

Smart Blood & Organ Donation CRM – Life Saving Support System

Overview

The ****Smart Blood & Organ Donation CRM**** is a Salesforce-based solution designed to connect hospitals, NGOs, donors, and patients in critical situations.

It simplifies donor registration, automates emergency matching, and ensures life-saving blood or organ donations reach patients on time.

Problem Statement

Patients in emergencies often struggle to find matching blood or organ donors quickly.

The existing process is slow, dependent on manual calls, and lacks a centralized donor database.

This leads to:

- Delay in finding matching donors
- Patients losing lives due to late response
- Donors not receiving timely notifications
- Lack of real-time tracking of donations and pledges

Proposed Salesforce Solution

- Centralized donor registration with blood group, organ pledge, and location
- Emergency request management by hospitals/NGOs
- Smart donor matching based on blood group, location, and availability
- Automated notifications (SMS/Email) to nearby donors in emergencies
- Appointment scheduling and donation tracking
- Donation history with reminders for next eligible donation date
- Real-time dashboards for hospitals, NGOs, and authorities

Use Cases

1. **Donor Registration & Management**– Donors can register with details like blood group, organ pledge, and availability.
2. **Hospital Emergency Requests** – Hospitals can log urgent blood/organ requests.
3. **Smart Matching & Alerts**– System automatically matches and notifies nearby eligible donors.
4. **Appointment Scheduling** – Donors confirm donation slots, hospitals update status.
5. **Donation History & Follow-ups** – Tracks donor history and sends reminders when eligible again.
6. **Organ Pledge Tracking** – Maintains a registry of pledged organs for future needs.
7. **NGO & Volunteer Collaboration**– NGOs manage blood camps and donor mobilization.
8. **Dashboards & Reporting** – Real-time insights on donor availability, requests, and completed donations.

Salesforce Features Used

Data Modeling: Custom objects for Donors, Requests, Appointments, Organ Pledges

Automation: Flow Builder, Email Alerts, Validation Rules, Approval Process

Apex Development: Triggers for donor matching, Scheduled Apex for reminders

Lightning UI: Record Pages, Tabs, LWC components for donor/hospital dashboards

Integration: SMS/Email API for donor alerts, possible NGO/hospital system integration

Reports & Dashboards: Donation trends, request fulfillment rate, NGO contribution

Example Dashboard Metrics

- Active donors by blood group
- Pending vs Completed donation requests
- Organ pledges by type (Kidney, Liver, Cornea, etc.)
- Lives saved through successful matches

Impact

This system ensures:

- Faster response to emergencies
- Increased donor participation through automation
- Transparent tracking of donations and pledges
- Social good by saving lives with technology

Future Enhancements

- Mobile App Integration for donor check-in
- AI-based donor prediction & availability scoring
- Government/Red Cross integration for wider impact
- Multi-language support for accessibility

Phase 1 Problem Understanding & Industry Analysis

Goal of Phase 1

To understand the pain points in the healthcare donation process, analyze stakeholders, and design an improved workflow using Salesforce CRM. This phase focuses on identifying requirements, mapping current vs. proposed processes, and researching industry best practices.

1 Requirement Gathering

What was done:

- Studied pain points of **blood banks, hospitals, NGOs, patients, and donors**.
- Identified key challenges in the current system.

Key Requirements Identified:

- Difficulty in quickly finding a suitable donor during emergencies.
- No centralized system to track donor availability (blood group, organ compatibility, location).
- Manual communication between hospitals, patients, and donors causes delays.
- Lack of reminders for repeat donations.
- No analytics for monitoring donor trends and shortages.

□ **Output:** Requirements document listing problems to solve.

2 Stakeholder Analysis

Stakeholders Identified:

- **Donors** → Individuals willing to donate blood/organs.
- **Patients & Families** → Need urgent access to donations.
- **Hospitals/Clinics** → Require quick donor matching and patient history tracking.
- **Blood Banks & NGOs** → Manage donor lists and blood stock.

- **Government Health Agencies** → Require donation data for policy and public health planning.

Goals of Stakeholders:

- Donors: Easy scheduling & reminders.
- Patients: Quick access to compatible donors.
- Hospitals: Real-time updates on donor availability.
- NGOs/Blood Banks: Centralized donor management.
- Government: Data analytics for shortages and trends.

□ **Output:** Stakeholder matrix (stakeholder + their goals).

3 Business Process Mapping

Current Process (Manual) – “As-Is”:

1. Patient urgently needs blood.
2. Family calls multiple hospitals/blood banks.
3. If a donor is found → manual coordination begins.
4. Organ donations delayed due to poor inter-hospital communication.

Proposed Process (With Salesforce CRM) – “To-Be”:

1. Patient request logged into CRM (via web/app/call center).
2. CRM searches donor database (blood group, organ compatibility, location).
3. Nearest donor notified automatically (SMS/Email/App).
4. Hospital updates donation status in real-time.
5. Dashboards show donations, stock, and shortages for administrators.

□ **Output:** “As-Is” and “To-Be” process flow diagrams.

4 Industry-Specific Use Case Analysis

Healthcare Industry Pain Points:

- Time-sensitive emergency donations.
- Lack of real-time donor-patient matching.
- No follow-up system for repeat donors.

Salesforce CRM Use Cases:

- **Lead Management:** Donor = Lead, Patient Request = Case.
- **Service Console:** Hospital staff manage requests on a single screen.
- **Automation:** Automated reminders for donor's next donation date.
- **Dashboards:** Admins monitor donation trends, stock, and shortages.

□ **Output:** Documented healthcare-specific Salesforce use cases.

5 AppExchange Exploration

Apps/Tools Explored:

- **Salesforce Health Cloud** – Comprehensive healthcare solution.
- **Donor Management Apps** – Used by nonprofits for donor records.
- **Appointment Management Add-ons** – Enhance scheduling and follow-ups.

Goal: Identify what's already available and plan unique features like **smart donor matching** and **AI-powered reminders**.

□ **Output:** AppExchange research notes.

End of Phase 1 Deliverables

- Problem statement & requirements document.
- Stakeholder analysis chart.
- Business process flow ("As-Is" vs "To-Be").
- Healthcare-specific Salesforce use cases.
- AppExchange research notes.

Phase 2 – Org Setup & Configuration

Goal of Phase 2

To configure the Salesforce Org with **company details, business hours, fiscal year, users, profiles, roles, OWD, and sharing rules** so that the system is secure and ready for custom object creation.

Steps Followed

1. Salesforce Edition & Org Setup

- Logged into **Salesforce Developer Edition**.
 - Went to **Setup → Quick Find → Company Information**.
 - Verified edition = Developer Edition.
-

2. Company Profile Setup

- **Setup → Quick Find → Company Information**.
- Edited:
 - Time Zone → *Asia/Kolkata*
 - Default Currency → *INR*
 - Default Languages → *English, Telugu, Gujarati*
- Saved changes.

SETUP

Company Information

Company Information

MediLink Team

The organization's profile is below.

[User Licenses \(110\)](#) |
 [Permission Set Licenses \(110\)](#) |
 [Feature Licenses \(11\)](#) |
 [Usage-based Entitlements \(110\)](#)

Organization Detail

Edit

| | | | |
|---------------------------------------|-------------------------------------|-------------------------------------|--|
| Organization Name | MediLink Team | Phone | |
| Primary Contact | OrgFarm EPIC | Fax | |
| Division | | Default Locale | English (India) |
| Address | United States | Default Language | English |
| Fiscal Year Starts In | January | Default Time Zone | (GMT+05:30) India Standard Time (Asia/Kolkata) |
| Activate Multiple Currencies | <input type="checkbox"/> | Currency Locale | English (India) - INR |
| Enable Data Translation | <input type="checkbox"/> | Used Data Space | 354 KB (7%) View |
| Newsletter | <input type="checkbox"/> | Used File Space | 17 KB (0%) View |
| Admin Newsletter | <input checked="" type="checkbox"/> | API Requests, Last 24 Hours | 0 (15,000 max) |
| Hide Notices About System Maintenance | <input type="checkbox"/> | Streaming API Events, Last 24 Hours | 0 (10,000 max) |
| Hide Notices About System Downtime | <input type="checkbox"/> | Restricted Logins, Current Month | 0 (0 max) |
| Locale Formats | ICU | Salesforce.com Organization ID | 00DgL00000AYvKZ |
| | | Organization Edition | Developer Edition |
| | | Instance | CANG8 |

Created By

OrgFarm EPIC, 8/31/2025, 8:52 AM

Edit

Modified By

Shaik Roshan Parveen, 9/16/2025, 10:48 PM

3. Business Hours & Holidays

- **Setup → Quick Find → Business Hours.**
- Clicked **New Business Hours.**
- Entered:
 - Label: *Hospital Business Hours*
 - Time Zone: Asia/Kolkata
- Checked **Active** and **Use these as default.**
- Set **Sunday → Closed.**
- Saved.

SETUP

Business Hours

Organization Business Hours

Select the days and hours that your support team is available. These hours, when associated with escalation rules, determine the times at which cases can escalate.

If you enter blank business hours for a day, that means your organization does not operate on that day.

[Holidays \(0\)](#)

Business Hours Detail

Edit

| | | | |
|---------------------|---|------------------------|--|
| Business Hours Name | Default Business Hours (00:00-18:00 IST) | Time Zone | (GMT+05:30) India Standard Time (Asia/Kolkata) |
| Business Hours | Sunday No Hours Monday 9:00 AM to 6:00 PM Tuesday 9:00 AM to 6:00 PM Wednesday 9:00 AM to 6:00 PM Thursday 9:00 AM to 6:00 PM Friday 9:00 AM to 6:00 PM Saturday 9:00 AM to 6:00 PM | Default Business Hours | <input checked="" type="checkbox"/> |

Active

☒

Created By

Shaik Roshan Parveen, 9/12/2025, 10:29 AM

Edit

Last Modified By

Shaik Roshan Parveen, 9/12/2025, 10:34 AM

Holidays

Add/Remove

No records to display

Back To Top

Always show me fewer / more records per related list

4. Fiscal Year Settings

- **Setup** → **Quick Find** → **Fiscal Year**.
- Verified that *Standard Fiscal Year* is enabled.
- Did not enable custom fiscal year.

SETUP
Fiscal Year

Setup
Organization Fiscal Year Edit: MediLink Team

To specify the fiscal year type for your organization, choose one of the options below.

Fiscal Year Information
Your organization can change the fiscal year start month, and specify whether the fiscal year name is set to the starting or ending year. For example, if your fiscal year starts in April 2025 and ends in March 2026, your Fiscal Year setting can be either 2025 or 2026.

⚠ Changing the fiscal year shifts fiscal periods and impacts opportunities and forecasts across your organization. If your forecast periods are set to quarterly, adjusting the fiscal year start month will erase existing forecast adjustments and quotas. Consider exporting a data backup before implementing this change.

