Project Overview

Project Title

Hospital Appointment and Bed Management CRM System

This project will be developed **incrementally and systematically**, following **all ten standard Salesforce project phases**, aligned with real-world enterprise implementation practices.

Phase 1: Problem Understanding and Industry Analysis

> *This phase focuses on analysis and planning, without system configuration, reflecting real project kickoff methodology.*

---

1. Business Problem Statement

Healthcare institutions face multiple operational challenges, including:

- High patient inflow
- Limited doctor availability
- Restricted bed capacity
- Risk of appointment overlap
- Requirement for approval of high-cost treatments
- Lack of centralized reporting

Challenges Without a CRM System:

- Duplicate or conflicting appointments
- Inefficient bed allocation
- Manual record keeping
- Limited visibility for management

Proposed Solution:

Implementation of a **Salesforce-based CRM system** to streamline hospital operations through automation, validation, and real-time reporting.

---

2. Stakeholder Identification

Stakeholder Role	Responsibility
System Administrator	Salesforce configuration and maintenance
Receptionist	Patient registration and appointment scheduling
Doctor	Schedule review and patient consultation
Hospital Manager	Approval of high-cost treatments
Patient	Receives notifications and confirmations

Stakeholder identification supports later design of roles, profiles, and security.

---

3. Business Process Flow

Patient visits or contacts hospital

↓

Receptionist verifies doctor availability

↓

Appointment is scheduled

↓

If treatment cost exceeds ₹20,000 → Manager approval required

↓

Appointment confirmation email is sent

↓

Bed allocation performed (if admission required)

---

4. Industry-Specific Requirements and Salesforce Mapping

Healthcare Requirement	**Salesforce Feature Utilized**
Prevent overlapping appointments	Validation Rules, Apex Logic
Manage limited bed availability	Custom Objects and Tracking
Approval of high-value treatments	Approval Processes
Automated notifications	Flows and Email Alerts
Management reporting	Standard and Custom Reports

---

5. Justification for Salesforce Platform

Salesforce is selected due to its:

- Centralized and secure data management
- Low-code and no-code automation capabilities
- Robust role-based access control
- Scalability for healthcare operations
- Industry recognition as an enterprise CRM platform

Phase 1 Completion Summary

Phase 1 successfully establishes:

- Clear problem definition
- Stakeholder understanding
- Business process visualization
- Technology alignment with industry needs

This mirrors the **initial analysis phase followed by professional Salesforce consultants** in real implementations.

Project: Hospital Appointment & Bed Management CRM

Phase 2 — Org Setup and Configuration

Objective:
Prepare the Salesforce organization to reflect a real-world hospital company setup.

Step 2.1 — Open Setup (Critical Step)

1. Locate the **top-right corner** of the Salesforce interface
2. Click the **Gear icon**
3. Select **Setup**

You are now inside the Salesforce administrative environment.

Step 2.2 — Company Information (Hospital Identity)

1. In **Setup**, locate the **Quick Find** search box on the left panel

Enter:
Company Information

2.
3. Click **Company Information**

Verify or Update the Following:

Company Name
CityCare Hospital

-

Time Zone
(GMT+05:30) India Standard Time

-

Default Currency
INR

-

Click **Save** if the fields are editable.

This configuration impacts reports, approvals, and time-based automation.

Step 2.3 — Fiscal Year (Reporting Foundation)

In **Quick Find**, type:
Fiscal Year

1.
2. Click **Fiscal Year**
3. Select **Standard Fiscal Year**
4. Set the **Start Month** to **January**
5. Click **Save**

This setting is required for revenue tracking and dashboards.

Step 2.4 — Business Hours (Hospital Working Time)

In **Quick Find**, type:
Business Hours

1.
2. Click **Business Hours**
3. Click **New**

Configuration Details:

Name
Hospital Working Hours

-
- **Time Zone:** India Standard Time (IST)
- **Working Days:**
  - Monday to Saturday: 9:00 AM – 6:00 PM
  - Sunday: Not selected
- **Active:** Enabled

Click **Save**.

Step 2.5 — Holidays (No Approvals on These Days)

In **Quick Find**, type:
Holidays

1.
2. Click **Holidays**
3. Click **New**

Example Configuration:

Holiday Name
Republic Day

-
- **Date:** 26 January
- **Recurring:** Enabled

Click **Save**.

Approvals and automation will not execute on configured holidays.

Step 2.6 — Users (Hospital Personnel)

User Roles Overview:

User Role	Responsibility
Administrator	System configuration
Receptionist	Appointment creation
Manager	Case and record approvals

Create Receptionist User

In **Quick Find**, type:
Users

1.
2. Click **Users**
3. Click **New User**

User Details:

- **First Name:** Reception
- **Last Name:** Staff
- **Alias:** recp
- **Email:** Your email address (Salesforce allows reuse)

Username
receptionist.hospital@sfdev.com

-
- **Profile:** Standard User
- **Role:** Leave blank for now
- **Active:** Enabled

Click **Save**.

Step 2.7 — Profiles (Access Control)

Key Concept:

- **Profile defines what a user can do in Salesforce**

Profiles used in this project:

- **Standard User:** Receptionist
- **System Administrator:** Administrator

No profile customization is required at this stage.

Step 2.8 — Roles (Data Visibility)

In **Quick Find**, type:
Roles

1.
2. Click **Roles**

3. Select **Set Up Roles**

Create the Following Role Hierarchy:

Hospital Manager
└── Receptionist

Steps:

- Add the role **Hospital Manager**
- Under it, add **Receptionist**

Click **Save**.

Managers automatically gain visibility into subordinate records.

Step 2.9 — Org-Wide Defaults (Security Baseline)

In **Quick Find**, type:
Sharing Settings

1.
2. Scroll to **Org-Wide Defaults**

Set the following:

- **Contact:** Public Read Only
- Custom objects will be configured later

Click **Save**.

Step 2.10 — Login Hours (Security Control)

1. Navigate to **Profiles**
2. Open **Standard User**
3. Click **Login Hours**
4. Configure:
   - Monday to Saturday: 9:00 AM – 6:00 PM

Click **Save**.

This restricts login access outside working hours.

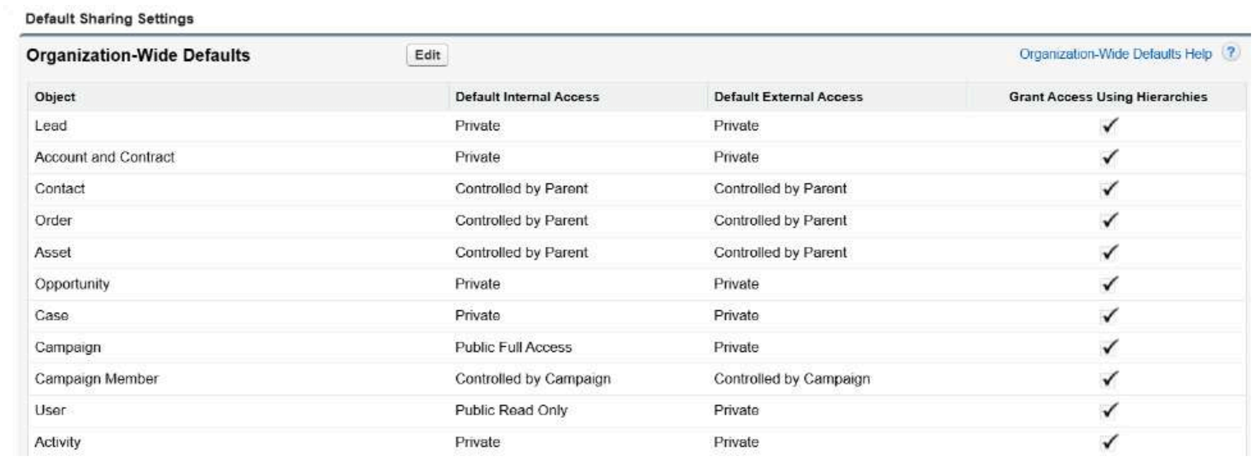
Phase 2 Completion Summary

Phase 2 is now complete.

Key Outcomes:

- Salesforce setup fundamentals
- Company-level configuration
- User, profile, and role management
- Core security controls
- Realistic hospital organization structure

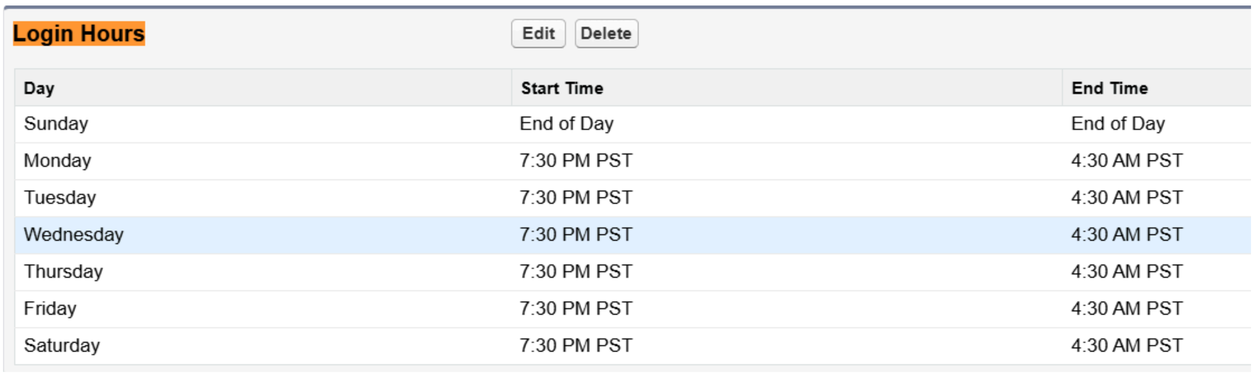
You have now progressed beyond the beginner level.



Default Sharing Settings

Organization-Wide Defaults

Object	Default Internal Access	Default External Access	Grant Access Using Hierarchies
Lead	Private	Private	✓
Account and Contract	Private	Private	✓
Contact	Controlled by Parent	Controlled by Parent	✓
Order	Controlled by Parent	Controlled by Parent	✓
Asset	Controlled by Parent	Controlled by Parent	✓
Opportunity	Private	Private	✓
Case	Private	Private	✓
Campaign	Public Full Access	Private	✓
Campaign Member	Controlled by Campaign	Controlled by Campaign	✓
User	Public Read Only	Private	✓
Activity	Private	Private	✓

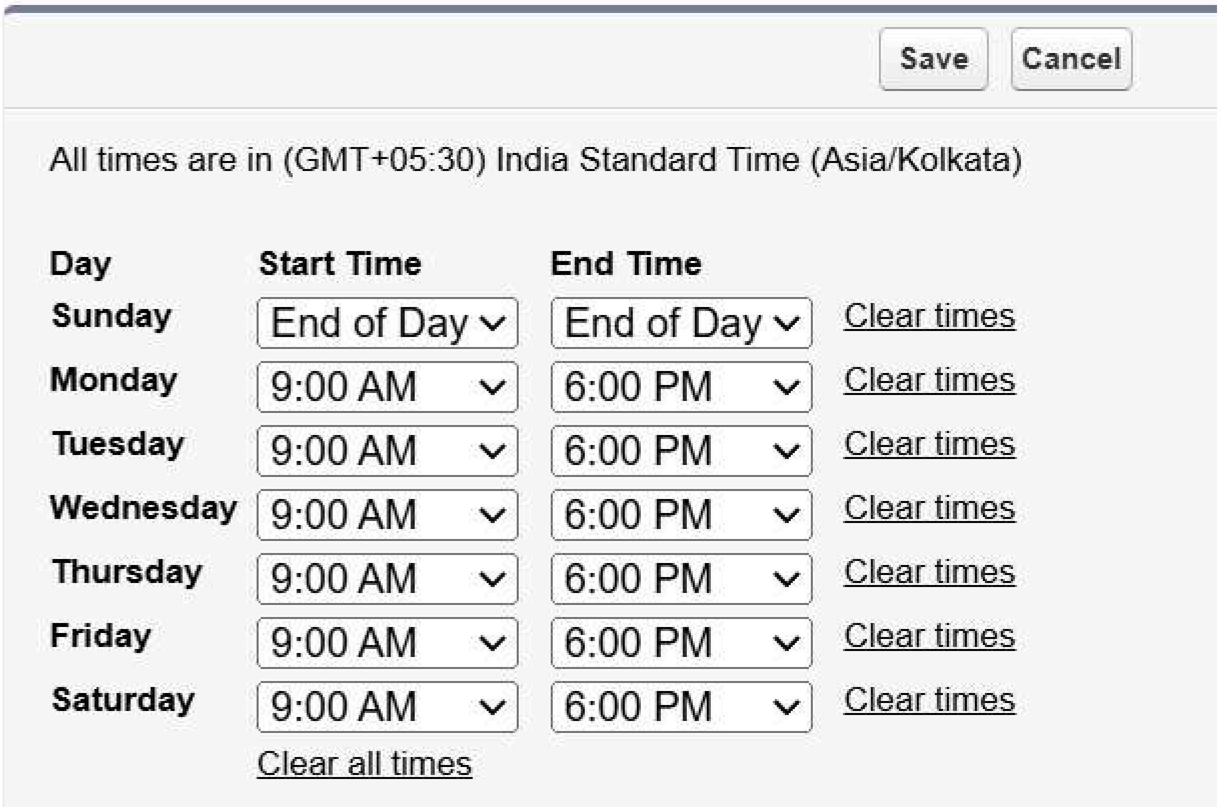


Login Hours

Day	Start Time	End Time
Sunday	End of Day	End of Day
Monday	7:30 PM PST	4:30 AM PST
Tuesday	7:30 PM PST	4:30 AM PST
Wednesday	7:30 PM PST	4:30 AM PST
Thursday	7:30 PM PST	4:30 AM PST
Friday	7:30 PM PST	4:30 AM PST
Saturday	7:30 PM PST	4:30 AM PST

Login Hours

Select the days and hours that users with this profile are allowed to log in.
times even for users in different time zones.

