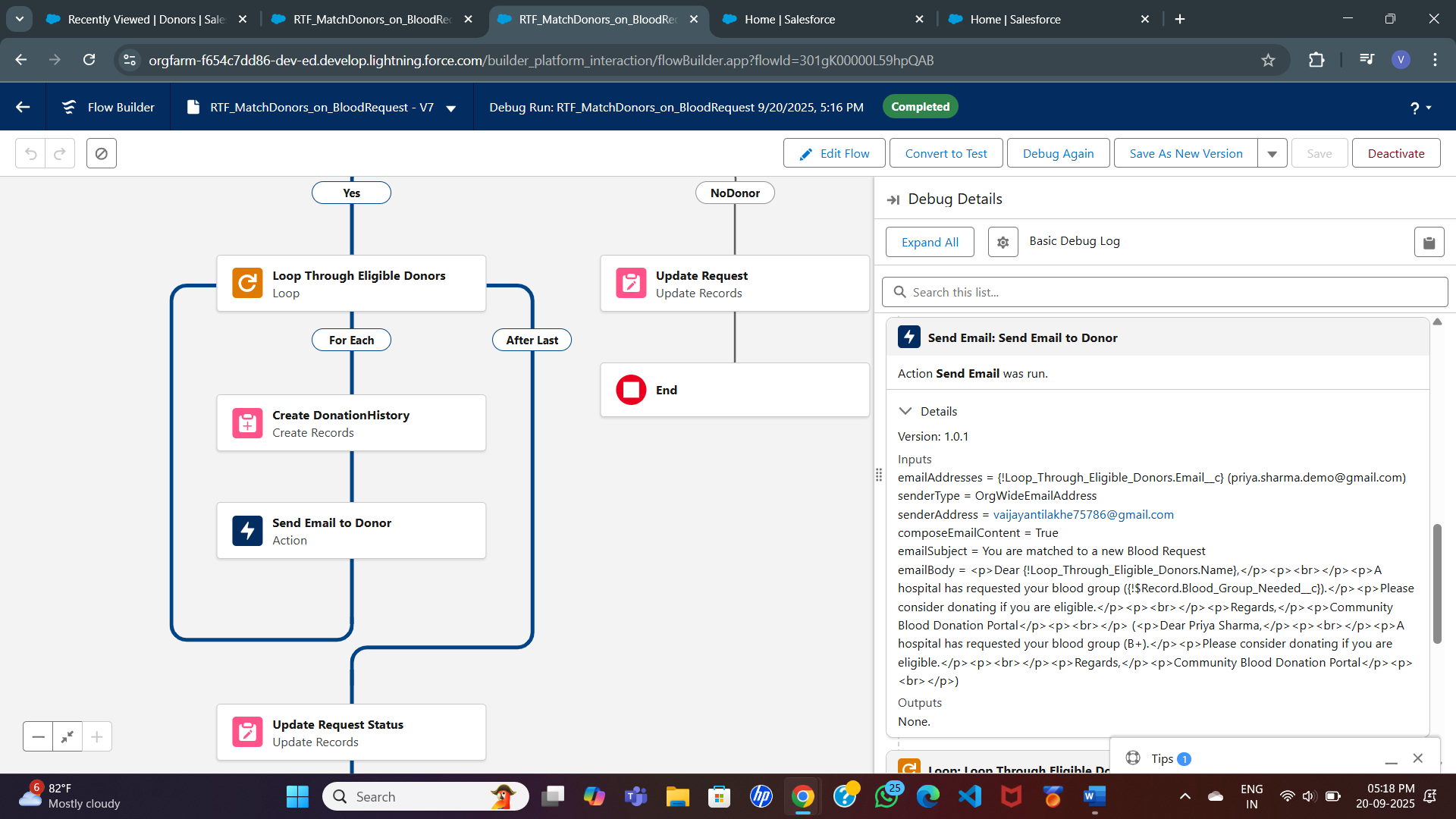






**Email Alerts / Notifications**

* Email notification to Donor when matched to a Request.



*Rationale:* Process automation ensures **accuracy, compliance with donation cycle, and faster matching** between Hospitals and Donors.

**🔹 Phase 5: Apex Programming (Developer)**

**Classes & Objects**

* Create an **Apex Class**: DonorMatchingService to handle donor eligibility logic.

**Apex Trigger Example**

* **Trigger:** Runs on after insert of Blood Request.
* **Logic:**
  + Fetch donors with same Blood Group.
  + Filter where Last Donation Date ≥ 90 days.
  + Mark as “Eligible” and relate them to the Request (via Donation History).

trigger BloodRequestTrigger on Blood\_Request\_\_c (after insert) {

for(Blood\_Request\_\_c req : Trigger.new){

List<Donor\_\_c> donors = [

SELECT Id, Name, Blood\_Group\_\_c, Last\_Donation\_Date\_\_c

FROM Donor\_\_c

WHERE Blood\_Group\_\_c = :req.Blood\_Group\_Needed\_\_c

AND Last\_Donation\_Date\_\_c <= :Date.today().addDays(-90)

];

List<Donation\_History\_\_c> historyRecords = new List<Donation\_History\_\_c>();

for(Donor\_\_c d : donors){

Donation\_History\_\_c h = new Donation\_History\_\_c();

h.Donor\_\_c = d.Id;

h.Blood\_Request\_\_c = req.Id;

h.Donation\_Date\_\_c = Date.today();

historyRecords.add(h);

}

if(!historyRecords.isEmpty()){

insert historyRecords;

}

}

}

**SOQL & Collections**

* SOQL is used to **query eligible donors**.
* Collections (Lists, Maps) store donors and donation history records.

**Batch Apex / Scheduled Apex (Future Scope)**

* Scheduled Apex to check every day for **donors who just became eligible** and send them notifications.