☒ **Standard Fiscal Year** ⓘ
☐ Custom Fiscal Year ⓘ

Change Fiscal Year Period Save Cancel

| | |
|-------------------------|---|
| Name | MediLink Team |
| Fiscal Year Start Month | January |
| Fiscal Year is Based On | <input checked="" type="radio"/> The ending month <input type="radio"/> The starting month |

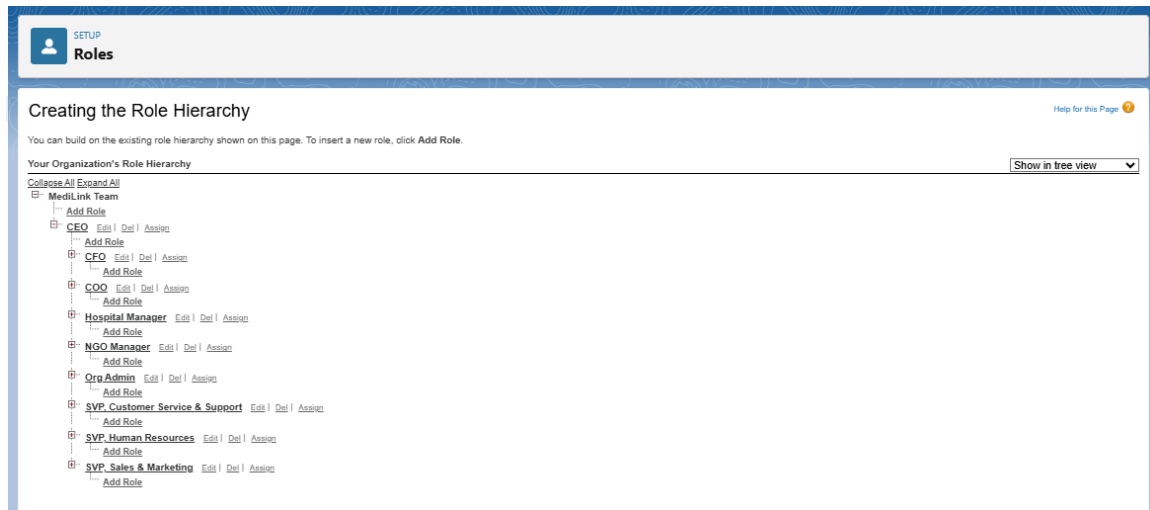
Save Cancel

5. User Setup & Licenses

- **Setup** → **Quick Find** → **Users** → **Users** → **New User**.
- Filled required fields:
 - First Name, Last Name, Email, Username.
 - User License: *Salesforce Platform*.
 - Profile: (selected custom profile after creation).
 - Role: (selected role from hierarchy).
- Checked **Generate password**.
- Saved.
- Repeated for each test user (Hospital Manager, Hospital Staff).

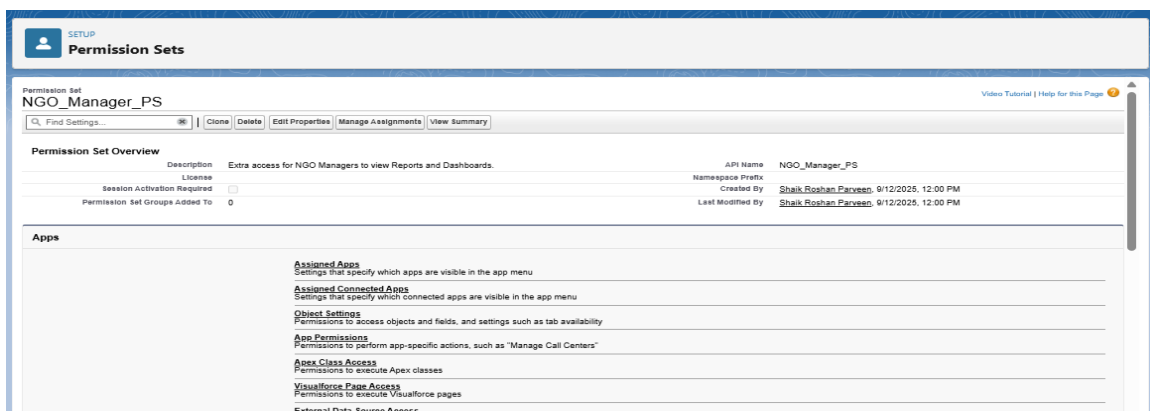
7. Roles & Role Hierarchy

- **Setup** → **Quick Find** → **Roles** → **Set Up Roles**.
- Clicked **Add Role**.
- Created roles in hierarchy:
 - System Admin (top)
 - Hospital Manager
 - Hospital Staff
- Saved each role.



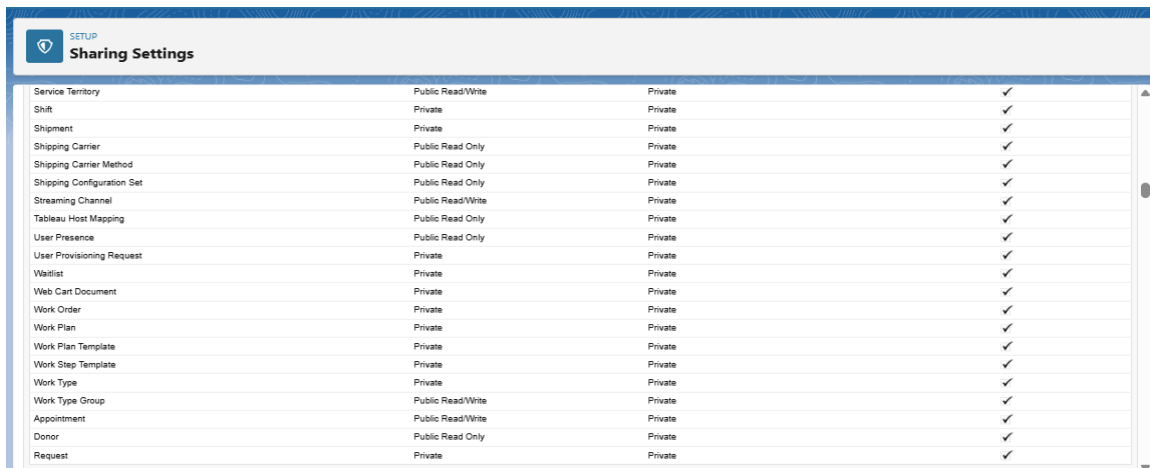
8. Permission Sets (Optional)

- **Setup** → **Quick Find** → **Permission Sets** → **New**.
- Created:
 - NGO_Manager_PS
 - Donor_Portal_Access_PS
- Saved (assignments not done yet).



9. Org-Wide Defaults (OWD)

- **Setup → Quick Find → Sharing Settings.**
- Edited OWD defaults:
 - Donor__c → Public Read
 - Request__c → Private
 - Appointment__c → Controlled by Parent
- **Save**

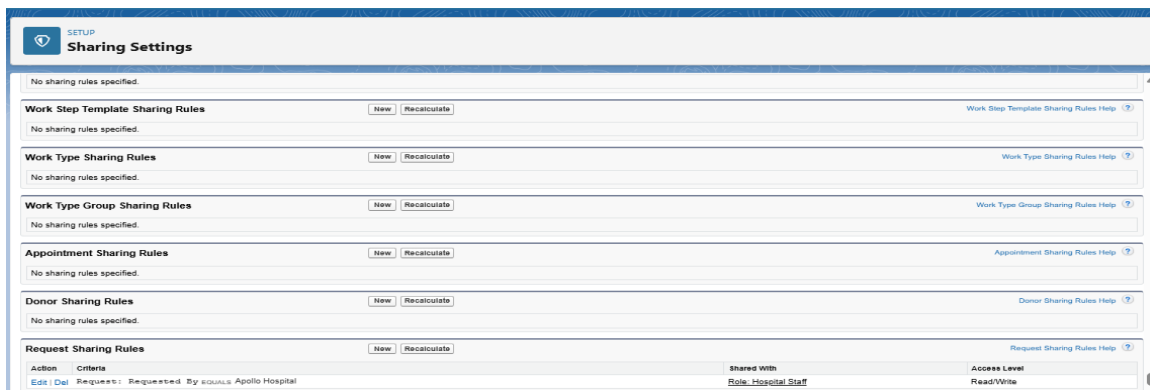


The screenshot shows the 'Sharing Settings' page in Salesforce Setup. It features a table with columns for 'Object', 'Default Sharing Method', and 'Controlled by Parent'. The table lists various objects and their corresponding sharing settings.

| Object | Default Sharing Method | Controlled by Parent |
|----------------------------|------------------------|----------------------|
| Service Territory | Public Read/Write | Private |
| Shift | Private | Private |
| Shipment | Private | Private |
| Shipping Carrier | Public Read Only | Private |
| Shipping Carrier Method | Public Read Only | Private |
| Shipping Configuration Set | Public Read Only | Private |
| Streaming Channel | Public Read/Write | Private |
| Tableau Host Mapping | Public Read Only | Private |
| User Presence | Public Read Only | Private |
| User Provisioning Request | Private | Private |
| Waitlist | Private | Private |
| Web Cart Document | Private | Private |
| Work Order | Private | Private |
| Work Plan | Private | Private |
| Work Plan Template | Private | Private |
| Work Step Template | Private | Private |
| Work Type | Private | Private |
| Work Type Group | Public Read/Write | Private |
| Appointment | Public Read/Write | Private |
| Donor | Public Read Only | Private |
| Request | Private | Private |

10. Sharing Rules

- **Setup → Quick Find → Sharing Settings → Scroll to Request Sharing Rules.**
- **Clicked New Sharing Rule.**
- Chose criteria: Requested by Apollo Hospital
- **Saved.**



The screenshot shows the 'Request Sharing Rules' section in the 'Sharing Settings' page. It displays a table with columns for 'Action', 'Criteria', 'Shared With', and 'Access Level'. The table shows a single rule created for 'Request: Requested By EQUALS Apollo Hospital'.

| Action | Criteria | Shared With | Access Level |
|--|--|----------------------|--------------|
| Edit Del | Request: Requested By EQUALS Apollo Hospital | Role: Hospital Staff | Read/Write |

11. Login Access Policies

- **Setup → Quick Find → Login Access Policies.**
- Checked **Administrators Can Log in as Any User.**
- Saved.

The screenshot shows the 'Login Access Policies' setup page in Salesforce. The page title is 'Login Access Policies' with a 'Help for this Page' link. Below the title, it says 'Control which support organizations your users can grant login access to.' There is a 'Manage Support Options' section with 'Save' and 'Cancel' buttons. Under 'Manage Support Options', there is a 'Setting' section with 'Enabled' and a checkbox for 'Administrators Can Log in as Any User' which is checked. Below this is a table with columns: 'Support Organization', 'Packages', 'Available to Users', and 'Available to Administrators Only'. The table has two rows: 'Salesforce.com Support' and 'Q Branch HQ Support'. The 'Salesforce.com Support' row has a blue dot in the 'Available to Users' column and an empty circle in the 'Available to Administrators Only' column. The 'Q Branch HQ Support' row has 'VideoFileViewer' in the 'Packages' column, 'Not Available' in the 'Available to Users' column, and a grey dot in the 'Available to Administrators Only' column. There are 'Save' and 'Cancel' buttons at the bottom of the table.

| Support Organization | Packages | Available to Users | Available to Administrators Only |
|------------------------|-----------------|----------------------------------|----------------------------------|
| Salesforce.com Support | | <input checked="" type="radio"/> | <input type="radio"/> |
| Q Branch HQ Support | VideoFileViewer | Not Available | <input type="radio"/> |

□ Outcome

At the end of Phase 2, I successfully configured:

- Company details (timezone, currency, languages).
- Business hours & fiscal year.
- Users, profiles, and roles.
- Security model with OWD & sharing rules.
- Admin login access for user testing.

Phase 3 Documentation – Data Modeling & Relationships

Goal of Phase 3

To design and implement the data model for the Hospital Donor Management System using Salesforce objects, fields, record types, layouts, and relationships. This ensures that all donor, request, and appointment data is properly structured, accessible, and secure.

1 Standard & Custom Objects

What was done:

- Reviewed **Standard Objects** (like User, Contact).
- Created **Custom Objects** specific to the project:
 - **Donor__c** → Stores donor details.
 - **Request__c** → Stores patient donation requests (blood/organ).
 - **Appointment__c** → Stores scheduled donation appointments.

☐ **Output:** Custom objects for Donor, Request, Appointment created.

2 Fields

What was done:

- Added necessary **Custom Fields** to capture data:

Donor__c Fields:

- Name, Gender, DOB, Phone, Email, Address, Blood Group, Organ Donor (Checkbox), Availability.