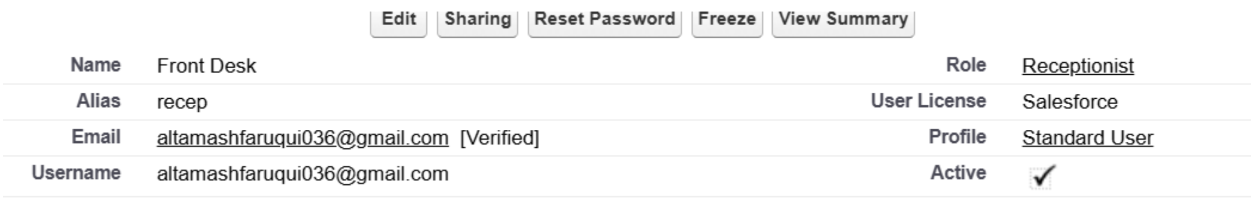


Save | Cancel

All times are in (GMT+05:30) India Standard Time (Asia/Kolkata)

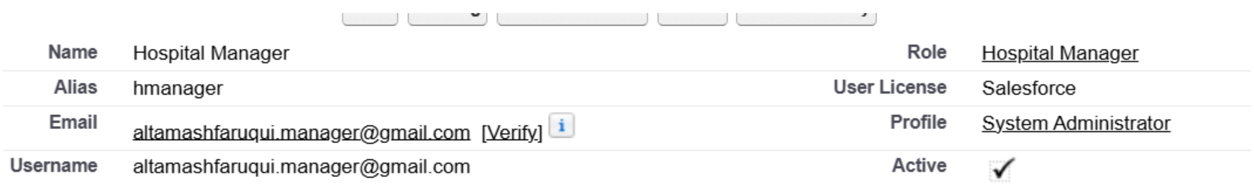
Day	Start Time	End Time	
Sunday	End of Day	End of Day	Clear times
Monday	9:00 AM	6:00 PM	Clear times
Tuesday	9:00 AM	6:00 PM	Clear times
Wednesday	9:00 AM	6:00 PM	Clear times
Thursday	9:00 AM	6:00 PM	Clear times
Friday	9:00 AM	6:00 PM	Clear times
Saturday	9:00 AM	6:00 PM	Clear times

Clear all times

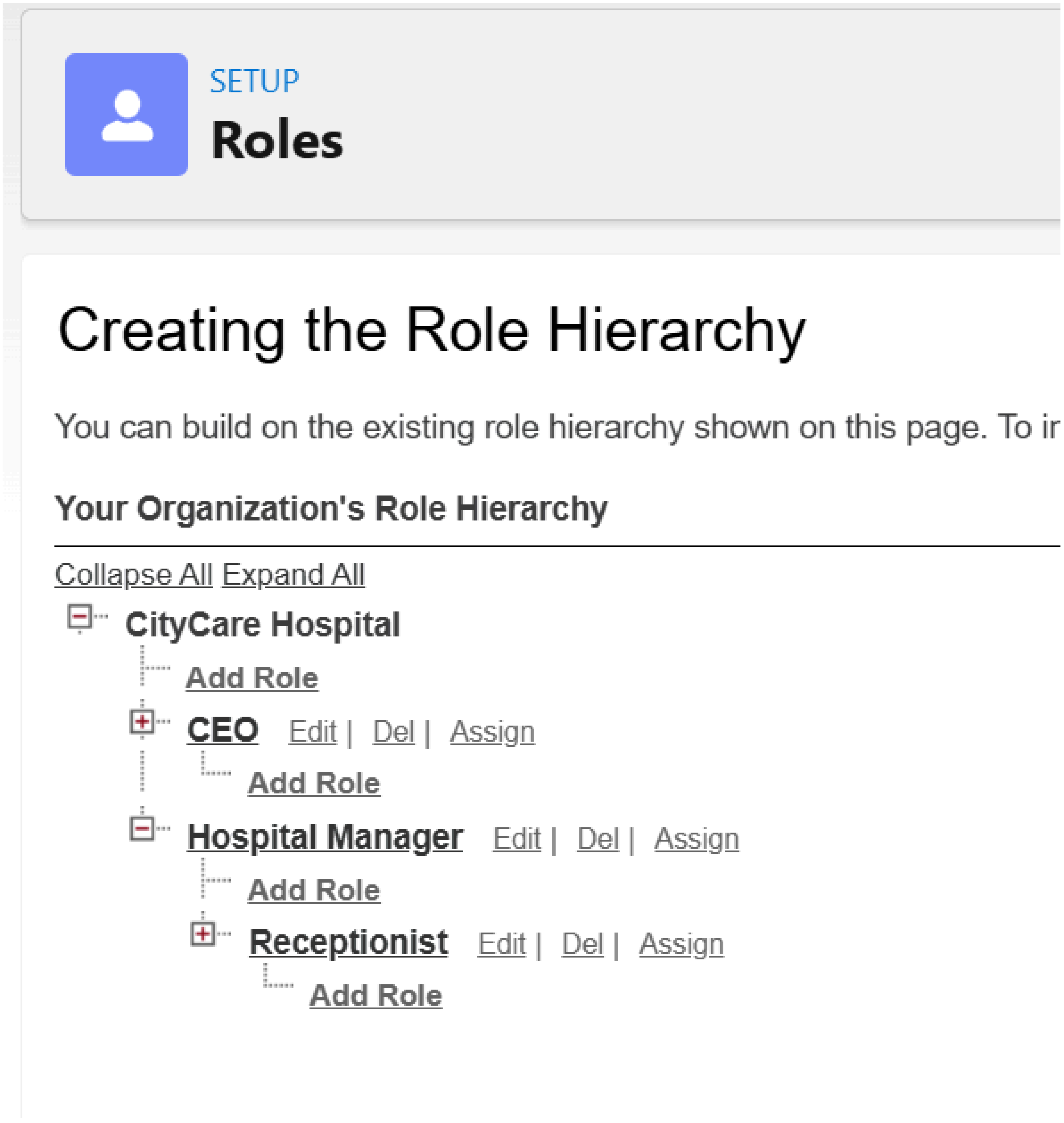


Edit | Sharing | Reset Password | Freeze | View Summary

Name	Front Desk	Role	Receptionist
Alias	recep	User License	Salesforce
Email	altamashfaruqui036@gmail.com [Verified]	Profile	Standard User
Username	altamashfaruqui036@gmail.com	Active	✓



Name	Hospital Manager	Role	Hospital Manager
Alias	hmanager	User License	Salesforce
Email	altamashfaruqui.manager@gmail.com [Verify]	Profile	System Administrator
Username	altamashfaruqui.manager@gmail.com	Active	✓



SETUP

Roles

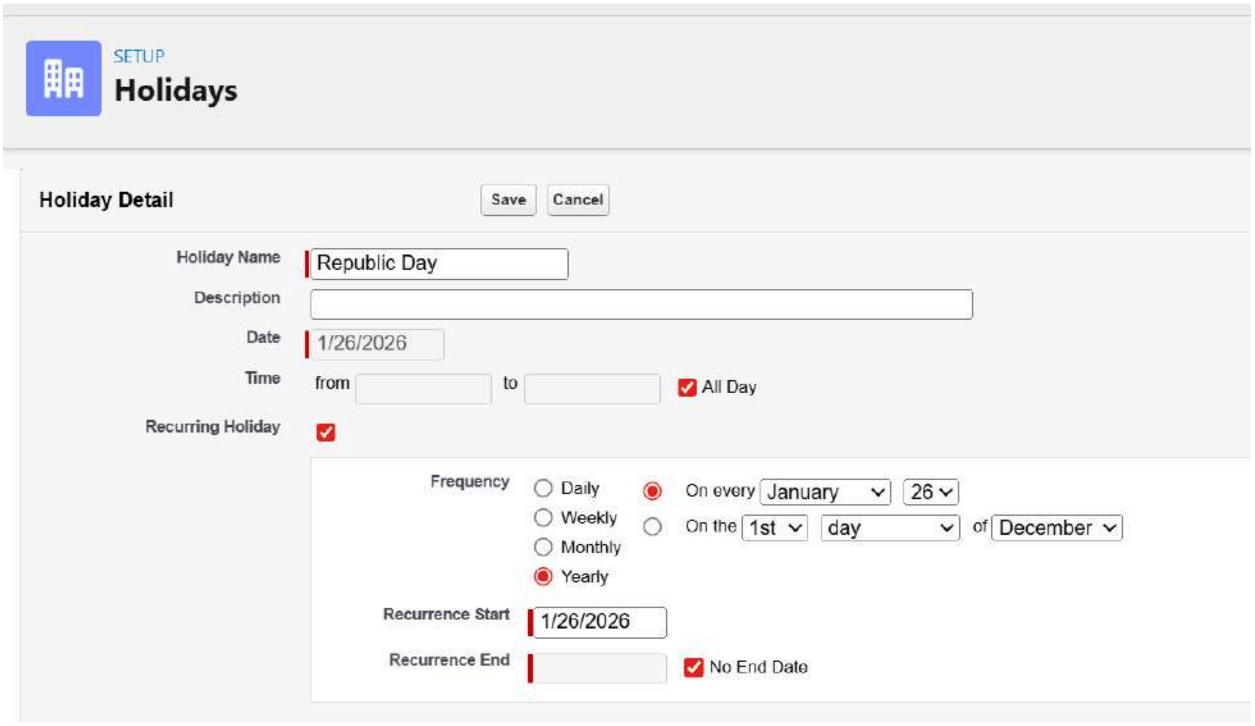
Creating the Role Hierarchy

You can build on the existing role hierarchy shown on this page. To ir

Your Organization's Role Hierarchy

Collapse All Expand All

- CityCare Hospital
  - Add Role
  - CEO Edit | Del | Assign
    - Add Role
  - Hospital Manager Edit | Del | Assign
    - Add Role
    - Receptionist Edit | Del | Assign
      - Add Role



Holiday Name: Republic Day

Description:

Date: 1/26/2026

Time from to [✓ All Day]

Recurring Holiday [✓]

Frequency: Daily / Weekly / Monthly / [✓ Yearly]

[✓ On every January 26]

On the 1st day of December

Recurrence Start: 1/26/2026

Recurrence End: [✓ No End Date]

◇ PHASE 3: Data Modeling & Relationships

(Hospital Appointment and Bed Management CRM System)

Phase 3 Objective:

Design the **core data structure** of the hospital system by creating **custom objects, fields, and relationships**.

Think of this phase as **building the hospital database** inside Salesforce.