**Test Classes**

* Write test methods to validate trigger works correctly.

***Rationale:*** Apex provides **flexibility and performance** for donor matching and notifications. Even if you use Flows, showing sample Apex in the report proves you understand both **Admin + Developer** perspectives.

**🔹 Phase 6: User Interface Development**

**Lightning App Builder**

* Create a custom **Lightning App** named **“Blood Donation Management”**.
* Add custom tabs for:
  + Donors
  + Hospitals
  + Blood Requests
  + Donation Camps
  + Donation History
* Provide role-based navigation:
  + **Hospital Users** → See Requests & Reports
  + **Volunteers** → See Donors & Camps
  + **Admins** → See everything

**Record Pages**

* **Donor Record Page**:
  + Compact Layout at top → Donor Name, Blood Group, Location, Eligibility.
  + Related Lists → Donation History, Upcoming Camps.
* **Request Record Page**:
  + Highlights → Request ID, Blood Group Needed, Units, Status.
  + Related Donors (eligible matches).
* **Hospital Page**:
  + Contact details, related requests.
* **Camp Page**:
  + Details of camp + donor participation.

**Tabs**

* Create **Custom Tabs** for each custom object (Donor, Request, Hospital, Camp, Donation History).
* Group them in the **Blood Donation Management App** for easy navigation.

**Home Page Layouts**

* Customize the **Home Page** for different profiles:
  + **Admin Home** → Quick links to reports, dashboards, donor approvals.
  + **Hospital Home** → Quick action to “Create New Request.”
  + **Volunteer Home** → Quick action to “Schedule Donation Camp.”

**Utility Bar**

* Add **Quick Actions** in the utility bar:
  + “Add New Donor”
  + “Create Request”
  + “Schedule Camp”
* Helps users perform frequent tasks faster.

**Lightning Web Components (Optional – Advanced)**

*(Only if you want to add some coding part for extra credit)*

* Example LWC: **Donor Eligibility Checker**
  + Input: Blood Group + Location
  + Output: List of eligible donors (queried using Apex).
* Example LWC: **Camp Registration Form**
  + Allow volunteers to add donors directly into a camp record.

**Apex with LWC (Optional)**

* Use Apex controller class to fetch data dynamically (e.g., list of eligible donors).
* Display results in LWC with **filtering options** (Blood Group, City).

**Navigation Service**

* Create a button on Blood Request record: **“View Eligible Donors”** → navigates to related Donor list page.
* Provide smooth transitions between objects.

***Rationale****:* This phase ensures the system is **user-friendly**, with clear navigation, quick actions, and dashboards that provide instant visibility. Even without LWCs, the **App Builder + Tabs + Page Layouts** will give your project a polished look for demo.

**🔹 Phase 7: Integration & External Access**

**Named Credentials**

* Configure a **Named Credential** for secure connection with external services (e.g., Twilio for SMS, Gmail API for email).

**External Services (Optional)**

* Could expose hospital requests as a **REST API** for other hospitals to query availability.
* Example: A partner hospital could send a callout → system responds with donor availability.

**Web Services (REST Callouts)**

* Use REST API integration for SMS/email notifications (optional).
* Example: When donor is eligible, system calls SMS API → sends reminder.

**Platform Events (Optional)**

* Trigger notifications when a **new request** is created.
* Volunteers subscribed to events can act quickly.

**Remote Site Settings**

* Add external endpoints (like SMS provider) to Salesforce.

*Rationale:* While external integration is optional for students, mentioning it shows awareness of **scalability**. For beginner scope → stick to **Salesforce email alerts**.

**🔹 Phase 8: Data Management & Deployment**

**Data Import Wizard**

* Import initial donor, hospital, and request records from CSV.
* Helps test the system with sample data.

**Data Loader**

* For bulk data operations (e.g., loading 100 donor records).

**Duplicate Rules**

* Prevent multiple records for the same donor (check by Email + Phone).

**Data Export**

* Weekly export of donor and request data for backup.

**Deployment**

* Entire project is developed and tested inside **Developer Org**.
* Sandbox/Change Sets → *Not required for student project*.

*Rationale:* Data management ensures **clean, reliable donor and hospital data** for smooth project demo.

**🔹 Phase 9: Reporting, Dashboards & Security Review**

**Reports**

* Donors by Blood Group and Location.
* Pending vs Fulfilled Requests.
* Donation History (Donors & Dates).
* Upcoming Donation Camps.

**Dashboards**

* **Admin Dashboard:** Blood availability, active requests, total donations.
* **Hospital Dashboard:** Requests created, fulfilled requests, donor matches.
* **Volunteer Dashboard:** Upcoming camps, donor participation.

**Security Review**

* **Sharing Settings:** Donor records private (sensitive info).
* **Field-Level Security:** Hide contact details from volunteers if not needed.
* **Audit Trail:** Track who updated donor or request data.
* **Login IP Ranges (Optional):** Restrict hospital logins to trusted IPs.

*Rationale:* Reporting shows **real-time insights** and Security ensures **data privacy**.

**🔹 Phase 10: Final Presentation & Demo Day**

**Pitch Presentation**

* Cover: Problem → Solution → Salesforce Implementation → Benefits → Future Enhancements.

**Demo Walkthrough**

* Flow:

Create Hospital Request → System finds Eligible Donors → Notifications sent → Request Fulfilled → Dashboard updates.

**Feedback Collection**

* Collect mentor feedback for improvement.

**Handoff Documentation**

* Submit project report (phases, ERD, workflows, dashboards).

**LinkedIn/Portfolio Showcase**

* Add project details, screenshots, and learning outcomes to LinkedIn/GitHub portfolio.

*Rationale:* The final phase focuses on **professional presentation & documentation**.