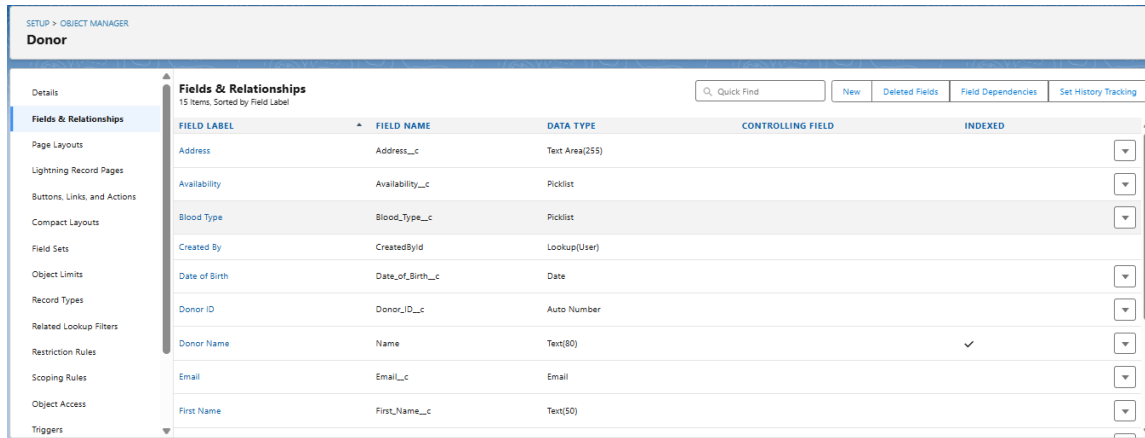
Request__c Fields:

- Request Type (Blood/Organ), Urgency, Status, Related Donor Lookup.

Appointment__c Fields:

- Appointment Date, Time, Notes, Donor Lookup, Request Lookup.

☐ **Output:** Custom fields created per object.



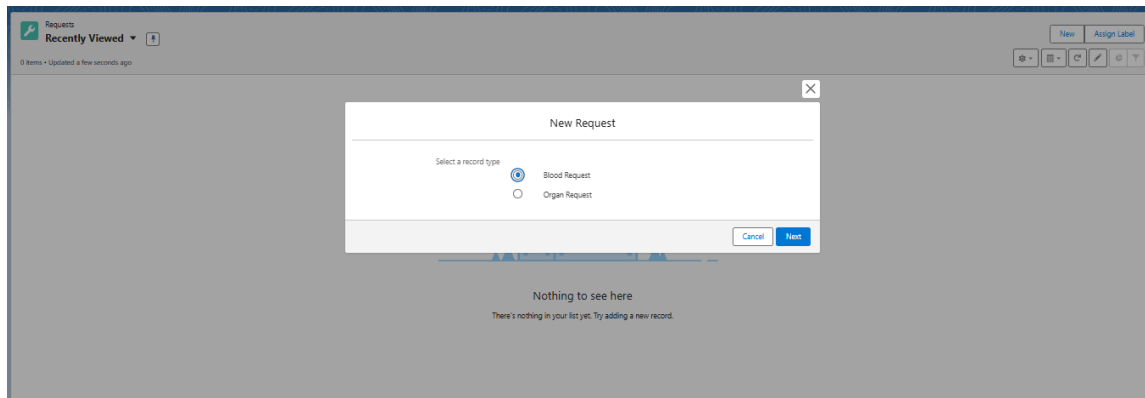
| FIELD LABEL | FIELD NAME | DATA TYPE | CONTROLLING FIELD | INDEXED |
|---------------|------------------|----------------|-------------------|-------------------------------------|
| Address | Address__c | Text Area(255) | | <input type="checkbox"/> |
| Availability | Availability__c | Picklist | | <input type="checkbox"/> |
| Blood Type | Blood_Type__c | Picklist | | <input type="checkbox"/> |
| Created By | CreatedById | Lookup(User) | | <input type="checkbox"/> |
| Date of Birth | Date_of_Birth__c | Date | | <input type="checkbox"/> |
| Donor ID | Donor_ID__c | Auto Number | | <input type="checkbox"/> |
| Donor Name | Name | Text(80) | | <input checked="" type="checkbox"/> |
| Email | Email__c | Email | | <input type="checkbox"/> |
| First Name | First_Name__c | Text(50) | | <input type="checkbox"/> |

3 Record Types

What was done:

- Applied **Record Types** only where needed:
 - **Donor__c**: Default record type only (all donors are similar).
 - **Request__c**: Two record types created → **Blood Request & Organ Request**.
 - **Appointment__c**: Default record type only.

☐ **Output:** Record Types designed to separate blood vs. organ requests.



4 Page Layouts

What was done:

- Created **different layouts for Manager & Staff** profiles.

Donor__c Layouts:

- **Manager Layout:** Full access to all donor fields (Name, Gender, DOB, Phone, Email, Availability, Organ Donor).
- **Staff Layout:** Limited view (hides sensitive fields like DOB, Email, Phone).

Request__c Layouts:

- **Manager Layout:** Full details including urgency & requested by.
- **Staff Layout:** Can create/edit but restricted from viewing/deleting manager-only fields.

Appointment__c Layouts:

- **Manager Layout:** Can see appointment notes.
- **Staff Layout:** Notes field hidden.

□ **Output:** Separate layouts for each profile with controlled visibility.

New Request: Blood Request

* = Required Information

Information

* Request Name

Owner

 Shaik Roshan Parveen

Request ID

Request Date

Blood Group Needed

Urgency Level

Donor

Request Status

Requested By

Cancel

Save & New

Save


New Request: Organ Request

* = Required Information

Information

* Request Name

Owner

 Shaik Roshan Parveen

Request ID

Request Date

Organ Needed

Urgency Level

Donor

Request Status

Requested By

Cancel

Save & New

Save

5 Compact Layouts

What was done:

- Created **Compact Layouts** for quick view in highlights panel.
- Example: Donor__c → Name, Blood Group, Availability, Phone.

□ **Output:** Compact layouts applied to display key fields in record previews.

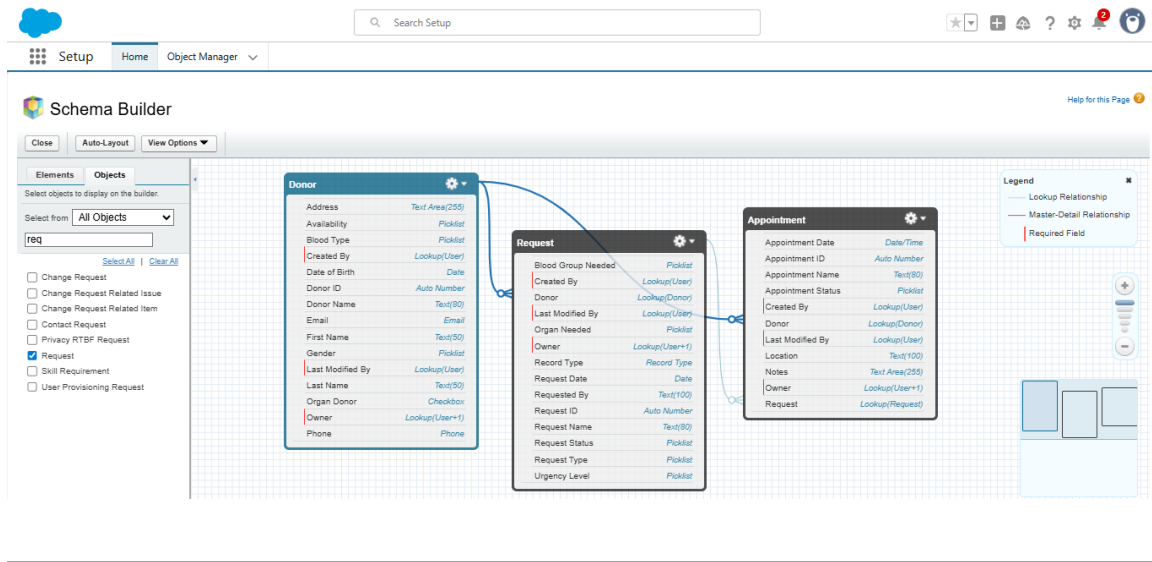
6 Schema Builder

What was done:

- Used **Schema Builder** to visualize objects and their relationships.
- Verified **Donor ↔ Request ↔ Appointment** connections.

- Ensured all relationships (lookup/master-detail) are properly mapped.

□ **Output:** Graphical data model available for reference.



7 Relationships

What was done:

- Applied correct relationship types:
- **Lookup Relationship:** Appointment__c → Donor__c (one donor can have many appointments).
- **Lookup Relationship:** Appointment__c → Request__c (one request can have many appointments).

□ **Output:** Clear relationship structure defined between objects.

Phase 4: Process Automation

Overview

In Phase 4, we implemented **Process Automation** in Salesforce to streamline operations for our *Blood & Organ Donation CRM*. This ensures data quality, reduces manual effort, and enables faster response during emergencies. We explored **Validation Rules, Flows, Email Alerts, Field Updates and Approval Processes**.

1. Validation Rules

Purpose: Ensure data accuracy and prevent invalid records.

Rules Implemented:

1. **Donor Minimum Age** — Donors must be at least 18 years old.
 - Formula: `TODAY() - Date_of_Birth__c < 6570`

The screenshot displays the 'New Donor' form in a Salesforce interface. The form is titled 'New Donor' and includes a search bar at the top. The form is divided into sections: 'Information' and 'New Section'. The 'Information' section contains fields for Donor Name (Rohit Sharma), Donor ID, First Name (Rohit), Last Name (Sharma), Gender (Male), Date of Birth (9/20/2023), Phone (9876543210), Email (shaikroshanparveen@gmail.com), and Address (vjayavada). The 'New Section' contains fields for Blood Type (O+) and Ability (available). A red error message box is displayed over the form, stating 'We hit a snag.' and 'Review the following fields: Date of Birth'. The error message also includes the text 'Donor must be at least 18 years old to register.' The form has buttons for 'Cancel', 'Save & New', and 'Save'.

2. **Appointment Date Check** — Appointment date/time cannot be in the past.

- Formula: `Appointment_Date__c < NOW()`

New Appointment

* = Required Information

Information

*Appointment Name

Urgent Meeting

Appointment ID

Appointment Date

Date

9/17/2025

Time

12:00 PM

Appointment date cannot be in the past.

Appointment Status

--None--

Notes

Owner

Shaik Roshan Parveen

Location

Vijayawada

Donor

hello

Request

Harry

We hit a snag.

Review the following fields

- [Appointment Date](#)

Cancel

Save & New

Save

3. **Blood Group Required** — Donor must have a blood group selected.

- Formula: `ISBLANK(TEXT(Blood_Type__c))`

Outcome: Only valid donor, appointment, and request data can be saved in the system.

* Donor Name
 Phool Kumari

Donor ID
 D-0003

First Name
 Phool

Last Name
 Kumari

Gender
 Female

Owner
 Shaik Roshan Parveen

Date of Birth
 9/12/2006

Phone
 9876543210

Email
 shaikroshanparveen@gmail.com

Address
 kadapa

New Section

Blood Type
 --None--

Availability
 Available

Blood Group must be selected.

Organ Donor
☐

Created By
 Shaik Roshan Parveen

Modified By
 Shaik Roshan Parveen, 9/20/2025, 9:08 PM

We hit a snag.

Review the following fields

- [Blood Type](#)

Cancel Save & New Save

2. Workflow Rules

Not implemented as standalone in Phase 4; replaced by more advanced Flow automations for better scalability and control.

3. Process Builder

Existing use cases replaced with Flow Builder for automating field updates and sending notifications.

4. Flow Builder

Purpose: Most powerful automation tool (Screen, Record-Triggered, Scheduled, Auto-launched).

Flows Implemented:

1. Record-Triggered Flow (Total Units Available)

- Trigger: On Blood_Request__c create/edit.
- Logic:
 - Get all donors with matching blood type.
 - Sum their Units_Available__c.
 - Update Request field Total_Units_Available__c.
- Condition: If Total Units \geq Required Units \rightarrow request is marked “Fulfillable.”

Outcome: Automatically calculates donor availability for each request.

2. Record-Triggered Flow (Appointment Fulfillment)

- **Trigger:** On Appointment__c create/edit.
- **Logic:**
 - Get the related Request__c record.
 - Check if Request_Status__c = In Progress and Donor__r.Availability__c = "Available".
 - If true \rightarrow
 - Send **Email Alert** (appointment confirmation).
 - Update the related Request__c.Request_Status__c = "Fulfilled".
- **Condition:** Only runs when Appointment is created or updated for a donor linked to an In Progress request.
- appointment is scheduled with an available donor.