🧠 What We Will Build in Phase 3

- Patients
- Doctors
- Appointments
- Relationships between them
- Page layouts for usability

☑ STEP 3.1 — Identify Objects (Hospital Context)

◇ Standard Objects Used

Object	Purpose
Contact	Patients

📌 Salesforce already provides Contact → we reuse it as **Patient**

◇ Custom Objects to Create

Custom Object	Purpose
Doctor	Stores doctor details
Appointment	Links patient & doctor

☑ STEP 3.2 — Create Custom Object: Doctor

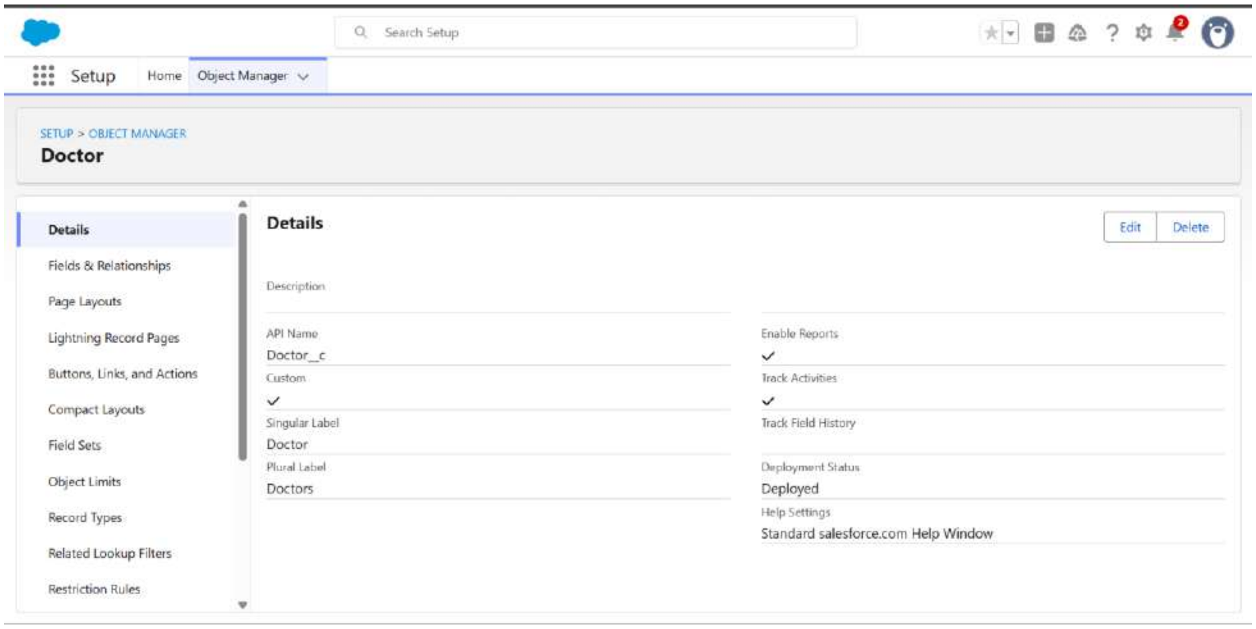
Navigation

Setup → Object Manager → Create → Custom Object

Object Details

Field	Value
Label	**Doctor**
Plural Label	Doctors
Object Name	`Doctor__c`
Record Name	Doctor Name (Text)
Allow Reports	✅
Allow Activities	✅

➡️ Click **Save**



Details

Description

API Name: Doctor__c

Custom: ✓

Singular Label: Doctor

Plural Label: Doctors

Enable Reports: ✓

Track Activities: ✓

Track Field History

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

☑ STEP 3.3 — Create Fields for Doctor Object

Go to:

Doctor → Fields & Relationships → New

Create these fields **one by one**:

Field Label	Type	Example
Specialization	Picklist	Cardiologist, ENT
Phone Number	Phone	
Email	Email	
Experience (Years)	Number	10
Availability Status	Picklist	Available / On Leave
Doctor ID	Auto Number	DOC-{0001}
Consultation Fee	Currency	INR

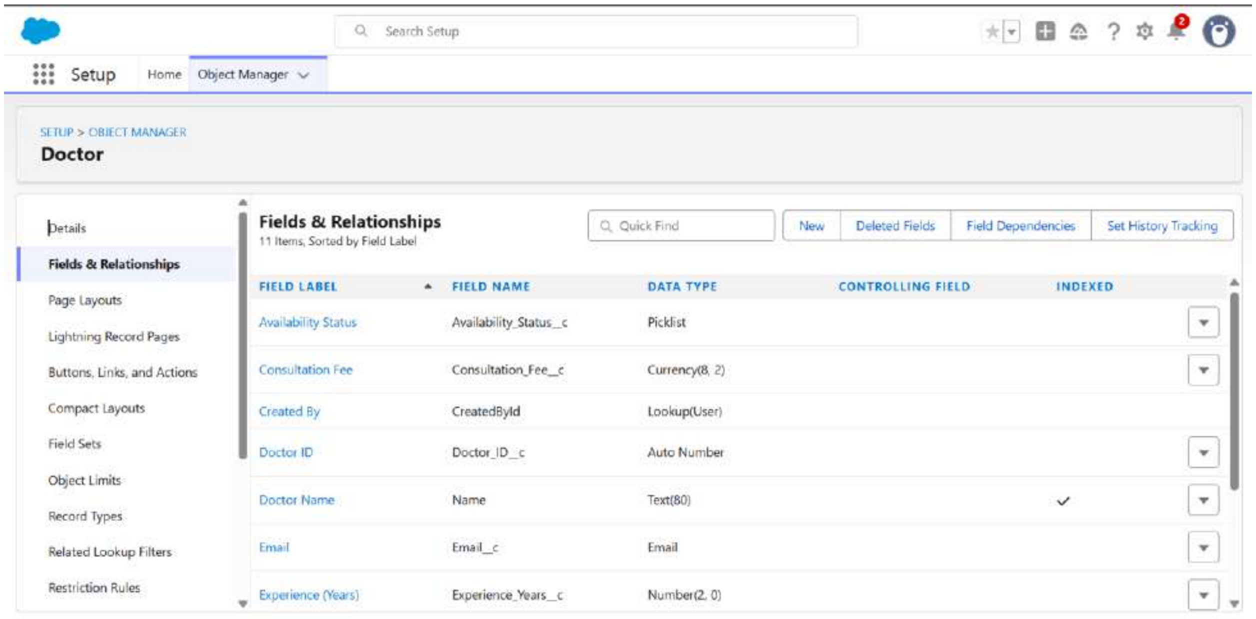
📌 Picklist Values

Specialization

```
Cardiologist
Orthopedic
Neurologist
ENT
General Physician
```

Availability Status

```
Available
On Leave
Unavailable
```



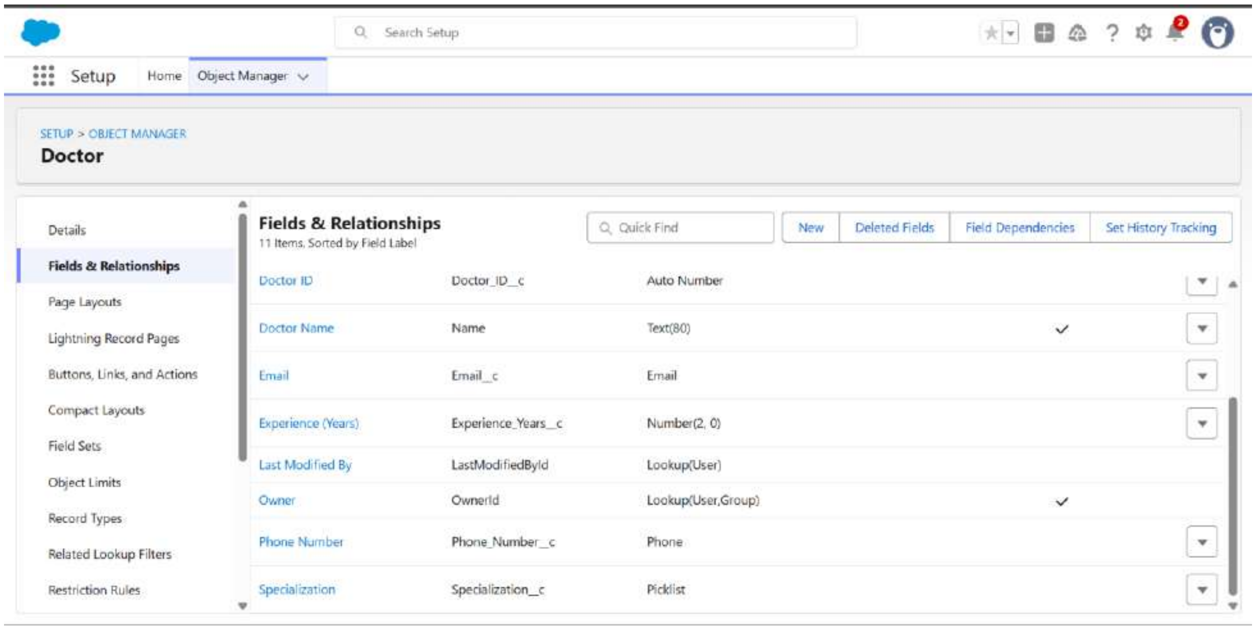
SETUP > OBJECT MANAGER

Doctor

Fields & Relationships

11 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Availability Status	Availability_Status__c	Picklist		
Consultation Fee	Consultation_Fee__c	Currency(8, 2)		
Created By	CreatedById	Lookup(User)		
Doctor ID	Doctor_ID__c	Auto Number		
Doctor Name	Name	Text(80)		✓
Email	Email__c	Email		
Experience (Years)	Experience_Years__c	Number(2, 0)		



SETUP > OBJECT MANAGER

Doctor

Fields & Relationships

11 Items, Sorted by Field Label

Doctor ID	Doctor_ID__c	Auto Number	
Doctor Name	Name	Text(80)	✓
Email	Email__c	Email	
Experience (Years)	Experience_Years__c	Number(2, 0)	
Last Modified By	LastModifiedById	Lookup(User)	
Owner	OwnerId	Lookup(User,Group)	✓
Phone Number	Phone_Number__c	Phone	
Specialization	Specialization__c	Picklist	

☑ STEP 3.4 — Create Custom Object: Appointment

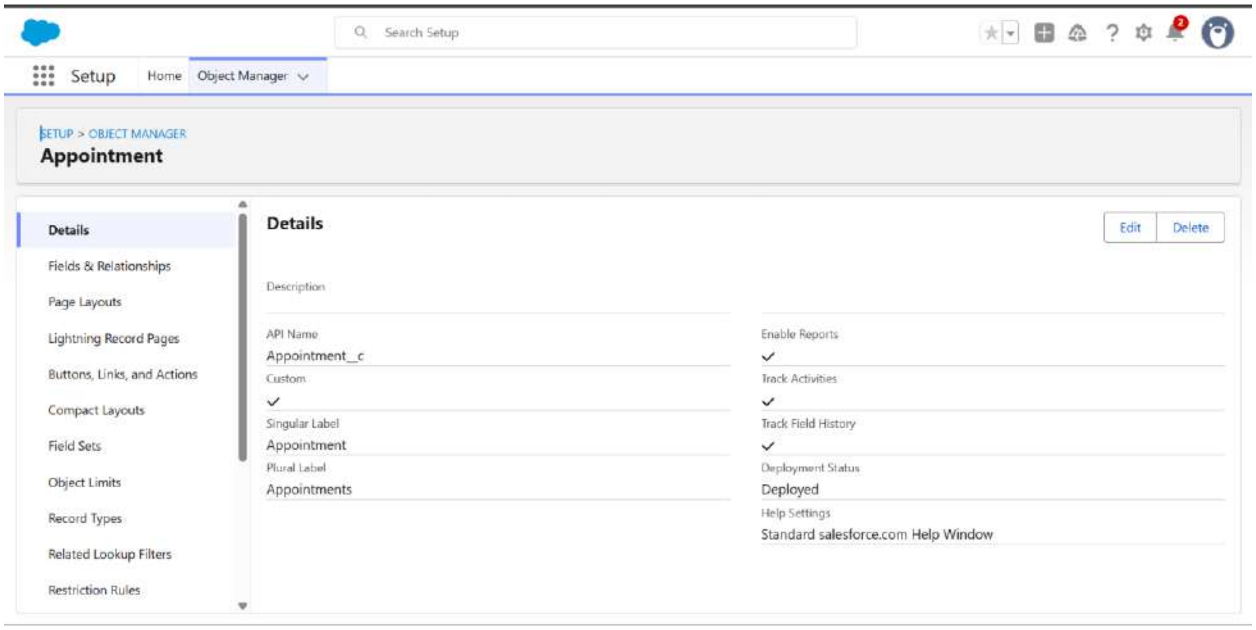
📍 Navigation

Object Manager → Create → Custom Object

Object Details

Field	Value
Label	**Appointment**
Plural Label	Appointments
Object Name	`Appointment__c`
Record Name	Appointment Number (Auto Number)
Format	AP-{0000}
Allow Reports	✓
Allow Activities	✓

➡ Click **Save**



Details

Description

API Name: Appointment__c
Custom: ✓
Singular Label: Appointment
Plural Label: Appointments
Enable Reports: ✓
Track Activities: ✓
Track Field History: ✓
Deployment Status: Deployed
Help Settings: Standard salesforce.com Help Window

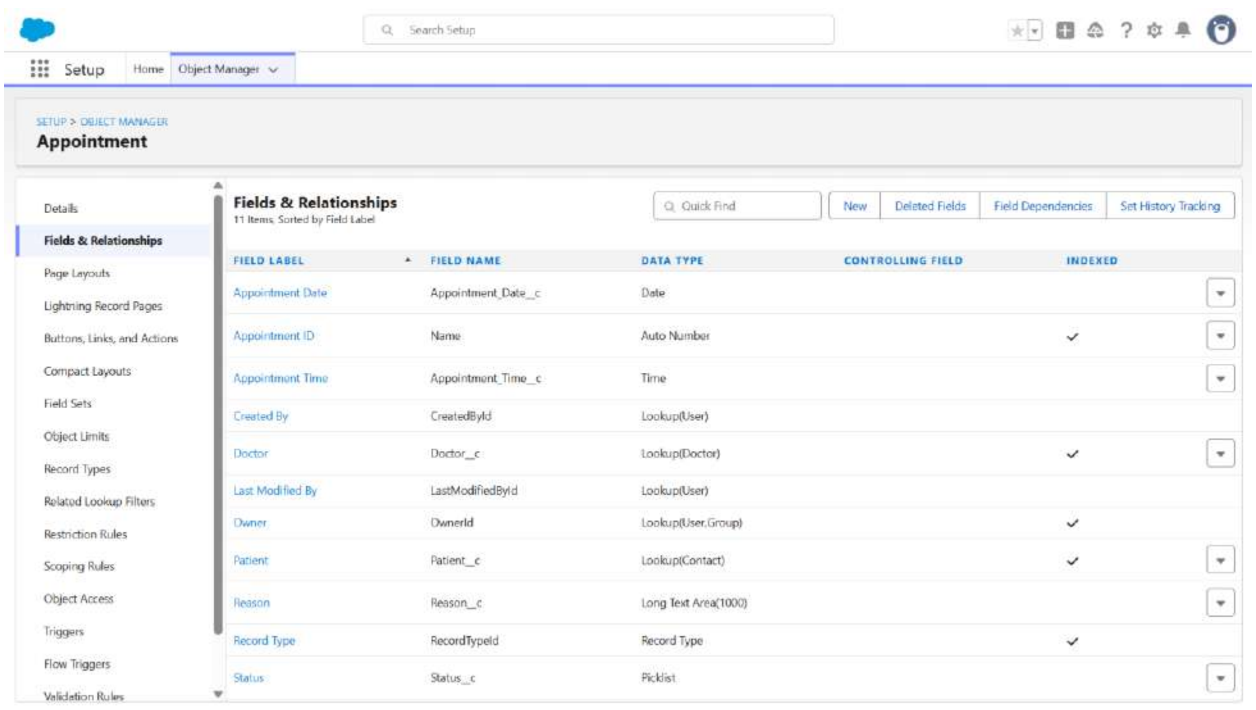
☑ STEP 3.5 — Create Fields for Appointment

Create these fields:

Field Label	Type	Purpose
Appointment Date	Date	Visit date
Appointment Time	Time	Visit time
Status	Picklist	Booking status
Reason	Long Text Area	Patient issue

Status Picklist Values

```
Booked
Checked In
Completed
Cancelled
```



Fields & Relationships
11 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Appointment Date	Appointment_Date__c	Date		
Appointment ID	Name	Auto Number		✓
Appointment Time	Appointment_Time__c	Time		
Created By	CreatedById	Lookup(User)		
Doctor	Doctor__c	Lookup(Doctor)		✓
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Patient	Patient__c	Lookup(Contact)		✓
Reason	Reason__c	Long Text Area(1000)		
Record Type	RecordTypeId	Record Type		✓
Status	Status__c	Picklist		

☑ STEP 3.6 — Create Relationships (MOST IMPORTANT)

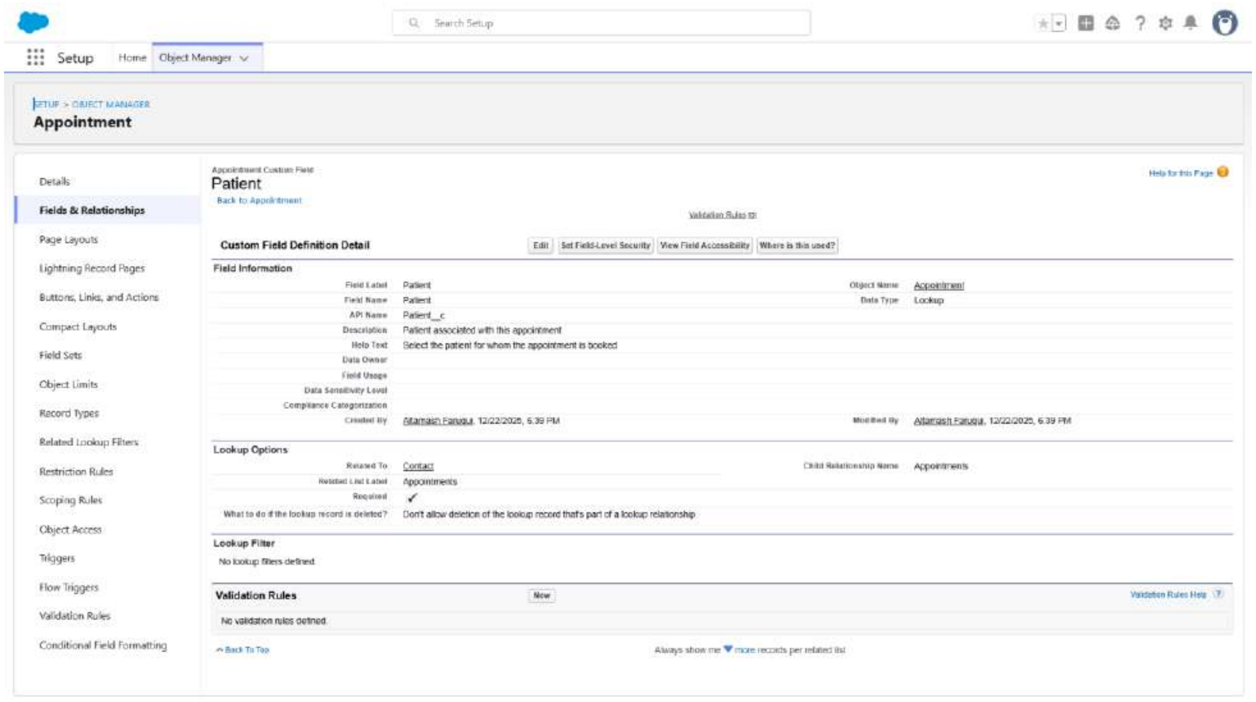
Relationship 1: Appointment → Patient

Path

Appointment → Fields → New → Lookup Relationship

Setting	Value
Related To	**Contact**
Field Label	Patient
Child Relationship Name	Appointments

✓ This connects **Appointments to Patients**



Appointment

Appointment Custom Field

Patient

Custom Field Definition Detail

Field Information

Field Label	Patient	Object Name	Appointment
Field Name	Patient	Data Type	Lookup
API Name	Patient__c		
Description	Patient associated with this appointment		
Help Text	Select the patient for whom the appointment is booked		
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			
Created By	Altamash Faruqui, 12/22/2025, 6:39 PM	Modified By	Altamash Faruqui, 12/22/2025, 6:39 PM

Lookup Options

Related To	Contact	Child Relationship Name	Appointments
Related List Label	Appointments		
Required	✓		
What to do if the lookup record is deleted?	Don't allow deletion of the lookup record that's part of a lookup relationship		

Lookup Filter

No lookup filters defined.

Validation Rules

No validation rules defined.

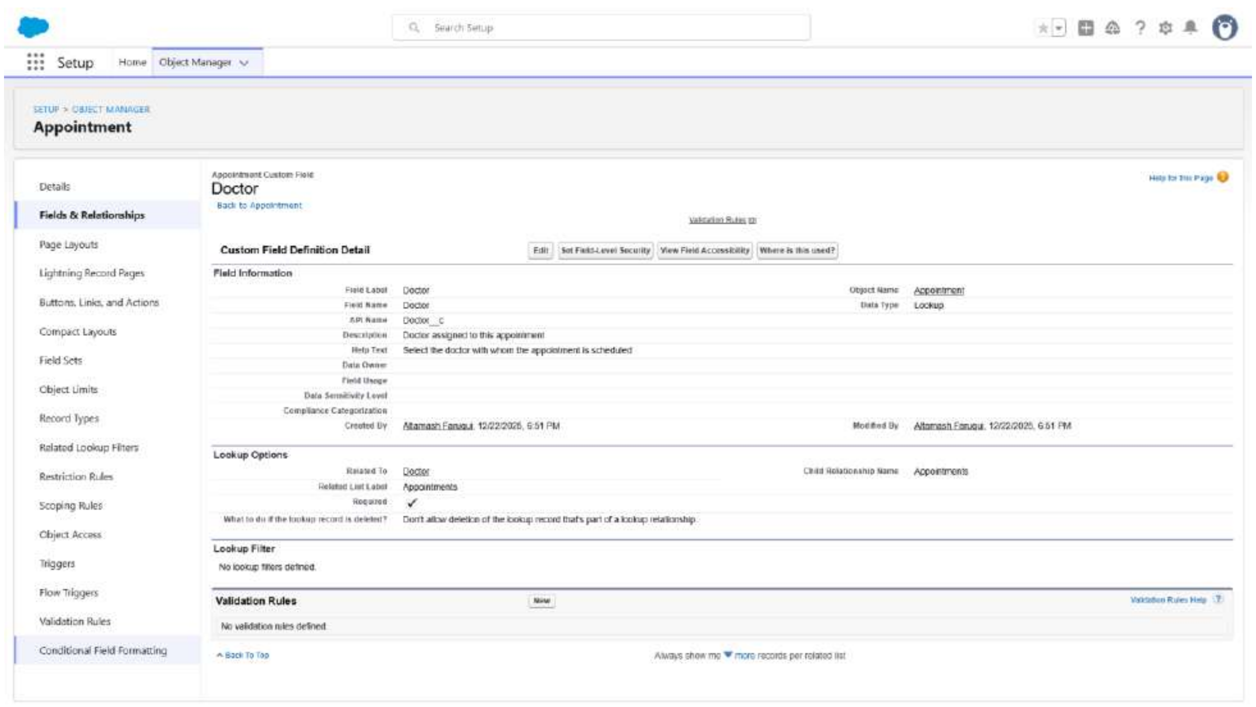
🔗 Relationship 2: Appointment → Doctor

📍 Path

Appointment → Fields → New → Lookup Relationship

Setting	Value
Related To	**Doctor**
Field Label	Doctor
Child Relationship Name	Appointments

✓ This connects **Appointments to Doctors**



Appointment Custom Field

Doctor

Custom Field Definition Detail

Field Information

Field Label	Doctor	Object Name	Appointment
Field Name	Doctor	Data Type	Lookup
API Name	Doctor__c		
Description	Doctor assigned to this appointment		
Help Text	Select the doctor with whom the appointment is scheduled		
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			
Created By	Altamash Faruqui, 12/22/2025, 6:51 PM	Modified By	Altamash Faruqui, 12/22/2025, 6:51 PM

Lookup Options

Related To	Doctor	Child Relationship Name	Appointments
Related List Label	Appointments		
Required	✓		
What to do if the lookup record is deleted?	Don't allow deletion of the lookup record that's part of a lookup relationship.		

Lookup Filter

No lookup filters defined.

Validation Rules

No validation rules defined.