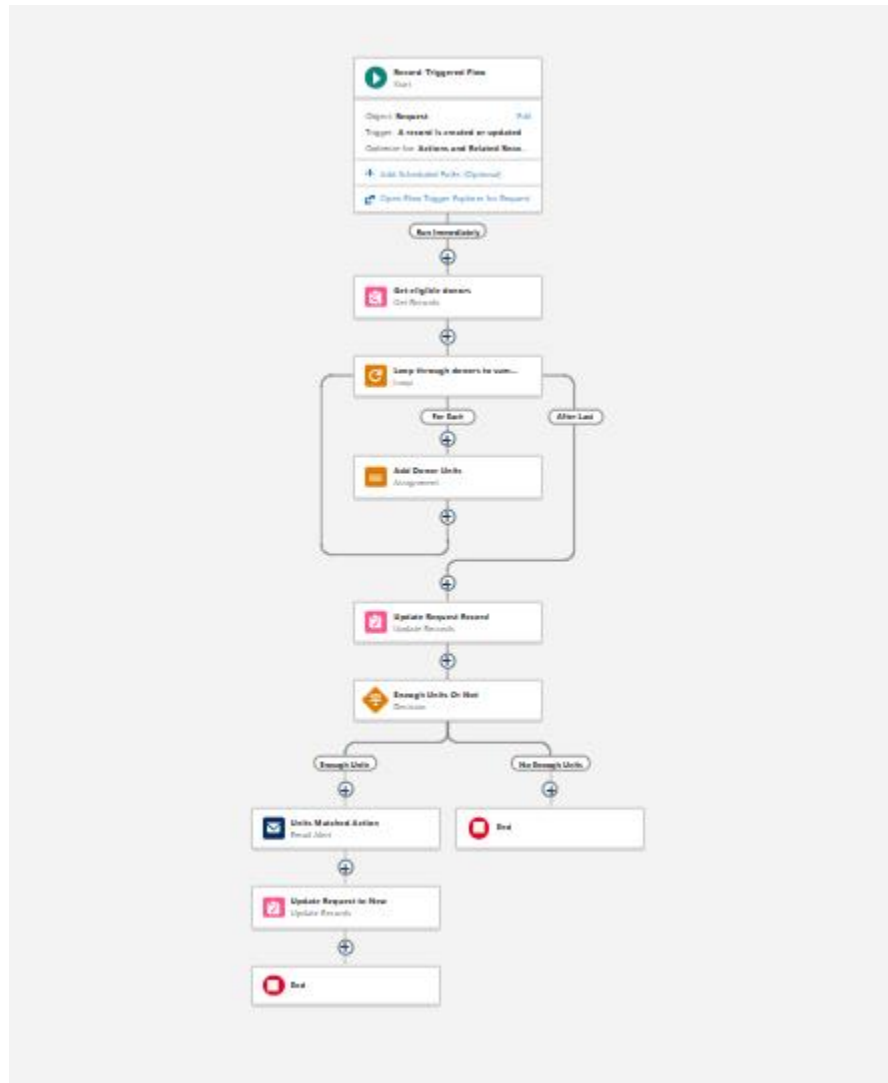


Fig: Total Units Available

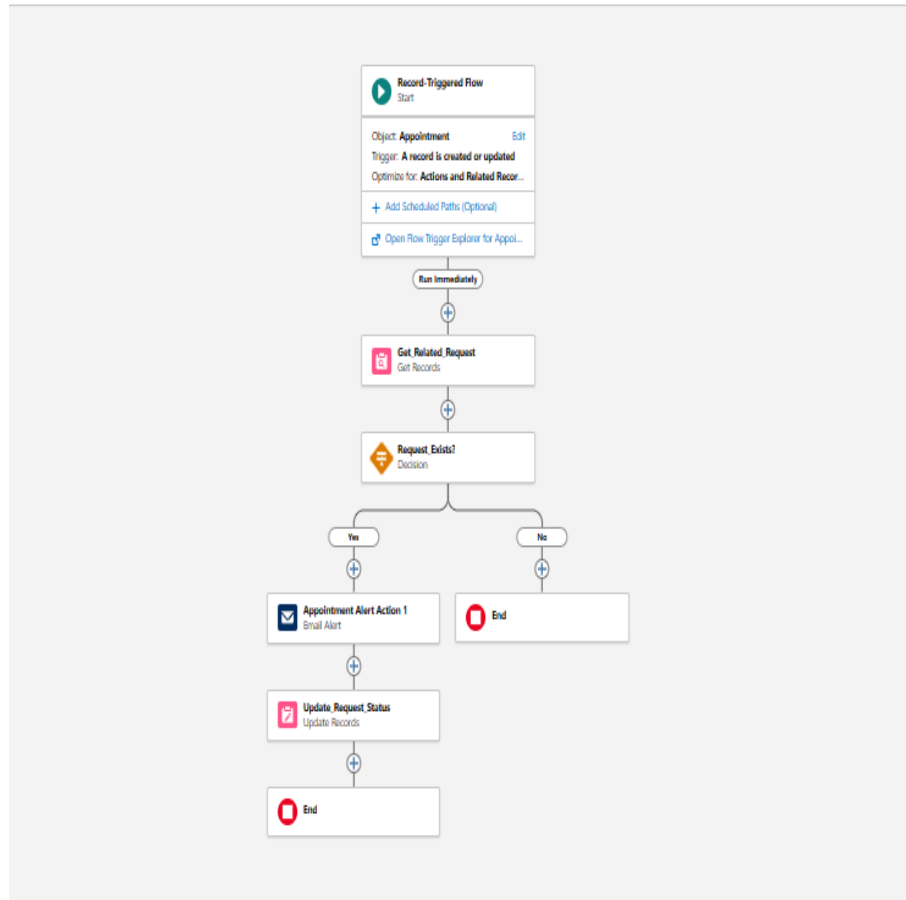


Fig: Appointment Fulfillment

5. Email Alerts

- Configured **Classic Email Templates** (New Request Notification).
- Appointment Confirmation Email, Blood Request Fulfillable Notification, Blood Request Approved.

Outcome: Consistent, automated communication.



Shaik Roshan Parveen via fqrulm4qvtikn.gi-ayvxua1.can98.bnc.salesforce.com
to hospital.staff@yourdomain.com, sysadmin2@yourdomain.com, hospital.manager@yourdomain.com, me ▾

8:48 PM (2 hours ago) ☆ ☺ ↶ ⋮

Why is this message in spam? This message is similar to messages that were identified as spam in the past.

Report not spam



Hello ,

The blood request re8 can now be fulfilled.

Blood Type: O+
Units Required: 5
Total Units Available: 20
Please take necessary action.

Thanks,
MediLink Org

Appointment Scheduled for Request:JMJ hospital Spam x



Shaik Roshan Parveen via yda7yHkicdv9.gi-ayvxua1.can98.bnc.salesforce.com
to hospital.staff@yourdomain.com, sysadmin2@yourdomain.com, hospital.manager@yourdomain.com, me ▾

9:08 PM (2 hours ago) ☆ ☺ ↶ ⋮

Why is this message in spam? This message is similar to messages that were identified as spam in the past.

Report not spam



Hello Coordinator,

An appointment has been scheduled for donor .
The request dear is now Fulfilled.

Hello {!Request__c.Requested_By__c},

Good news! Your blood request has been ****approved****.

Request Details:

- Request ID: {!Request__c.Request_Name__c}
- Units Approved: {!Request__c.Units_Required__c}
- Blood Group: {!Request__c.Blood_Group_Needed__c}

You will be contacted shortly regarding the next steps.

Thank you,
Medilink Org

6. Field Updates

- In Process Builder, `Status__c` automatically updates to **Closed** once fulfilled.
- Helps maintain request lifecycle without manual intervention.

The screenshot shows a 'Request' form for 'R-0059'. At the top, there are five tabs: 'Urgency Level', 'Request Status Fulfilled', 'Requested By', 'Blood Group Needed O+', and 'Organ Needed'. Below these is a 'Details' section with two columns of fields. The left column includes 'Request Name req1', 'Request ID R-0059', 'Blood Group Needed O+', 'Donor', 'Units Required 5', 'Total Units Available 0', 'Coordinator', and 'Created By Shaik Roshan Parveen, 9/23/2025, 4:03 AM'. The right column includes 'Owner Shaik Roshan Parveen', 'Request Date', 'Urgency Level', 'Request Status Fulfilled', 'Requested By', and 'Last Modified By Shaik Roshan Parveen, 9/23/2025, 8:33 AM'. Each field has a small edit icon to its right.

9. Approval Process

- Designed an **Approval Process** for **Blood Donation Requests**.
- Request goes through: Coordinator(or any user assigned).
- Once approved, request is marked as “Approved for Blood.”

Outcome: Sensitive requests follow compliance and multi-level approval.

The screenshot shows a 'Request Approval' process instance. At the top, there is a 'Process Instance Step' section with a red status icon and the text 'Request Approval Approved'. Below this is a table with four columns: 'Submitter Shaik Roshan Parveen', 'Date Submitted Sep 21, 2025', 'Actual Approver Shaik Roshan Parveen', and 'Assigned To Shaik Roshan Parveen'. Below the table is a 'Details' section with a 'New Approval' link. To the right of the details section is a 'No Comments' button.

Approval Request Spam x



Shaik Roshan Parveen via k475n87lm97j5u-gl-ayvxzua1.can98.bnc.salesforce.com
to me ▾

10:27 AM (13 hours ago)

Why is this message in spam? This message is similar to messages that were identified as spam in the past.

Report not spam

Shaik Roshan Parveen has requested your approval for the following item: <https://orgfarm-ac8e3ceb9-dev-ed.develop.my.salesforce.com/p/process/ProcessInstanceWorkitemWizardStageMana>

Please click this link to approve or reject this record.

Thank you,
Salesforce

Request Approved Spam x



Shaik Roshan Parveen via 7wixikgqxhukksam.uedrgn0irvohxryr.6fk9o.gl-ayvxzua1.can98.bnc.salesforce.com
to me ▾

Why is this message in spam? This message is similar to messages that were identified as spam in the past.

Report not spam

Request test is now In Progress.

Conclusion

Phase 4 introduced **intelligent automation** into the CRM system:

- Data is validated before saving.
- Coordinators and volunteers are notified instantly.
- Status updates and donor availability calculations happen automatically.
- Sensitive requests follow a structured approval chain.

This automation reduces errors, saves time, and ensures **faster life-saving response** in critical donation scenarios.

Phase 5: Apex Programming (Developer)

Overview

In Phase 5, we implemented **Apex Programming** to extend automation beyond declarative tools. Apex was used for custom logic to process blood requests, manage donor availability, and schedule automated jobs. This ensured **real-time updates, data consistency, and scalability**.