☑ STEP 3.7 — Record Types (Appointments)

Navigation

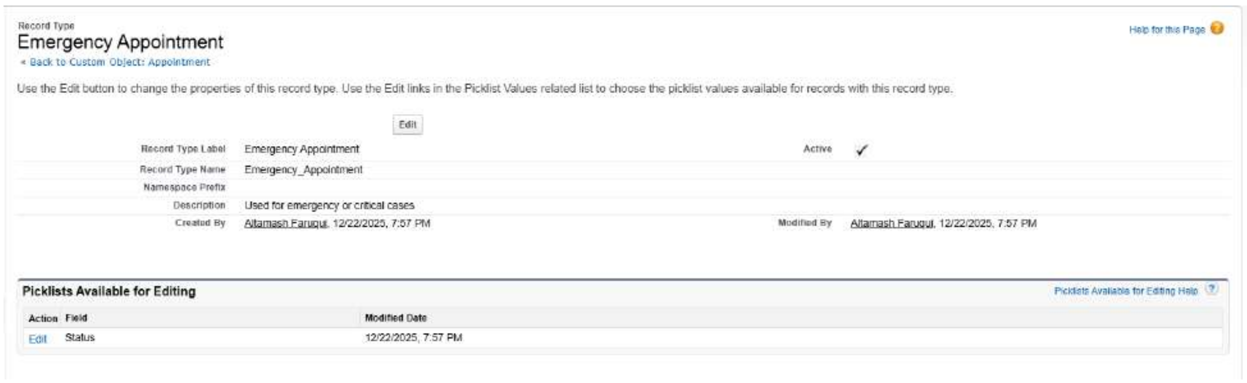
Appointment → Record Types → New

Create **2 Record Types**:

Record Type	Purpose
OPD Appointment	Regular consultation
Emergency Appointment	Emergency cases

✓ Assign both to:

- Hospital Manager
- Receptionist



Record Type

Emergency Appointment

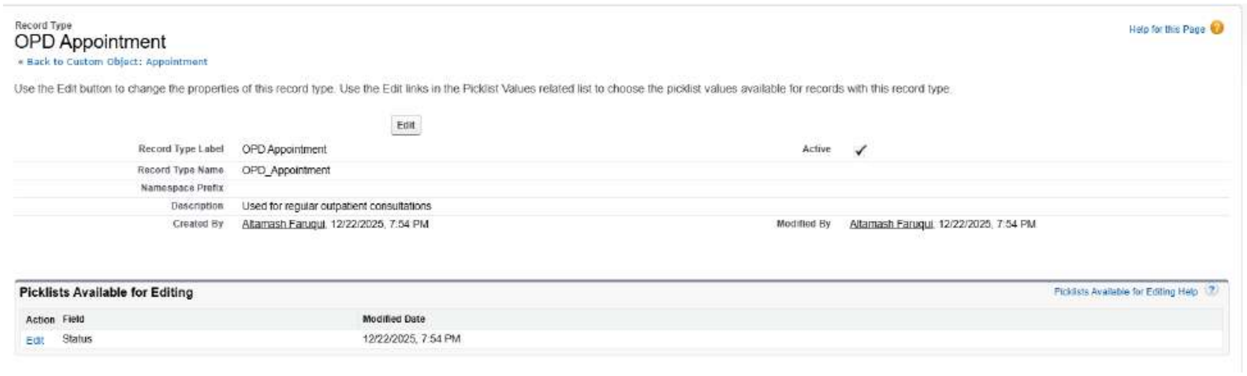
« Back to Custom Object: Appointment

Use the Edit button to change the properties of this record type. Use the Edit links in the Picklist Values related list to choose the picklist values available for records with this record type.

Record Type Label	Emergency Appointment	Active	✓
Record Type Name	Emergency_Appointment		
Namespace Prefix			
Description	Used for emergency or critical cases		
Created By	Altamash Faruqui, 12/22/2025, 7:57 PM	Modified By	Altamash Faruqui, 12/22/2025, 7:57 PM

Picklists Available for Editing

Action	Field	Modified Date
Edit	Status	12/22/2025, 7:57 PM



Record Type

OPD Appointment

« Back to Custom Object: Appointment

Use the Edit button to change the properties of this record type. Use the Edit links in the Picklist Values related list to choose the picklist values available for records with this record type.

Record Type Label	OPD Appointment	Active	✓
Record Type Name	OPD_Appointment		
Namespace Prefix			
Description	Used for regular outpatient consultations		
Created By	Altamash Faruqui, 12/22/2025, 7:54 PM	Modified By	Altamash Faruqui, 12/22/2025, 7:54 PM

Picklists Available for Editing

Action	Field	Modified Date
Edit	Status	12/22/2025, 7:54 PM

☑ STEP 3.8 — Page Layout Configuration

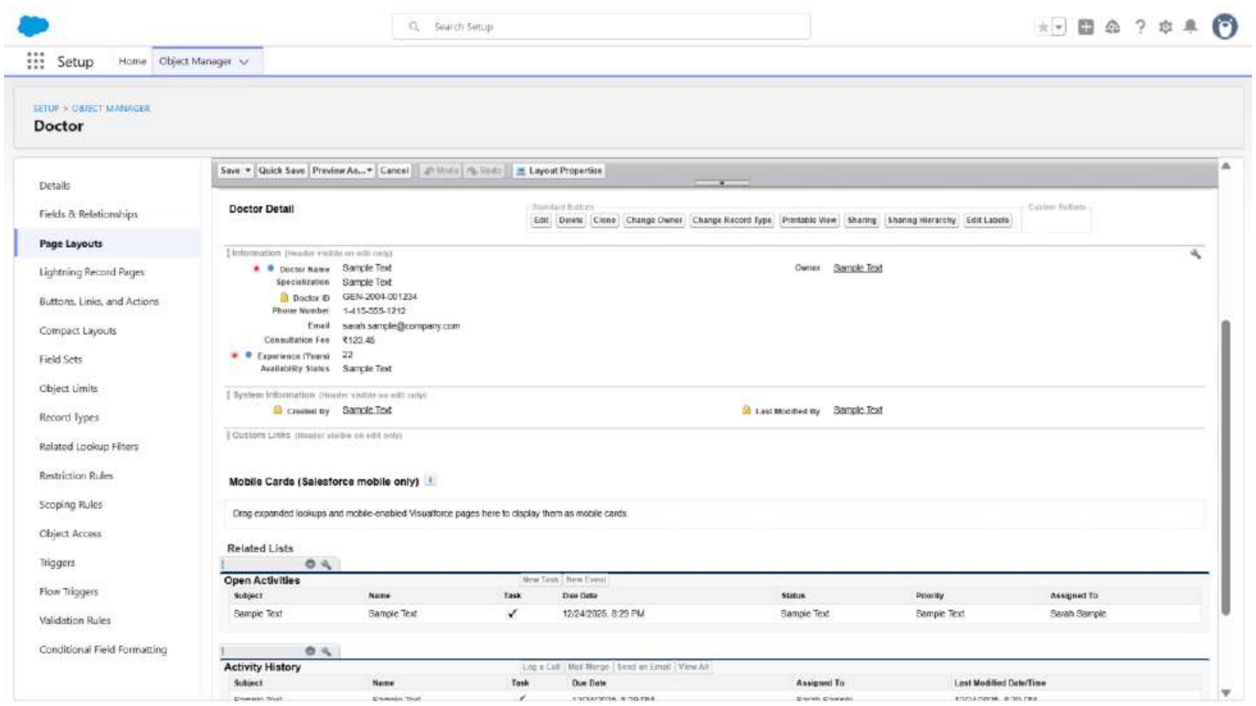
Doctor Page Layout

Add fields:

- Doctor Name
- Specialization
- Phone
- Email
- Availability Status

Add Related List:

- **Appointments**



Doctor Detail

Information (Header visible on edit only)

Doctor Name	Sample Text	Owner	Sample Text
Specialization	Sample Text		
Doctor ID	GEN-2004-001234		
Phone Number	1-415-555-1212		
Email	sarah.sample@company.com		
Consultation Fee	₹123.45		
Experience (Years)	22		
Availability Status	Sample Text		

System Information (Header visible on edit only)

Created By	Sample Text	Last Modified By	Sample Text

Custom Links (Header visible on edit only)

Mobile Cards (Salesforce mobile only)

Drag expanded lookups and mobile-enabled Visualforce pages here to display them as mobile cards.

Related Lists

Open Activities

Subject	Name	Task	Due Date	Status	Priority	Assigned To
Sample Text	Sample Text	✓	12/24/2025, 8:29 PM	Sample Text	Sample Text	Sarah Sample

Activity History

Subject	Name	Task	Due Date	Assigned To	Last Modified Date/Time

Appointment Page Layout

Add fields:

- Patient
- Doctor
- Appointment Date
- Appointment Time
- Status
- Reason

STEP 3.9 — Compact Layouts (Optional but Professional)

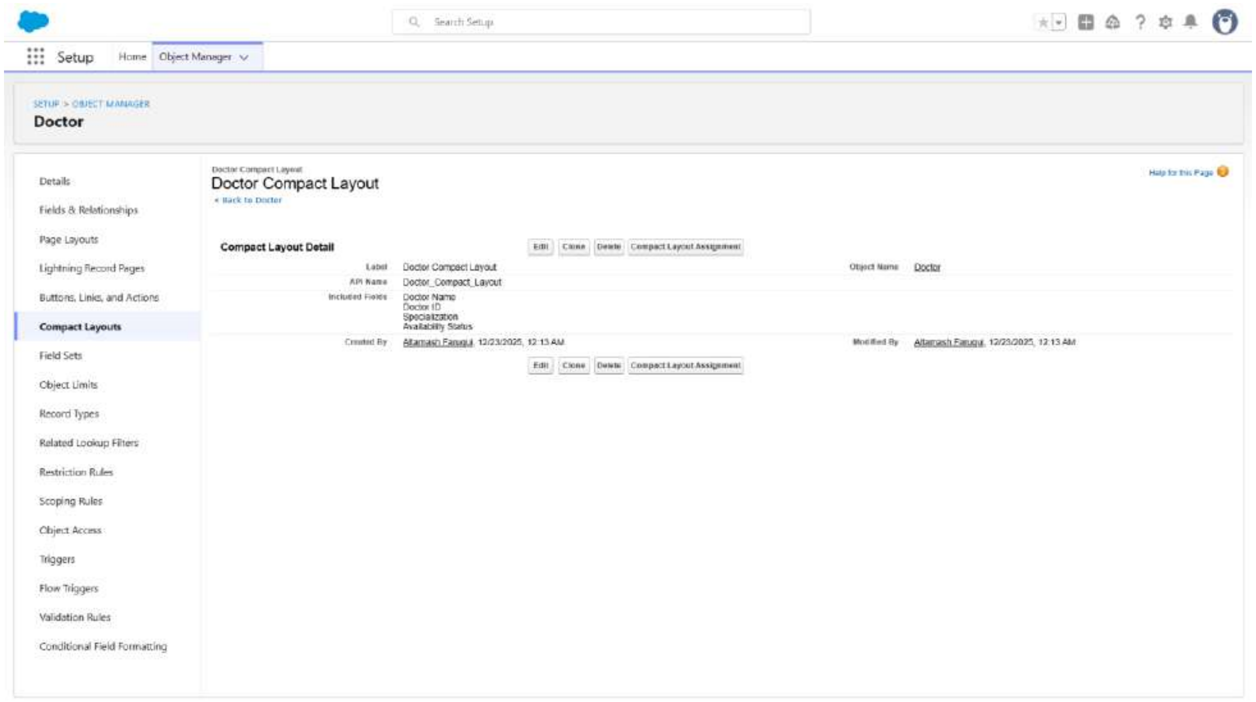
Doctor Compact Layout

Show:

- Doctor Name
- Specialization
- Availability Status

📍 Path:

Doctor → Compact Layouts → New



Doctor Compact Layout

Doctor Compact Layout

Compact Layout Detail

Label	Doctor Compact Layout	Object Name	Doctor
API Name	Doctor_Compact_Layout		
Included Fields	Doctor Name Doctor ID Specialization Availability Status		
Created By	Altamash Faruqui, 12/23/2025, 12:13 AM	Modified By	Altamash Faruqui, 12/23/2025, 12:13 AM

☑ STEP 3.10 — Schema Builder Verification

📍 Navigation

Setup → Schema Builder

✓ Ensure:

- Appointment → Lookup → Contact (Patient)
- Appointment → Lookup → Doctor

This is your **final data model proof**.

🎯 PHASE 3 COMPLETION CHECKLIST

✓ Doctor object created

✓ Appointment object created

✓ Patient handled via Contact

✓ Relationships correctly set

✓ Record types configured

✓ Page layouts clean & usable

✅ **PHASE 3 SUCCESSFULLY COMPLETED**

🔜 Next Phase

When you're ready, say:

"Proceed with Phase 4 — Validation Rules & Automation"

We'll build:

- Appointment date validation
- Doctor availability logic
- Auto-status updates
- Real hospital-style automation 🚑 ✨

☑ PHASE 4 — BUSINESS LOGIC & AUTOMATION

Goal of Phase 4:

Add *intelligence* to the system so users **cannot make wrong entries** and **processes become automatic**.

◇ WHY PHASE 4 IS IMPORTANT

Right now:

- Users can technically enter **wrong appointments**
- Doctor availability doesn't auto-control bookings
- Status changes are manual and error-prone

📌 **Phase 4 fixes all of this using Salesforce logic**

🧩 PHASE 4 — MODULE BREAKDOWN

Step	Feature
4.1	Validation Rules
4.2	Appointment Status Control
4.3	Doctor Availability Automation
4.4	Optional Email Alerts
4.5	Final Testing

☑ STEP 4.1 — VALIDATION RULES (Mandatory)

🎯 Purpose

Prevent **invalid appointments** from being created.

🔒 Validation Rule 1

❌ No Appointment in the Past

📍 Object

Appointment

🧠 Logic

Appointment Date + Time **must be future**

🧾 Formula

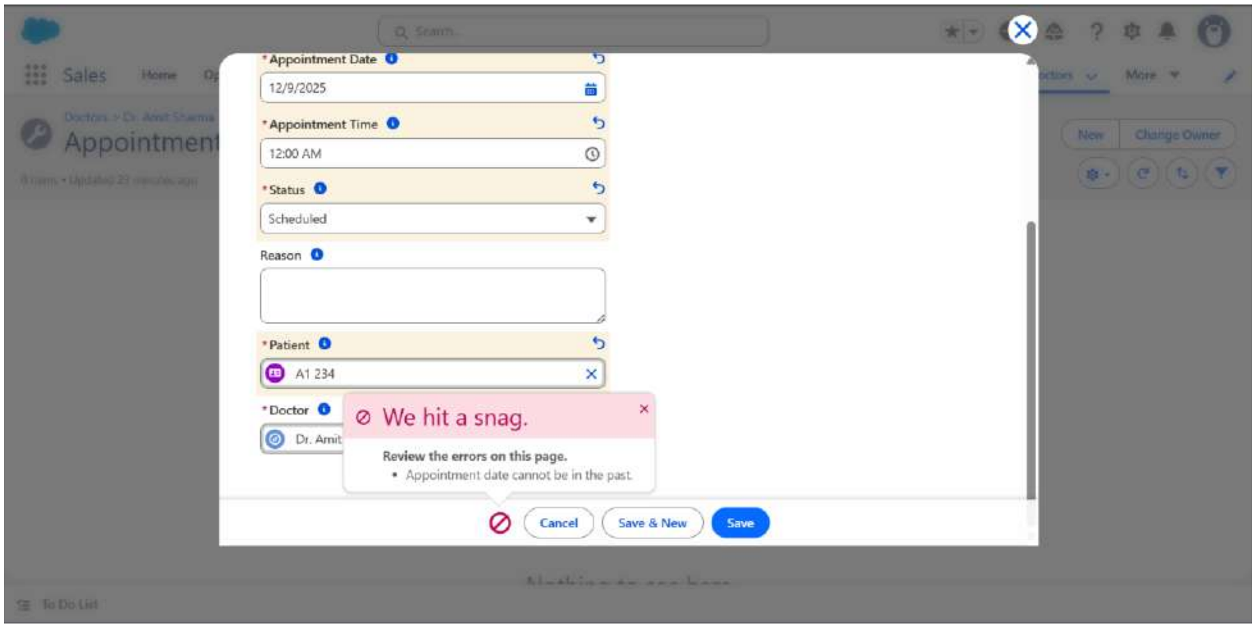
```
Appointment_Date__c < TODAY()
```

🛑 Error Message

Appointment date cannot be in the past.

📌 Error Location

Field: **Appointment Date**


* Appointment Date
12/9/2025
* Appointment Time
12:00 AM
* Status
Scheduled
Reason
* Patient
A1 234
* Doctor
Dr. Amit

We hit a snag.
Review the errors on this page.
- Appointment date cannot be in the past.

Validation Rule 2

Prevent Booking if Doctor is Unavailable

Object

Appointment

Logic

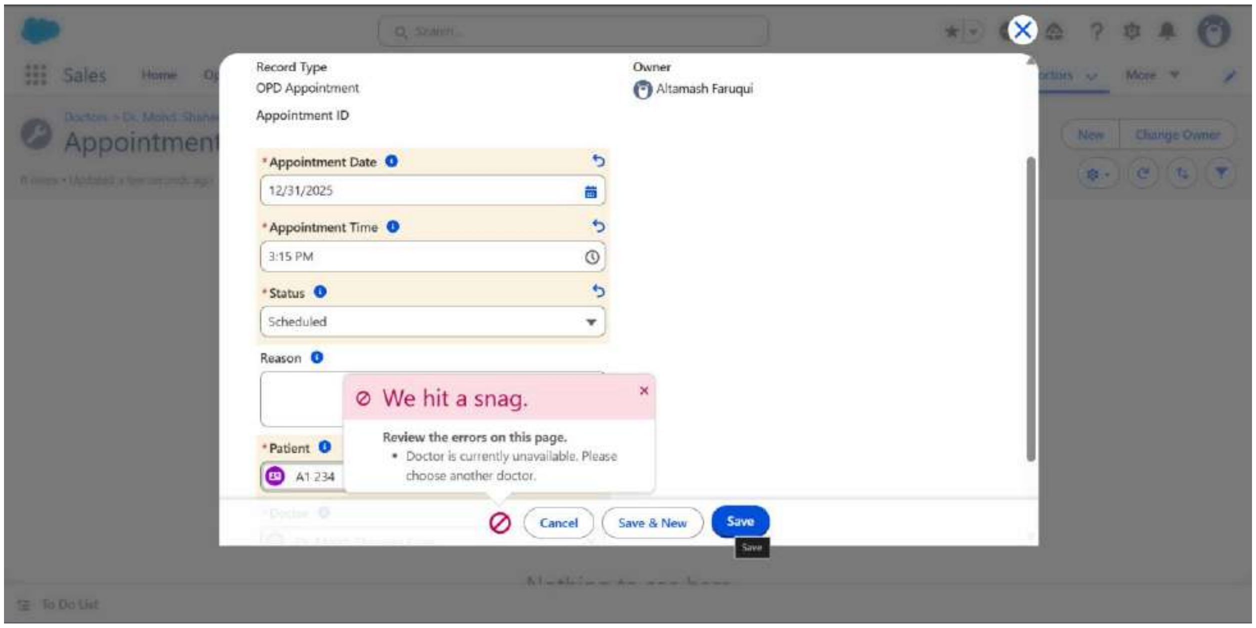
If linked Doctor's Availability Status = `Unavailable` → block booking

Formula

```
Doctor__r.Availability_Status__c = "Unavailable"
```

Error Message

Doctor is currently unavailable. Please choose another doctor.


Record Type
OPD Appointment
Owner
Altamash Faruqui
Appointment ID
*Appointment Date
12/31/2025
*Appointment Time
3:15 PM
*Status
Scheduled
Reason
*Patient
A1 234
We hit a snag.
Review the errors on this page.
- Doctor is currently unavailable. Please choose another doctor.

Validation Rule 3

Emergency Appointments Must Have Reason

Object

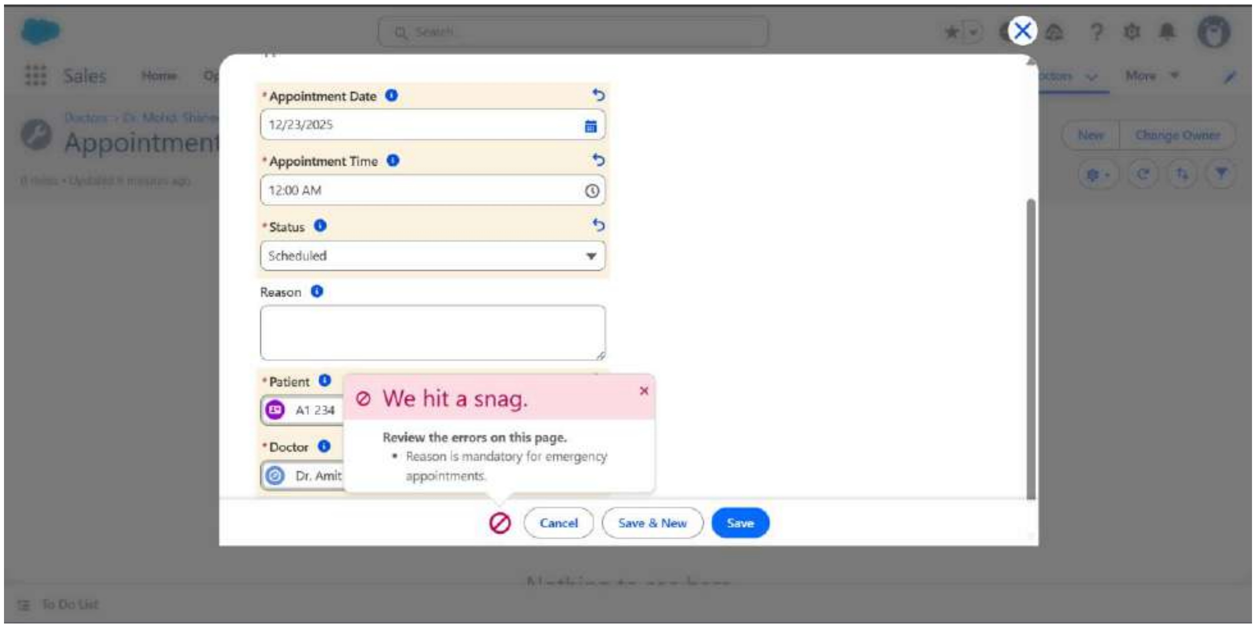
Appointment

Formula

```
AND(
ISPICKVAL(RecordType.Name , "Emergency Appointment"),
ISBLANK(Reason__c)
)
```

Error Message

Reason is mandatory for emergency appointments.



We hit a snag.

Review the errors on this page.

- Reason is mandatory for emergency appointments.

STEP 4.2 — STATUS CONTROL (Business Rule)

Appointment Status Values

Status
Scheduled
Completed
Cancelled

Rule: Completed Appointment Cannot Be Edited

Object

Appointment

Formula

```
ISPICKVAL(Status__c , "Completed")
```

Error Message

Completed appointments cannot be modified.

STEP 4.3 — AUTOMATION (FLOW)

Goal

Automatically update **Doctor Availability**

Automation Logic

Event	Doctor Availability
Appointment Created	Unavailable
Appointment Completed	Available
Appointment Cancelled	Available

Tool Used

Record-Triggered Flow

Flow Type

After Save

Object

Appointment

🧠 Flow Conditions

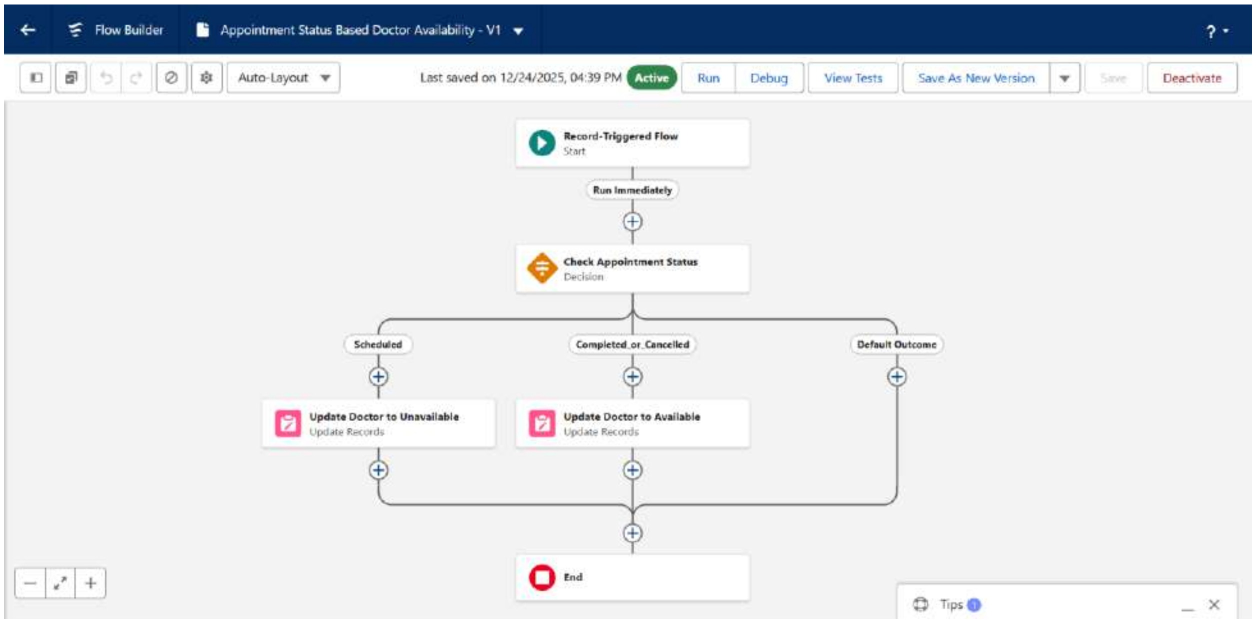
```
Status = Scheduled
```

➞ Update Doctor.Availability_Status = `Unavailable`

```
Status = Completed OR Cancelled
```