1. Apex Classes & Objects

Implemented Classes:

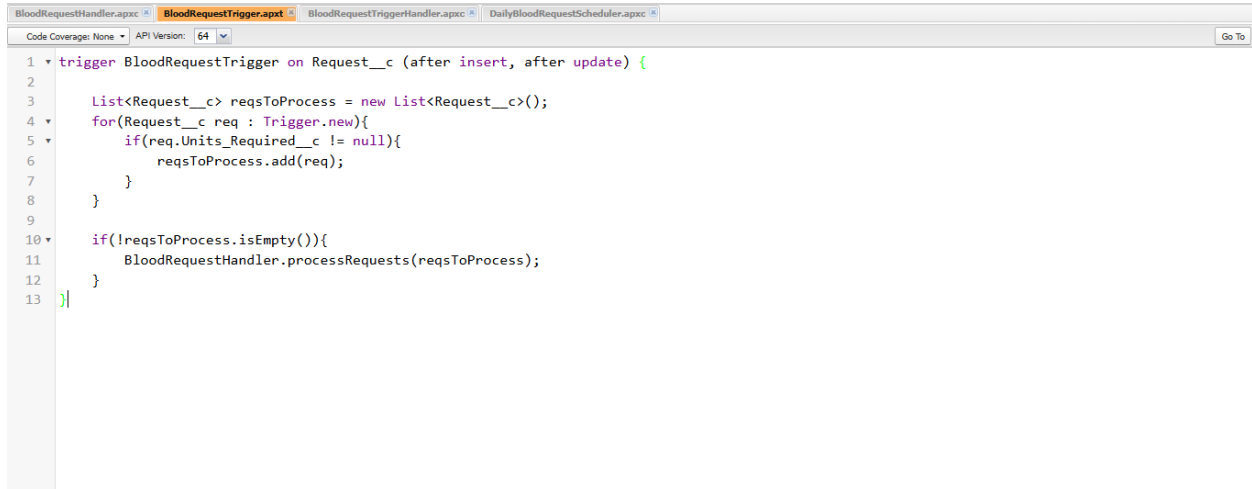
- **BloodRequestHandler.cls**
 - Contains core logic for processing blood requests.
 - Prevents recursion using a static flag.
 - Calculates total units available by blood type.
 - Deducts donor units when a request is fulfillable.
 - Updates request status (New → In Progress).
- **DailyBloodRequestScheduler.cls**
 - Implements `Schedulable` interface.
 - Runs daily to re-check pending requests.
 - Updates donor availability and request status in bulk.



```
1 public class BloodRequestHandler {
2
3     // Static variable to prevent recursion
4     public static Boolean isProcessing = false;
5
6     public static void processRequests(List<Request__c> reqList) {
7
8         // Prevent recursion
9         if(isProcessing) return;
10        isProcessing = true;
11
12        if(reqList == null || reqList.isEmpty()) return;
13
14        // Fetch fresh Request records from DB
15        List<Request__c> freshRequests = [
16            SELECT Id, Name, Blood_Group_Needed__c, Units_Required__c, Request_Status__c, Total_Units_Available__c
17            FROM Request__c
18            WHERE Id IN :reqList
19        ];
20
21        // Collect all blood types needed
22        Set<String> bloodTypes = new Set<String>();
```

2. Apex Triggers (after insert, after update)

- **BloodRequestTrigger.trigger**
 - Fires on Request__c after insert and update.
 - Calls BloodRequestHandler.processRequests() to execute the business logic.
 - Ensures requests are processed immediately upon creation/update.



```
1 trigger BloodRequestTrigger on Request__c (after insert, after update) {
2
3     List<Request__c> reqsToProcess = new List<Request__c>();
4     for(Request__c req : Trigger.new){
5         if(req.Units_Required__c != null){
6             reqsToProcess.add(req);
7         }
8     }
9
10    if(!reqsToProcess.isEmpty()){
11        BloodRequestHandler.processRequests(reqsToProcess);
12    }
13 }
```

3. Trigger Design Pattern

- Followed **one trigger per object** principle.
- Trigger kept lean by delegating logic to BloodRequestHandler.cls.
- Added a static Boolean variable to prevent recursive updates.

4. SOQL

- Used SOQL queries to fetch required records:
 - Fetch **Requests** with needed fields.
 - Fetch **Donors** filtered by blood type and availability.
- Example:

```
List<Donor__c> allDonors = [
    SELECT Id, Name, Units_Available__c, Blood_Type__c
    FROM Donor__c
    WHERE Blood_Type__c IN :bloodTypes
    AND Units_Available__c > 0
    ORDER BY Units_Available__c DESC
];
```

5. Collections: List, Set, Map

- **List:** Stored donors to be updated in bulk.
 - **Set:** Collected distinct blood types required by pending requests.
 - **Map:** Grouped donors by blood type (`Map<String, List<Donor__c>>`).
-

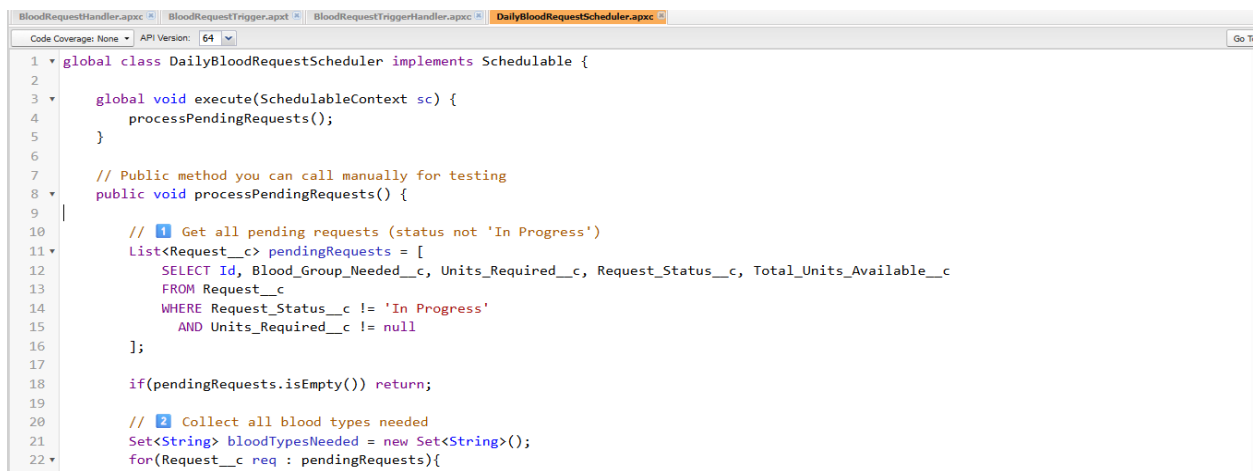
6. Control Statements

- **If-Else:** Checked whether requests had enough donors.
 - **For Loops:** Iterated through donors to sum units and deduct availability.
 - **Break Statements:** Stopped looping once required units were deducted.
-

7. Scheduled Apex

- `DailyBloodRequestScheduler.cls` implemented `Schedulable`.
- Runs daily to:
 - Find pending requests.
 - Recalculate donor availability.
 - Update request statuses accordingly.

Outcome: Automation runs in background without manual intervention.



```
1 global class DailyBloodRequestScheduler implements Schedulable {
2
3     global void execute(SchedulableContext sc) {
4         processPendingRequests();
5     }
6
7     // Public method you can call manually for testing
8     public void processPendingRequests() {
9
10        // 1 Get all pending requests (status not 'In Progress')
11        List<Request__c> pendingRequests = [
12            SELECT Id, Blood_Group_Needed__c, Units_Required__c, Request_Status__c, Total_Units_Available__c
13            FROM Request__c
14            WHERE Request_Status__c != 'In Progress'
15            AND Units_Required__c != null
16        ];
17
18        if(pendingRequests.isEmpty()) return;
19
20        // 2 Collect all blood types needed
21        Set<String> bloodTypesNeeded = new Set<String>();
22        for(Request__c req : pendingRequests){
```

□ Conclusion

Phase 5 introduced **Apex-driven automation** into the project. With handler classes, triggers, SQL queries, collections, and a scheduled job, the system now:

- Automatically matches donors with requests.
- Deducts donor units in real time.
- Updates request statuses (New → In Progress).
- Re-checks pending requests daily.

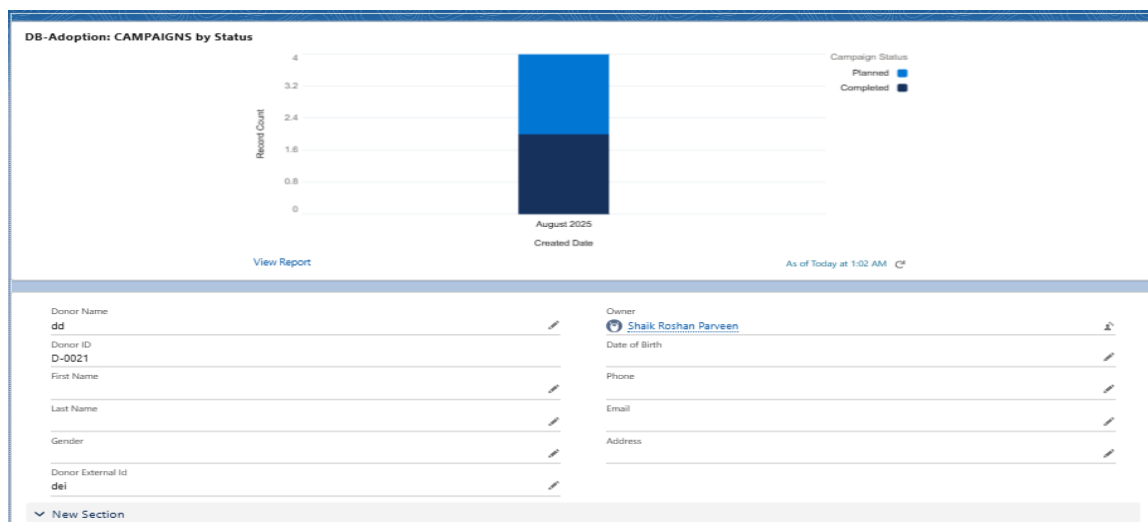
This ensures the CRM can **scale**, remain **data-consistent**, and handle **complex donor-request matching** beyond point-and-click automation.

Phase 6: User Interface Development

In this phase, the focus was on designing and customizing the Salesforce user interface to make it intuitive, user-friendly, and aligned with the requirements of the Blood Request Management system. The following steps were undertaken:

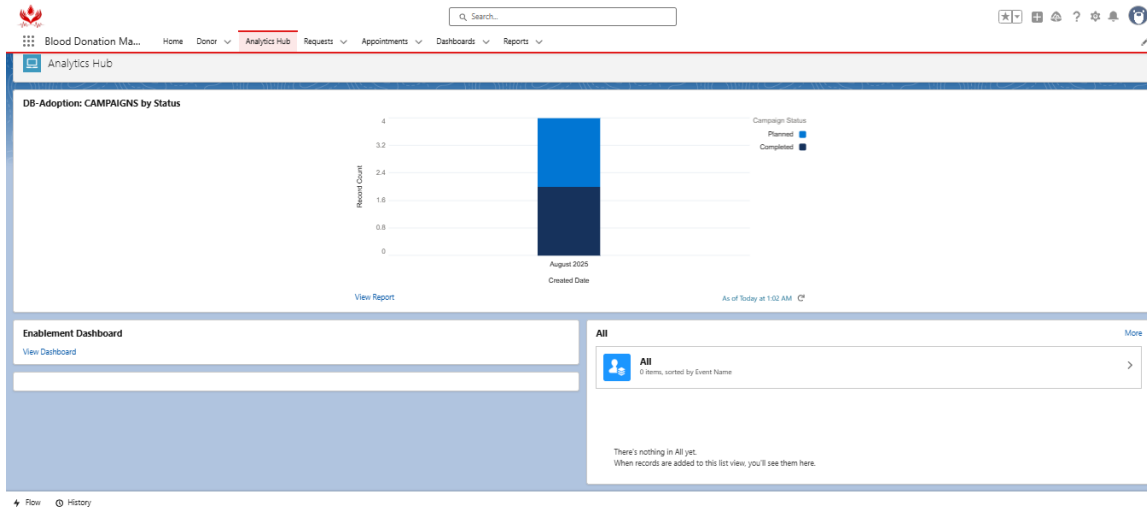
1. Lightning App Builder

- Used **Lightning App Builder** to create custom pages and apps for the system.
- Developed a dedicated **Blood Request Management app** to provide users with quick access to blood requests, donors, and appointments.
- Configured **custom components** and **standard components** (like Lists, Charts, and Rich Text) to display relevant information dynamically.



2. Record Pages

- Customized **Record Pages** for Request__c, Donor__c, and Appointment__c.
- Ensured that fields, sections, and related lists are arranged logically for ease of data entry and review.
- Configured **conditional visibility** for components based on field values (e.g., only show appointment details if a request is in progress).



3. Tabs

- Created custom **Tabs** for Blood Requests, Donors, and Appointments in the Lightning app.
- Enabled easy navigation for users, ensuring that frequently used objects were accessible from the app navigation bar.

SETUP

Tabs

Custom Tabs

You can create new custom tabs to extend Salesforce functionality or to build new application functionality.

Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Salesforce window. Visualforce tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app. Lightning Page tabs allow you to add Lightning Pages to Lightning Experience and the mobile app.

Custom Object Tabs

New What Is This?

| Action | Label | Tab Style | Description |
|------------|--------------|-----------|-------------|
| Edit Del | Appointments | Hand | |
| Edit Del | Donor | Star | |
| Edit Del | Requests | Wrench | |

Web Tabs

New What Is This?

No Web Tabs have been defined

Visualforce Tabs

New What Is This?

No Visualforce Tabs have been defined

Lightning Component Tabs

New What Is This?

No Lightning component tabs have been defined

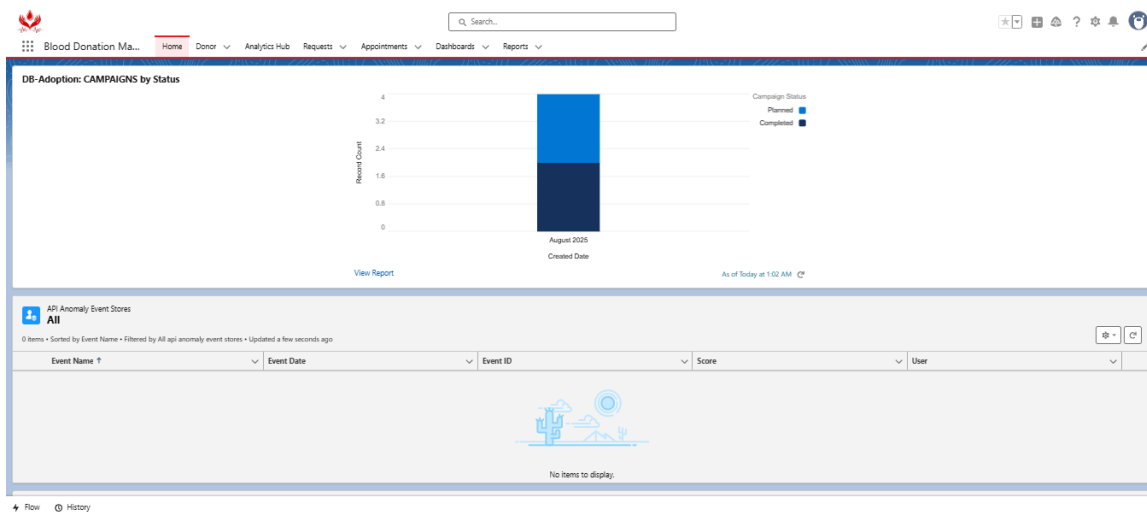
Lightning Page Tabs

New What Is This?

| Action | Label | Tab Style | Description |
|------------|---------------|-----------|----------------------------------|
| Edit Del | Analytics Hub | Laptop | Created by Lightning App Builder |

4. Home Page Layouts

- Designed a **custom Home Page Layout** to provide users with an overview of important metrics.
- Added **dashboards, charts, and recent records lists** to display critical data such as total pending blood requests, donors with available units, and upcoming appointments.
- Ensured a responsive and user-friendly interface that supports quick decision-making.



Phase 7: Integration & External Access

1. Named Credentials

- Store external system credentials (username/password, OAuth, etc.) securely.
- Simplifies authentication for API callouts.
- Avoids hardcoding credentials in Apex or flows.

2. External Services

- Connect Salesforce to external REST APIs declaratively.
- Automatically generate Apex actions from OpenAPI (Swagger) schemas.
- Can use in Flows, Process Builder, or Apex without manual coding.

3. Web Services (REST/SOAP)

- **REST/SOAP APIs** allow Salesforce to communicate with external systems.
 - Use **Apex callouts** for sending/receiving data.
 - Can expose Salesforce objects/data to external systems via **Apex REST/SOAP services**.
-

4. Callouts

- Make HTTP requests from Salesforce to external services (GET, POST, PUT, DELETE).
 - Often used for real-time integrations with other apps or APIs.
 - Can be synchronous (immediate response) or asynchronous (future methods, queues).
-

5. Platform Events

- Event-driven architecture within Salesforce.
 - Publish and subscribe to events between Salesforce and external systems.
 - Useful for real-time notifications, integrations, or workflow triggers.
-

6. Change Data Capture (CDC)

- Track changes (create, update, delete, undelete) in Salesforce objects.
 - External systems or processes can subscribe to these changes.
 - Enables real-time sync with external databases or apps.
-

7. Salesforce Connect

- Access external data in real-time without storing it in Salesforce.
 - Use **External Objects** to map tables from other systems.
 - Useful for large datasets or integrating with ERP/legacy systems.
-

8. API Limits

- Salesforce enforces limits on API calls (REST, SOAP, Bulk API, etc.).
 - Important to monitor and optimize integration to avoid hitting limits.
-

9. OAuth & Authentication

- Securely authenticate external apps and users with Salesforce.
 - Supports OAuth 2.0 flows, JWT, and connected apps.
 - Often used with Named Credentials and External Services.
-

10. Remote Site Settings

- Add external URLs to allow Apex callouts to these sites.
 - Prevents unauthorized access to external systems.
 - Required for any callout to an endpoint outside Salesforce.
-

□ Summary:

This phase is all about **connecting Salesforce to other systems, sending/receiving data, tracking events, and maintaining secure authentication**. You can integrate in real-time, schedule data syncs, or expose Salesforce data externally.

Phase 8: Data Management & Deployment

In this phase, the focus was on managing data efficiently, ensuring data integrity, and deploying changes across environments. The following tools and functionalities were used:

1. Data Import Wizard

- **Purpose:** To import records such as Donor__c and Request__c into Salesforce.
- **Steps Taken:**
 1. Accessed **Setup → Data → Data Import Wizard**.
 2. Selected the object to import (e.g., Donor, Request).

3. Uploaded the **prepared CSV file** containing donor and request data.
 4. Mapped CSV fields to Salesforce fields carefully to ensure accuracy.
 5. Reviewed and started the import process.
- **Outcome:** Successfully imported initial dataset for donors and requests, creating a foundation for testing and further operations.

Getting close...

Choose data | Edit mapping | Start Import

Import your Data into Salesforce
You can import up to 50,000 records at a time.

What kind of data are you importing?

Standard objects | Custom objects

Appointments >

Donations >

Donor ✓

Requests >

What do you want to do?

Add new records ✓

Match by: -None-

Which User field in your file designates record owners? External ID

Trigger workflow rules and processes? ☐

Update existing records >

Where is your data located?

Drag CSV file here to upload

CSV

File: Choose File donor.csv ✓

Character Code: ISO-8859-1 (General US & Western European, ISO-LATIN-1) ✓

Values Separated By: Comma ✓

Cancel Previous Next

SETUP Bulk Data Load Jobs

Bulk Data Load Job
750gL00000E678Q

View the details of a bulk data load job.

[Back to List: Bulk Data Load Jobs](#)

Bulk Data Load Job Detail

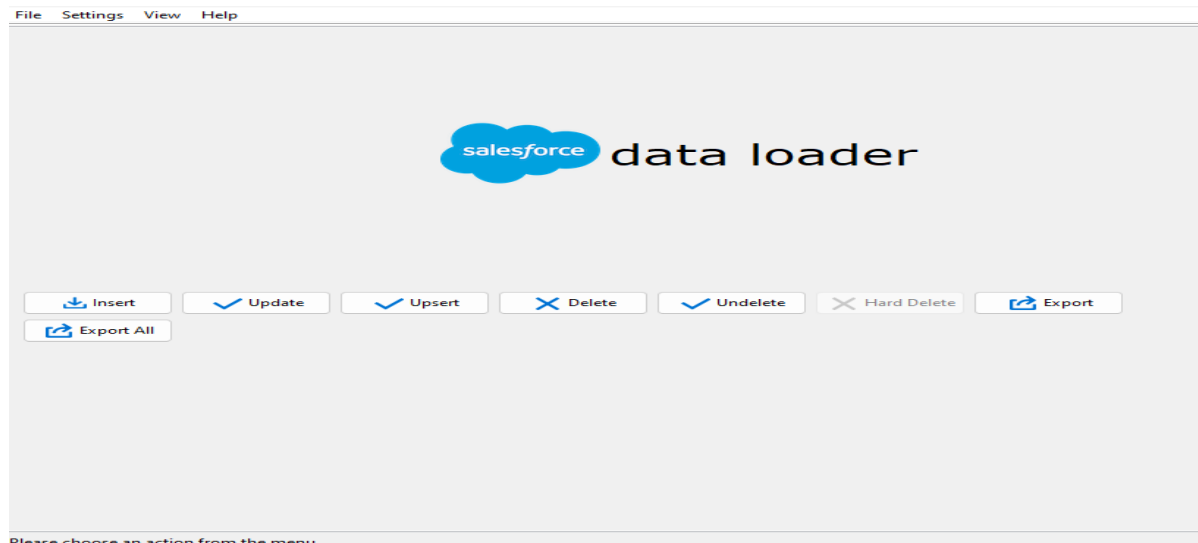
| Job ID | Job Type | Bulk V1 | Status | Closed |
|------------------------------------|---------------------|---------|--------------------------------|--------|
| 750gL00000E678Q | Operation | Insert | Progress | 149 |
| Submitted By: Shaik Roshan Parveen | | | Total Processing Time (m) | |
| Start Time: 9/25/2025, 2:57 AM PST | Queued Batches | 0 | API Active Processing Time (m) | 58 |
| End Time: 9/25/2025, 2:57 AM PST | In Progress Batches | 0 | Apex Processing Time (m) | 1 |
| Time to Complete (h:m:ss): 00:00 | Completed Batches | 1 | | |
| Object: Donor | Failed Batches | 0 | | |
| External ID Field | Progress | 100% | | |
| Content Type: CSV | Records Processed | 5 | | |
| Concurrency Mode: Parallel | Records Failed | 5 | | |
| API Version: 54.0 | Retries | 0 | | |

Reload

2. Data Loader

- **Purpose:** To perform large-scale data operations like insert, update, export, and delete.
- **Steps Taken:**
 1. Installed **Salesforce Data Loader** on the local machine.
 2. Logged in using Salesforce credentials.
 3. Selected the operation type (Insert, Update, Upsert).
 4. Chose the object (e.g., Donor__c, Request__c).
 5. Uploaded CSV file and mapped the fields.
 6. Executed the operation and reviewed success/failure files.

- **Outcome:** Efficiently handled bulk data operations, ensuring records were accurately updated and created.



3. Duplicate Rules

- **Purpose:** To prevent duplicate records from being created and maintain data integrity.
- **Steps Taken:**
 1. Accessed **Setup** → **Duplicate Management** → **Duplicate Rules**.
 2. Created rules for Donor__c and Request__c objects.
 3. Configured **matching rules** based on unique identifiers like Donor_External_Id__c and Request_External_Id__c.
 4. Set the actions to **Alert or Block** duplicates during record creation or updates.
- **Outcome:** Ensured that duplicate donors and requests could not be created, maintaining clean and accurate data.

SETUP

Matching Rules

Edit Rule Donor Duplicate Rule matching rule

Help for this Page

Save Cancel

Rule Details

Object

Donor

Rule Name

Donor Duplicate Rule match

Unique Name

Donor_Duplicate_Rule_mai

Description

Matching Criteria

Tell the rule which fields to compare and how.

Field

Email

Donor External Id

--None--

--None--

--None--

Matching Method

Exact

Exact

Exact

Exact

Exact

Match Blank Fields

☐

☐

☐

☐

AND

AND

AND

AND

Add Filter Logic...

Save Cancel

SETUP

Duplicate Rules

New Duplicate Rule

Donor

Save Save & New Cancel

Duplicate Rule Edit

Rule Name

Donor Duplicate Rule

Description

Prevents duplicate donor records based on email or external ID.

Object

Donor

Record-Level Security

Enforce sharing rules

Bypass sharing rules

Actions

Specify what happens when a user tries to save a duplicate record.

Action On Create

Allow

Alert

Report

Action On Edit

Allow

Alert

Report

Alert Text

Use one of these records?