➞ Update Doctor.Availability_Status = `Available`

📌 This makes your project "REAL-WORLD READY"



📧 STEP 4.4 — EMAIL ALERT (Optional but Professional)

Trigger

When Emergency Appointment is created

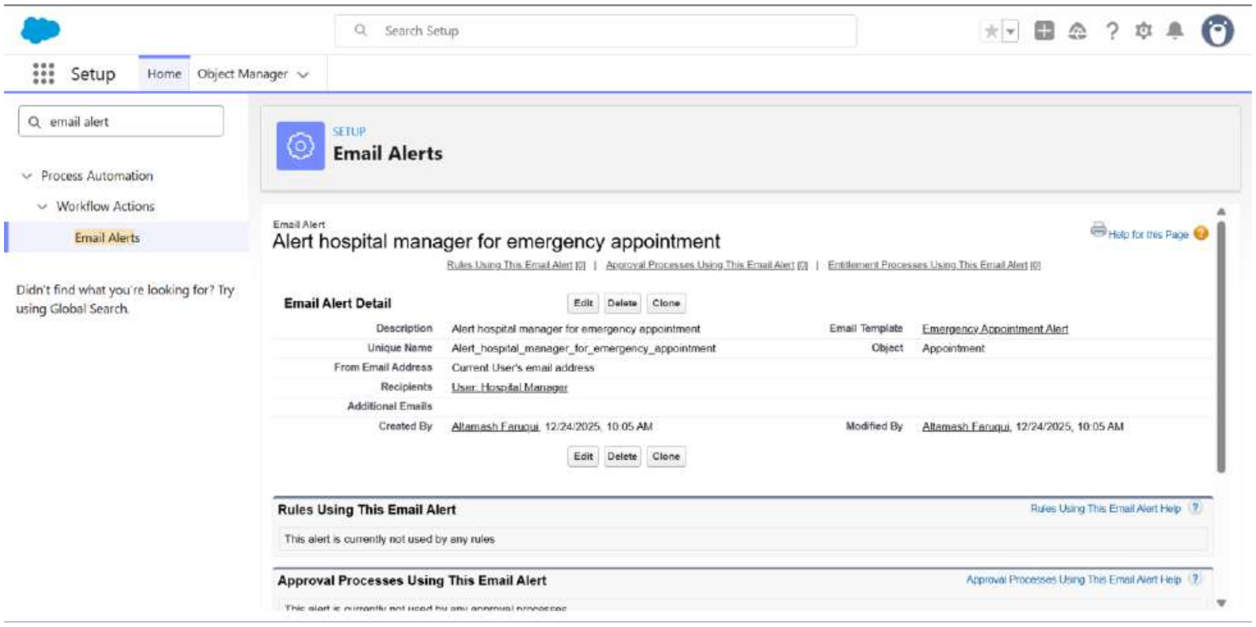
Recipient

Hospital Manager

Email Content

Emergency appointment created. Immediate attention required.

📌 Use **Email Alert + Flow**



Email Alert
Alert hospital manager for emergency appointment

Rules Using This Email Alert [0] | Approval Processes Using This Email Alert [0] | Entitlement Processes Using This Email Alert [0]

Email Alert Detail

Description	Alert hospital manager for emergency appointment	Email Template	Emergency Appointment Alert
Unique Name	Alert_hospital_manager_for_emergency_appointment	Object	Appointment
From Email Address	Current User's email address		
Recipients	User: Hospital Manager		
Additional Emails			
Created By	Altamash Faruqui, 12/24/2025, 10:05 AM	Modified By	Altamash Faruqui, 12/24/2025, 10:05 AM

Rules Using This Email Alert — Rules Using This Email Alert Help

This alert is currently not used by any rules

Approval Processes Using This Email Alert — Approval Processes Using This Email Alert Help

This alert is currently not used by any approval processes

🧪 STEP 4.5 — TESTING (Must Mention in PDF)

Test Case	Result
Past appointment	❌ Blocked
Unavailable doctor	❌ Blocked
Emergency without reason	❌ Blocked
Appointment completion	✅ Doctor becomes available

PHASE 5 — Apex Programing

🎯 Phase 5 Goal

Automate **Doctor availability management** based on **Appointment lifecycle**, while ensuring:

- Data consistency
- Validation rule compliance
- Real-world hospital workflow simulation

◇ Business Problem Being Solved

In a real hospital system:

- A doctor **must not be double-booked**
- Doctor availability should **change automatically**
- Manual updates lead to **human errors**

👉 **Phase 5 introduces backend automation using Apex Trigger** to solve this.

◇ Core Business Rules (Derived from Your Implementation)

Appointment Status	Doctor Availability
Scheduled	Unavailable
Completed	Available
Cancelled	Available

⚠️ Validation Rule enforces:

Appointment **cannot be updated** if Doctor is unavailable

🔷 Objects Involved

1️⃣ Doctor__c

Key Fields:

- `Availability_Status__c` *(Picklist)*
    - Available
    - On Leave
    - Unavailable

2️⃣ Appointment__c

Key Fields:

- `Doctor__c` *(Lookup → Doctor)*
- `Patient__c` *(Lookup → Contact)*
- `Status__c` *(Picklist)*
    - Scheduled
    - Completed
    - Cancelled
- `Appointment_Date__c`
- `Appointment_Time__c`

🔷 Automation Implemented (Apex Trigger)

🔧 Trigger Name

`AppointmentTrigger`

🔧 Trigger Events

```
after insert, after update
```

🔧 Trigger Responsibility

- Monitor Appointment status changes
- Automatically update related Doctor availability
- Prevent manual intervention errors

🔹 Trigger Logic Flow (Step-by-Step)

☑ Step 1: Collect Related Doctors

```
Set<Id> doctorIds
```

Purpose:

- Bulk-safe handling
- Avoid SOQL inside loops

☑ Step 2: Fetch Doctors in One Query

```
Map<Id, Doctor__c> doctorMap
```

Purpose:

- Performance optimization
- Governor-limit safe

☑ Step 3: Status-Based Decision Making

When Appointment = Scheduled

```
Doctor.Availability_Status__c = 'Unavailable';
```

➡ Prevents double booking

When Appointment = Completed / Cancelled

```
Doctor.Availability_Status__c = 'Available';
```

➡ Doctor becomes free for new appointments

☑ Step 4: Single DML Operation

```
update doctorsToUpdate;
```

✓ Bulk-safe

✓ Scalable

✓ Production-ready

◇ Apex Trigger Code

trigger AppointmentTrigger on Appointment__c (after insert, after update) {

```
Set<Id> doctorIds = new Set<Id>();

for (Appointment__c appt : Trigger.new) {
    if (appt.Doctor__c != null) {
        doctorIds.add(appt.Doctor__c);
    }
}

if (doctorIds.isEmpty()) {
    return;
}

Map<Id, Doctor__c> doctorMap = new Map<Id, Doctor__c>(
    [
        SELECT Id, Availability_Status__c
        FROM Doctor__c
        WHERE Id IN :doctorIds
    ]
);

List<Doctor__c> doctorsToUpdate = new List<Doctor__c>();

for (Appointment__c appt : Trigger.new) {

    Doctor__c doc = doctorMap.get(appt.Doctor__c);
    if (doc == null) continue;

    // If appointment is scheduled → Doctor unavailable
    if (appt.Status__c == 'Scheduled') {
        doc.Availability_Status__c = 'Unavailable';
        doctorsToUpdate.add(doc);
    }

    // If appointment completed or cancelled → Doctor available
    if (
        appt.Status__c == 'Completed' ||
        appt.Status__c == 'Cancelled'
    ) {
        doc.Availability_Status__c = 'Available';
        doctorsToUpdate.add(doc);
    }
}
```

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


}
```

🔷 Anonymous Apex Testing Strategy (Phase 5 Validation)

To **respect validation rules**, the following execution order is used:

🧪 Test Scenario

1️⃣ Create Doctor (Available)

2️⃣ Create Appointment (Scheduled → Trigger makes Doctor Unavailable)

3️⃣ **Reset Doctor to Available manually** *(to pass validation)*

4️⃣ Update Appointment to Completed

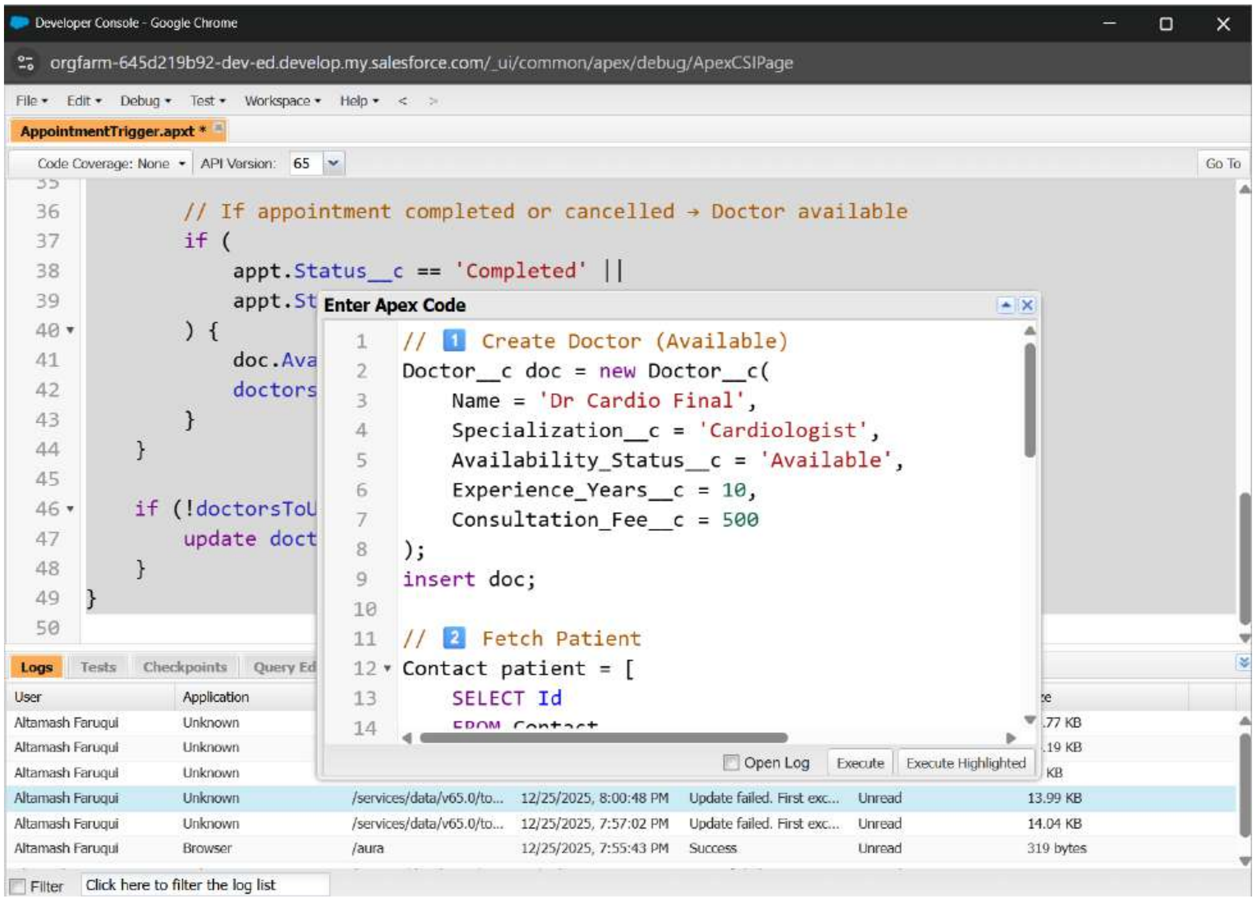
5️⃣ Trigger restores Doctor availability automatically

This validates:

- Trigger correctness
- Validation rule enforcement
- End-to-end automation flow

🔷 Apex Anonymous code

```
// 1️⃣ Create Doctor (Available)
Doctor__c doc = new Doctor__c(

   Name = 'Dr Cardio Final',
    Specialization__c = 'Cardiologist',
    Availability_Status__c = 'Available',
    Experience_Years__c = 10,
    Consultation_Fee__c = 500
);
insert doc;