Matching Rules

Define how duplicate records are identified.

Compare Donor With

Donor

Matching Rule

--Select a Matching Rule--

Donor ID

Date of Birth

First Name

Phone

Last Name

Email

Gender

--None--

Address

Donor External Id

dei

New Section

Blood

--f

Units

Ability

None--

in Donor

Similar Records Exist

This record looks like an existing record. Make sure to check any potential duplicate records before saving.

View Duplicates

Cancel

Save & New

Save

4. Data Export & Backup

1. **Purpose:** To back up Salesforce data for security, audit, and recovery purposes.
2. **Steps Taken:**
 1. Accessed **Setup** → **Data** → **Data Export**.
 2. Selected objects to export (Donor__c, Request__c, Appointment__c).
 3. Chose **manual or scheduled export** (weekly or monthly).
 4. Downloaded the exported **.zip files** containing CSV files of all selected records.
3. **Outcome:** Maintained regular backups of all Salesforce data to prevent loss and support future recovery needs.

The screenshot displays the Salesforce 'Data Export' setup page. At the top, there's a header with the 'SETUP' icon and 'Data Export' text. Below this, the 'Monthly Export Service' section is visible, including a help link. A yellow box indicates 'Next scheduled export: None'. There are 'Export Now' and 'Schedule Export' buttons. Below these, a table shows export details: 'Scheduled By' (Shah Roshan Parveen), 'Schedule Date' (9/25/2025), and 'Export File Encoding' (ISO-8859-1). At the bottom, a table lists the export file with columns for 'Action', 'File Name', and 'File Size'. The file 'WE_00DgL00000AYvXZUA1_1.ZIP' is listed with a size of 3.5K and a 'download' link.

| Action | File Name | File Size |
|--------------------------|-----------------------------|-----------|
| download | WE_00DgL00000AYvXZUA1_1.ZIP | 3.5K |

Result of Phase 8:

This phase ensured **accurate data import, bulk data handling, prevention of duplicates, and reliable backup mechanisms** for the Blood Request Management system. These processes enhanced data integrity, security, and operational efficiency.

Phase 9: Reporting, Dashboards & Security Review

This phase focused on creating **reports, dashboards, and implementing security settings** in Salesforce for the Blood Request Management System.

1. Reports

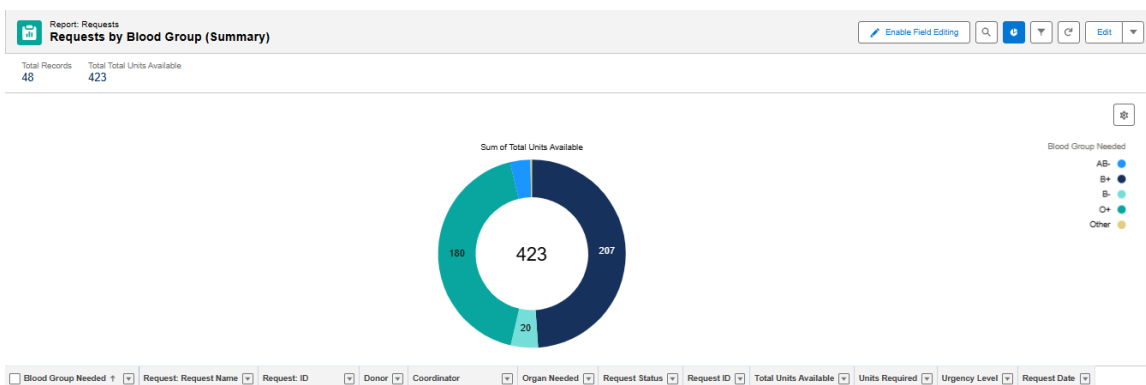
Reports were created to track and analyze requests, donors, and appointments.

- **Tabular Report:**
 - **Steps:** Setup → Reports → New Report → Select Object → Add Columns & Filters → Save & Run.
 - **Use:** Simple list view of records like all active blood requests.

The screenshot shows a Salesforce report interface for 'New Donor Report'. It includes a table with columns: Donor: Donor Name, First Name, Last Name, Email, Phone, Blood Type, Units Available, Availability, Date of Birth, and Donor External Id. The table contains 10 rows of data, with a total of 10 records and 95 units available.

| | Donor: Donor Name | First Name | Last Name | Email | Phone | Blood Type | Units Available | Availability | Date of Birth | Donor External Id |
|----|-------------------|------------|-----------|-------|------------|------------|-----------------|--------------|---------------|-------------------|
| 1 | dd | - | - | - | - | AB- | 50 | - | - | - |
| 2 | Donor 4 | - | - | - | - | AB- | 15 | - | - | - |
| 3 | Donor 7 | - | - | - | - | B- | 15 | - | - | - |
| 4 | dd | - | - | - | - | B- | 10 | - | - | del |
| 5 | jake | jake | - | - | - | B- | 4 | - | - | - |
| 6 | Donor 3 | Donar | 3 | - | - | B+ | 1 | - | - | - |
| 7 | Manu | - | - | - | - | B+ | 0 | - | - | - |
| 8 | Donor 6 | - | - | - | - | O+ | 0 | - | - | - |
| 9 | Ramu | - | - | - | 9999999999 | O+ | - | - | - | - |
| 10 | hello | - | - | - | - | - | - | - | - | - |
| 11 | | | | | | | 95 | | | |

- **Summary Report:**
 - **Steps:** Setup → Reports → New Report → Add Grouping (e.g., Blood Group) → Add Summary fields (Sum of Units).
 - **Use:** Show total units required per blood group.

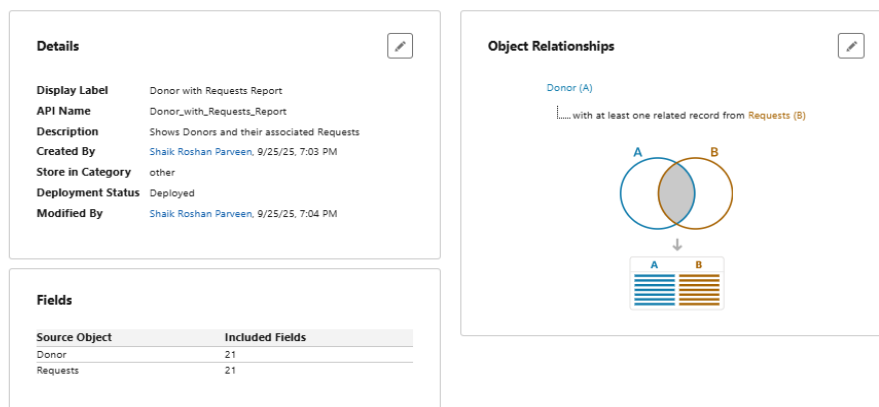


Outcome: Reports helped in monitoring blood request status and donor availability.

2. Report Types

Custom report types were created for combining data from multiple objects.

- **Steps:**
Setup → Report Types → New Custom Report Type → Select Primary Object (Request__c) → Add Related Object (Donor__c) → Deploy → Save.



Details

Display Label: Donor with Requests Report
API Name: Donor_with_Requests_Report
Description: Shows Donors and their associated Requests
Created By: Shaik Roshan Parveen, 9/25/25, 7:03 PM
Store in Category: other
Deployment Status: Deployed
Modified By: Shaik Roshan Parveen, 9/25/25, 7:04 PM

Fields

| Source Object | Included Fields |
|---------------|-----------------|
| Donor | Z1 |
| Requests | Z1 |

Object Relationships

Donor (A)
... with at least one related record from Requests (B)

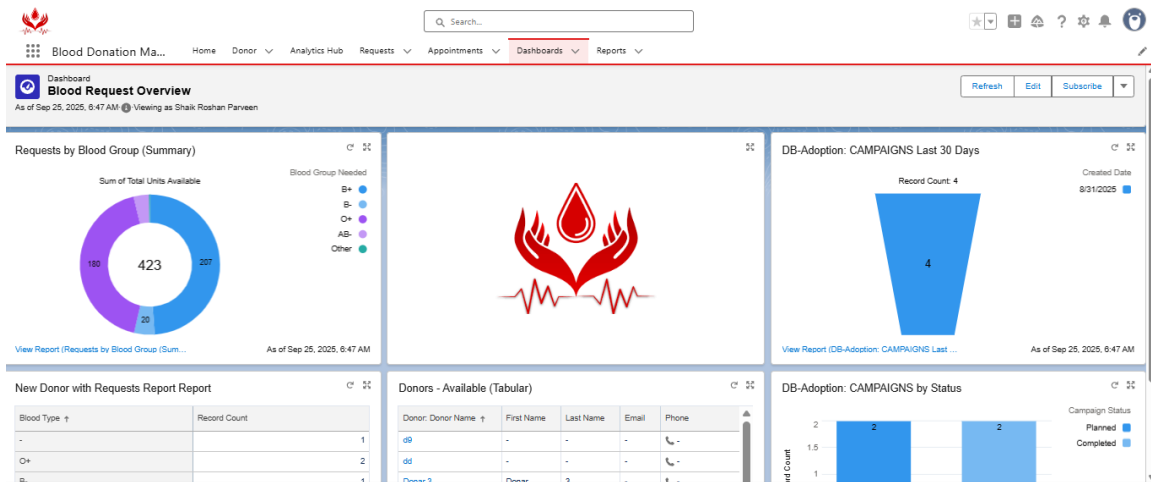
The diagram shows two overlapping circles, A (Donor) and B (Requests). Below the circles, a downward arrow points to a table icon with columns labeled A and B, representing the combined data view.

Outcome: Enabled cross-object reporting (e.g., Requests linked with Donors).

3. Dashboards

Dashboards were created to provide a visual overview of system data.

- **Steps:**
Setup → Dashboards → New Dashboard → Add Components → Choose Source Report → Select Chart Type → Save.

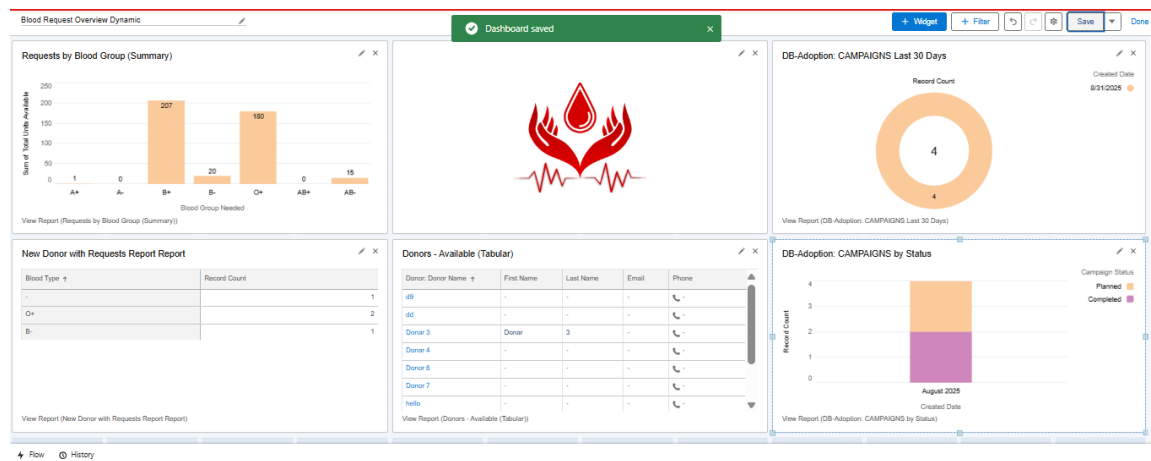


Outcome: Graphical view of requests by blood group, donor availability, and fulfilled requests.

4. Dynamic Dashboards

Dynamic dashboards were enabled to show data according to the logged-in user.

- **Steps:**
In Dashboard Edit → View Dashboard As → Run as Logged-in User → Save.

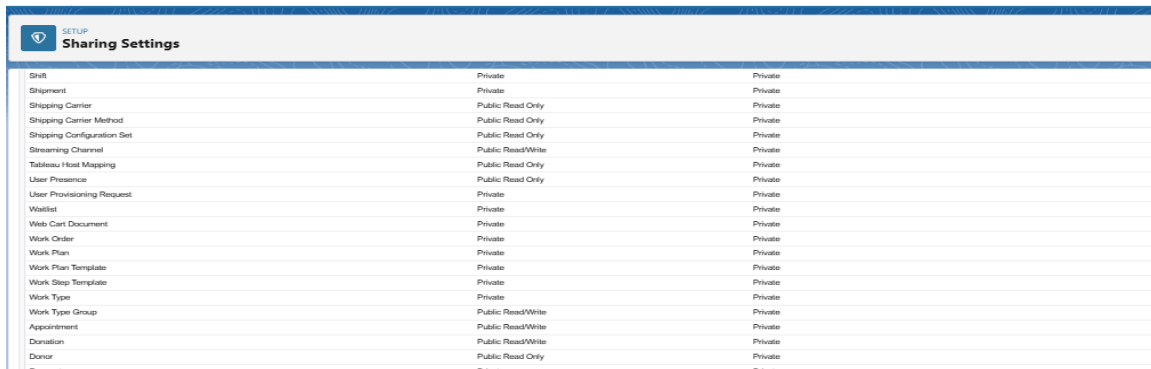


Outcome: Each coordinator/user sees only the data relevant to them.

5. Sharing Settings

Sharing rules and Organization-Wide Defaults (OWD) were set to manage record access.

- **Steps:**
Setup → Sharing Settings → Configure OWD for each object → Add Sharing Rules → Save.



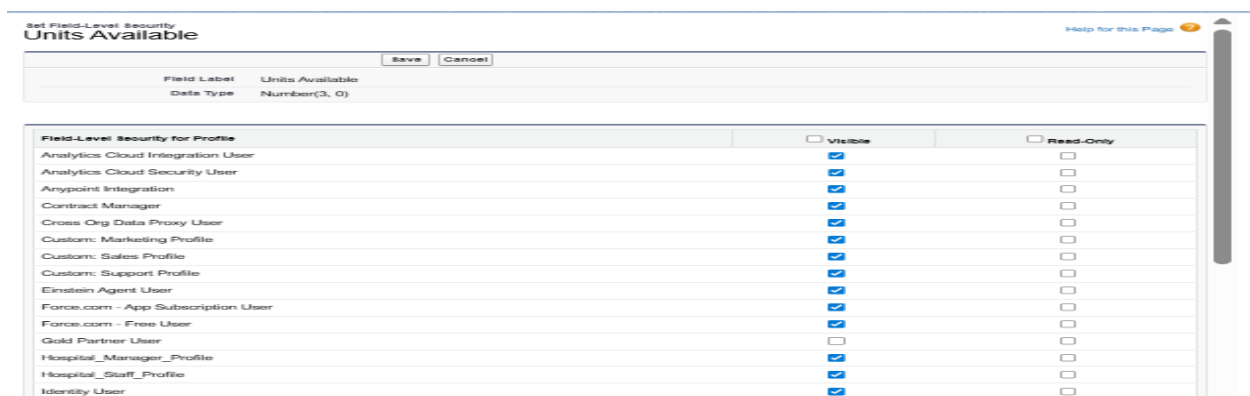
| Object | Sharing Model | Sharing Method |
|----------------------------|-------------------|----------------|
| Shift | Private | Private |
| Shipment | Private | Private |
| Shipping Carrier | Public Read Only | Private |
| Shipping Carrier Method | Public Read Only | Private |
| Shipping Configuration Set | Public Read Only | Private |
| Streaming Channel | Public Read/Write | Private |
| Tableau Host Mapping | Public Read Only | Private |
| User Presence | Public Read Only | Private |
| User Provisioning Request | Private | Private |
| Waolist | Private | Private |
| Web Cart Document | Private | Private |
| Work Order | Private | Private |
| Work Plan | Private | Private |
| Work Plan Template | Private | Private |
| Work Step Template | Private | Private |
| Work Type | Private | Private |
| Work Type Group | Public Read/Write | Private |
| Appointment | Public Read/Write | Private |
| Donation | Public Read/Write | Private |
| Donor | Public Read Only | Private |
| Event | Public Read Only | Private |

Outcome: Controlled data access while maintaining collaboration.

6. Field Level Security

Field visibility was controlled for different profiles.

- **Steps:**
Setup → Object Manager → Select Object → Fields & Relationships → Select Field → Set Field-Level Security → Save.



| Field Label | Units Available | Save | Cancel |
|-------------|-----------------|--------------|--------|
| Field Label | Data Type | Number(3, 0) | |

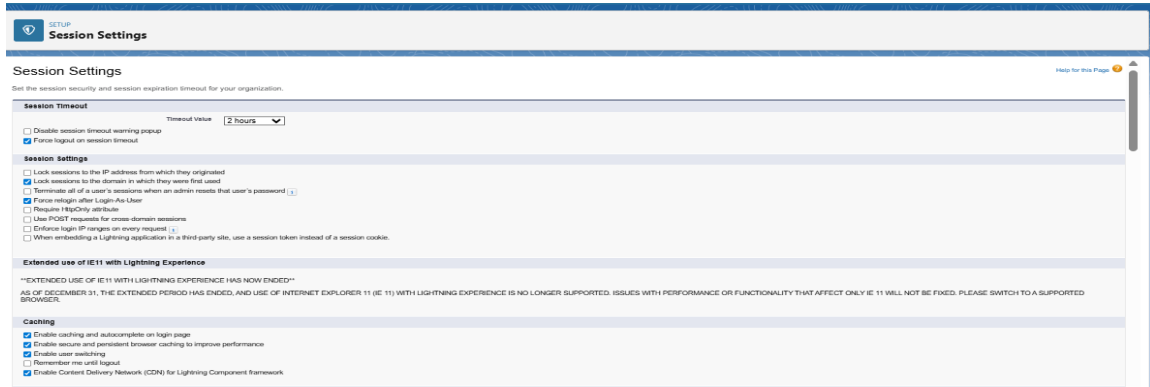
| Field-Level Security for Profile | Visible | Read-Only |
|-----------------------------------|-------------------------------------|--------------------------|
| Analytics Cloud Integration User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Analytics Cloud Security User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Anypoint Integration | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Contract Manager | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Cross Org Data Proxy User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Custom: Marketing Profile | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Custom: Sales Profile | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Custom: Support Profile | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Einstein Agent User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Force.com - App Subscription User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Force.com - Free User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Gold Partner User | <input type="checkbox"/> | <input type="checkbox"/> |
| Hospital_Manager_Profile | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Hospital_Staff_Profile | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Identity User | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

Outcome: Restricted sensitive fields (like Units_Available__c) to authorized users only.

7. Session Settings

Session security policies were applied.

- **Steps:**
Setup → Session Settings → Configure Session Timeout, Security Levels → Save.



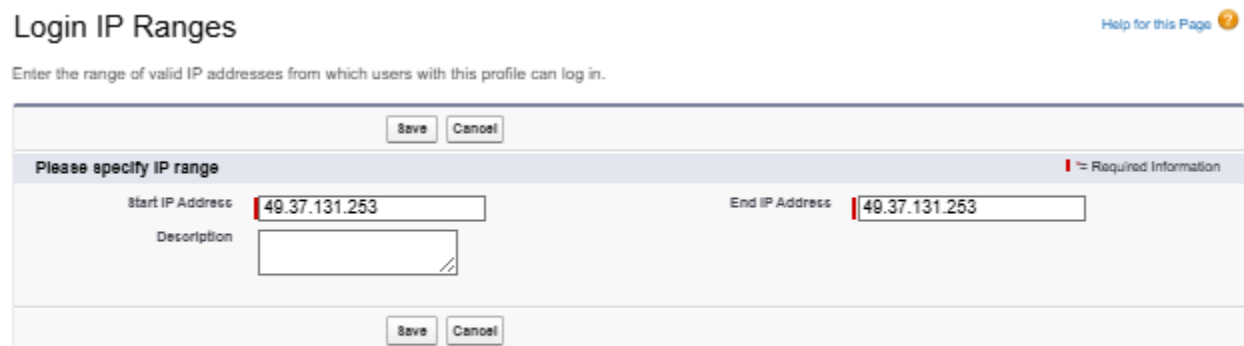
The screenshot shows the 'Session Settings' page in a web application. The page has a header with 'SETUP' and 'Session Settings'. Below the header, there's a section titled 'Session Timeout' with a 'Timeout Value' dropdown set to '2 hours'. There are checkboxes for 'Disable session timeout warning popup' (unchecked) and 'Force login on session timeout' (checked). The 'Session Settings' section includes checkboxes for 'Lock sessions to the IP address from which they originated' (unchecked), 'Lock sessions to the domain in which they were first used' (checked), 'Terminate all of a user's sessions when an admin resets that user's password' (unchecked), 'Force relogin after Login As User' (checked), 'Require HttpOnly attribute' (unchecked), 'Use POST requests for cross domain sessions' (unchecked), 'Enforce login IP ranges on every request' (unchecked), and 'When embedding a Lightning application in a third party site, use a session token instead of a session cookie' (unchecked). The 'Extended use of IE11 with Lightning Experience' section contains a warning message about the end of extended support for IE11. The 'Caching' section has checkboxes for 'Enable caching and autocomplete on login page' (checked), 'Enable secure and persistent browser caching to improve performance' (checked), 'Enable user switching' (checked), 'Remember me until logout' (unchecked), and 'Enable Content Delivery Network (CDN) for Lightning Component framework' (checked).

Outcome: Prevented unauthorized access and ensured session security.

8. Login IP Ranges

Login IP ranges were set to restrict access to trusted networks.

- **Steps:**
Setup → Profiles → Select Profile → Login IP Ranges → Add New Range → Save.



The screenshot shows the 'Login IP Ranges' page. It has a header with 'Login IP Ranges' and a 'Help for this Page' link. Below the header, there's a text prompt: 'Enter the range of valid IP addresses from which users with this profile can log in.' There are 'Save' and 'Cancel' buttons at the top. The main section is titled 'Please specify IP range' and includes a legend 'I = Required Information'. It has two text input fields: 'Start IP Address' with the value '49.37.131.253' and 'End IP Address' with the value '49.37.131.253'. There is also a 'Description' text area. At the bottom, there are 'Save' and 'Cancel' buttons.

Outcome: Restricted users from logging in outside the defined IP ranges.

□ Overall Result

- Reports & dashboards provide **clear visibility of blood requests and donor availability**.
- Dynamic dashboards allow **role-based insights**.
- Security settings (Sharing, FLS, Session, IP Ranges) ensure **data protection and controlled access**.

Phase 10: Final Presentation & Demo Day

1. Pitch Presentation

• Project Name & Introduction

- *Blood Bank Management System*
- Developed in **Salesforce Lightning Experience**
- Purpose: Streamline blood donation and request processes

• Problem Statement

- Manual tracking of blood requests is error-prone
- Difficulty in matching donors with requests
- Delays in scheduling appointments and notifying stakeholders

• Project Goals

- Automate donor-request matching
- Calculate total available units for each request
- Send automatic appointment notifications
- Maintain accurate request and donor records
- Generate reports and dashboards for monitoring

• Key Features

- **Record-Triggered Flows:** Auto-calculate available blood units
- **Scheduled Flows:** Daily processing of pending requests
- **Appointment Automation:** Email alerts to donors & coordinators
- **Reports & Dashboards:** Visualize requests, donors, and units
- **Security & Access Control:** Sharing rules, FLS, session/IP restrictions

- **Impact / Benefits**

- Faster response to blood requests
- Reduced manual errors
- Improved donor coordination
- Real-time insights for admins and coordinators

- **Technology Used**

- Salesforce Lightning, Flows (Record-Triggered & Scheduled)
- Apex classes for advanced automation
- Email Alerts and Notifications
- Data Import Wizard & Data Loader for bulk data management

2.Demo Walkthrough

Demo Video Link:

https://drive.google.com/file/d/1Mn6kROTsgjhrzqxYkAqq9uCsDHVey1z_/view?usp=sharing