// 2️⃣ Fetch Patient
Contact patient = [
    SELECT Id
    FROM Contact
    LIMIT 1
];

// 3️⃣ Create Appointment (Scheduled)
Appointment__c appt = new Appointment__c(
    Doctor__c = doc.Id,
    Patient__c = patient.Id,
    Appointment_Date__c = Date.today().addDays(1),
    Appointment_Time__c = Time.newInstance(10, 0, 0, 0),
    Status__c = 'Scheduled'
);
insert appt;

// 🔥 4️⃣ IMPORTANT FIX — make Doctor Available BEFORE update
doc.Availability_Status__c = 'Available';
update doc;

// 5️⃣ Now safely complete Appointment
appt.Status__c = 'Completed';
update appt;

// 6️⃣ Verify Doctor status
Doctor__c d = [
    SELECT Availability_Status__c
    FROM Doctor__c
    WHERE Id = :doc.Id
];
System.debug('Doctor Status after Completed = ' +
d.Availability_Status__c);
```


Developer Console - Google Chrome
orgfarm-645d219b92-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help < >
AppointmentTrigger.apxt *
Code Coverage: None API Version: 65 Go To

```
        // If appointment completed or cancelled → Doctor available
        if (
            appt.Status__c == 'Completed' ||
            appt.St
        ) {
            doc.Ava
            doctors
        }
    }

    if (!doctorsToU
        update doct
    }
}
```

```
// 1 Create Doctor (Available)
Doctor__c doc = new Doctor__c(
    Name = 'Dr Cardio Final',
    Specialization__c = 'Cardiologist',
    Availability_Status__c = 'Available',
    Experience_Years__c = 10,
    Consultation_Fee__c = 500
);
insert doc;

// 2 Fetch Patient
Contact patient = [
    SELECT Id
    FROM Contact
```

```
User              Application
Altamash Faruqui  Unknown                                                                       .77 KB
Altamash Faruqui  Unknown                                                                       .19 KB
Altamash Faruqui  Unknown                                                                       KB
Altamash Faruqui  Unknown   /services/data/v65.0/to...  12/25/2025, 8:00:48 PM  Update failed. First exc...  Unread  13.99 KB
Altamash Faruqui  Unknown   /services/data/v65.0/to...  12/25/2025, 7:57:02 PM  Update failed. First exc...  Unread  14.04 KB
Altamash Faruqui  Browser   /aura                       12/25/2025, 7:55:43 PM  Success                     Unread  319 bytes
```

◇ Key Technical Learnings (Very Important)

Validation Rules > Apex Triggers

- Validation rules execute **after trigger logic**
- DML order matters
- Real-world Apex must respect existing business constraints

Picklist-Driven Automation

- No unnecessary boolean fields
- Clean, readable, admin-friendly logic

🔑 Bulk-Safe Trigger Design

- No SOQL in loops
- No DML in loops
- Handles multiple appointments at once

🔹 Phase 5 Outcomes

✅ Doctor availability updates automatically

✅ Appointment lifecycle fully controlled

✅ Data integrity maintained

✅ Real hospital workflow simulated

✅ Production-ready backend logic

PHASE 6 — UI Development

🎯 Phase 6 Goal

Create a **role-friendly, hospital-style UI** so that:

- Receptionists book appointments fast
- Doctors see their schedules
- Managers track beds & availability

🔹 STEP 6.1 — Create a Dedicated Lightning App (MOST IMPORTANT)

📍 Navigation

Setup → App Manager → New Lightning App

🧩 App Details

- **App Name:** Hospital Management
- **Developer Name:** Hospital_Management
- **Description:** Hospital Appointment & Bed Management CRM

Click **Next**

🎨 App Options

- Navigation Style → **Standard**
- Disable “Enable Setup” (optional)

Click **Next**

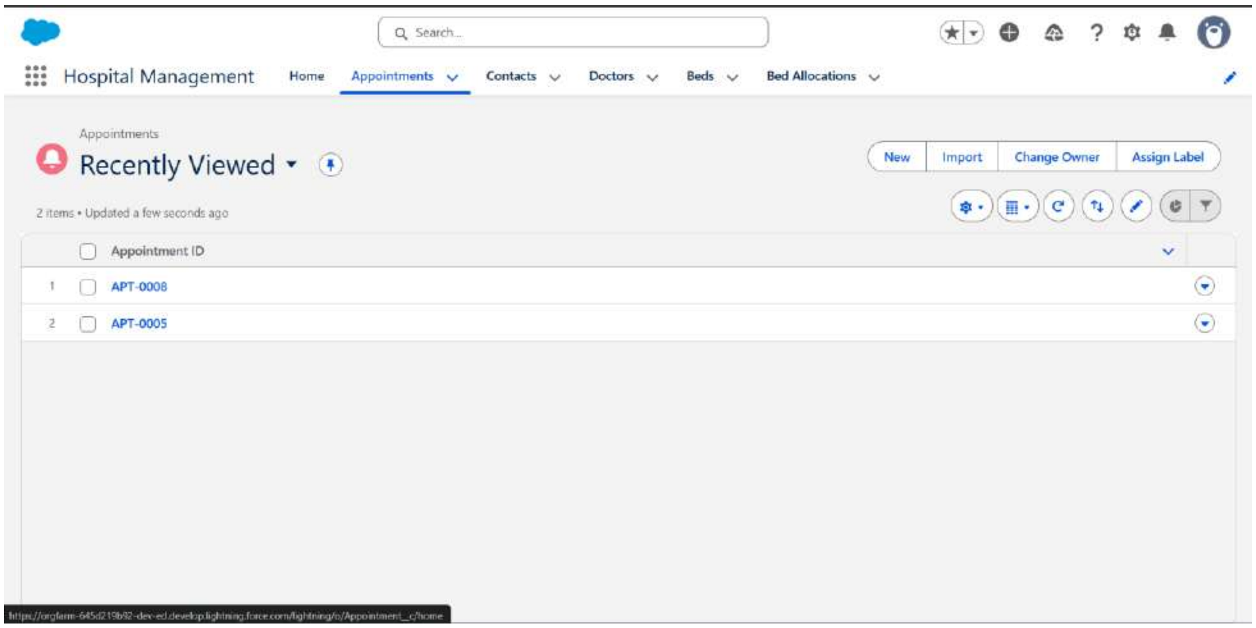
👥 Assign User Profiles

Select:

- System Administrator
- Hospital Manager
- Receptionist

Click **Save & Finish**

✅ You now have a **dedicated hospital app**



◇ STEP 6.2 — Create Custom Tabs (Core Objects)

📍 Navigation

Setup → Tabs → New

🧾 Create these Tabs

1️⃣ Doctor Tab

- Object → `Doctor__c`
- Tab Style → Medical icon
- Visibility → Default On

2️⃣ Appointment Tab

- Object → `Appointment__c`
- Tab Style → Calendar icon
- Visibility → Default On

3️⃣ Bed Tab

- Object → `Bed__c` *(if created earlier)*
- Tab Style → Hospital icon

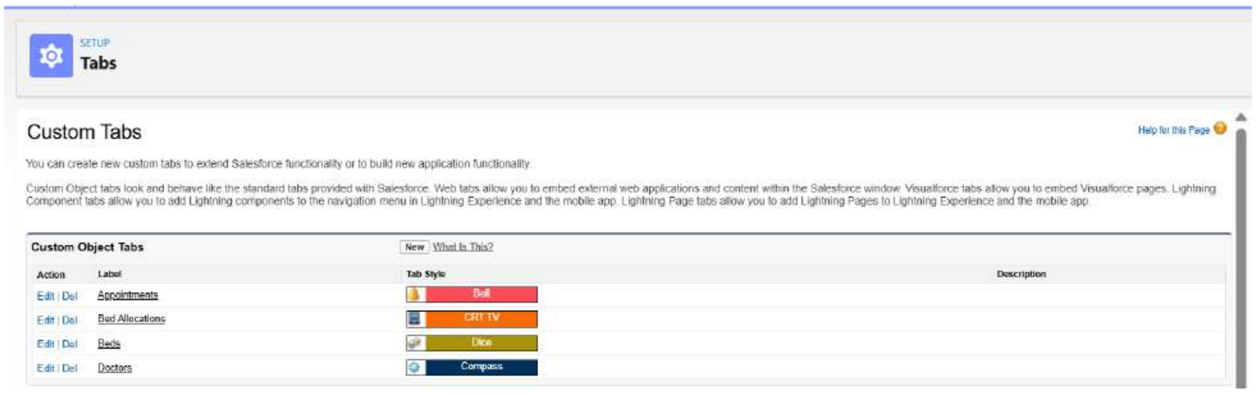
📌 Add Tabs to App

App Manager → Hospital Management → Edit → Navigation Items

Add:

- Doctors
- Appointments
- Beds
- Contacts (Patients)

Remove unnecessary tabs.



SETUP
Tabs

Custom Tabs

Help for this Page

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	Bell	
Edit | Del	Bed Allocations	CRT TV	
Edit | Del	Beds	Dice	
Edit | Del	Doctors	Compass	

◇ STEP 6.3 — Doctor Record Page (Lightning App Builder)

Navigation

Setup → Object Manager → Doctor → Lightning Record Pages → New

Page Type

- Record Page
- Label: **Doctor Record Page**

Layout Design

Header

- Highlights Panel
  - Doctor Name
  - Availability Status
  - Specialization

Left Column

- Record Detail (Compact)

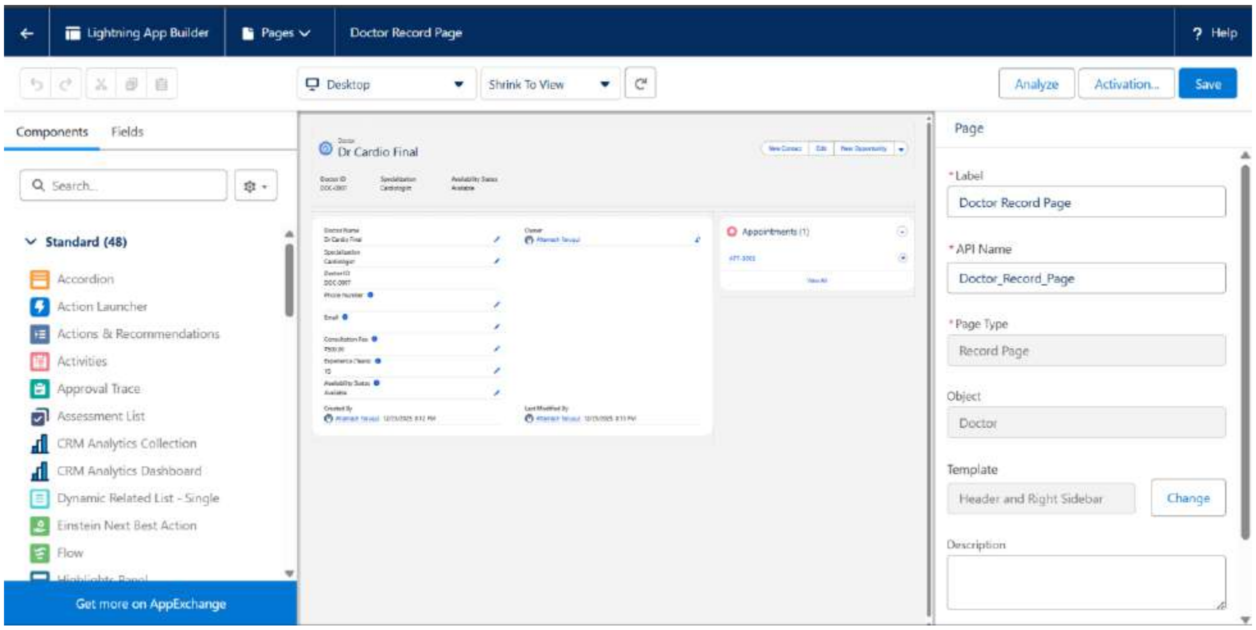
Right Column

- Related List – Single
    - Appointments

Page Assignment

- Assign to **Hospital Management App**
- Profiles:
    - Doctor
    - Hospital Manager

Save & Activate



*Label: Doctor Record Page

*API Name: Doctor_Record_Page

*Page Type: Record Page

Object: Doctor

Template: Header and Right Sidebar [Change]

Description:

◇ STEP 6.4 — Appointment Record Page (MOST USED)

📍 Navigation

Setup → Object Manager → Appointment → Lightning Record Pages → New

🧱 Layout Structure

- Header + Two Column

▧ Header

- Highlights Panel
    - Status
    - Appointment Date
    - Doctor
    - Patient

▨ Left Column

- Record Details
- Activity Timeline

▨ Right Column

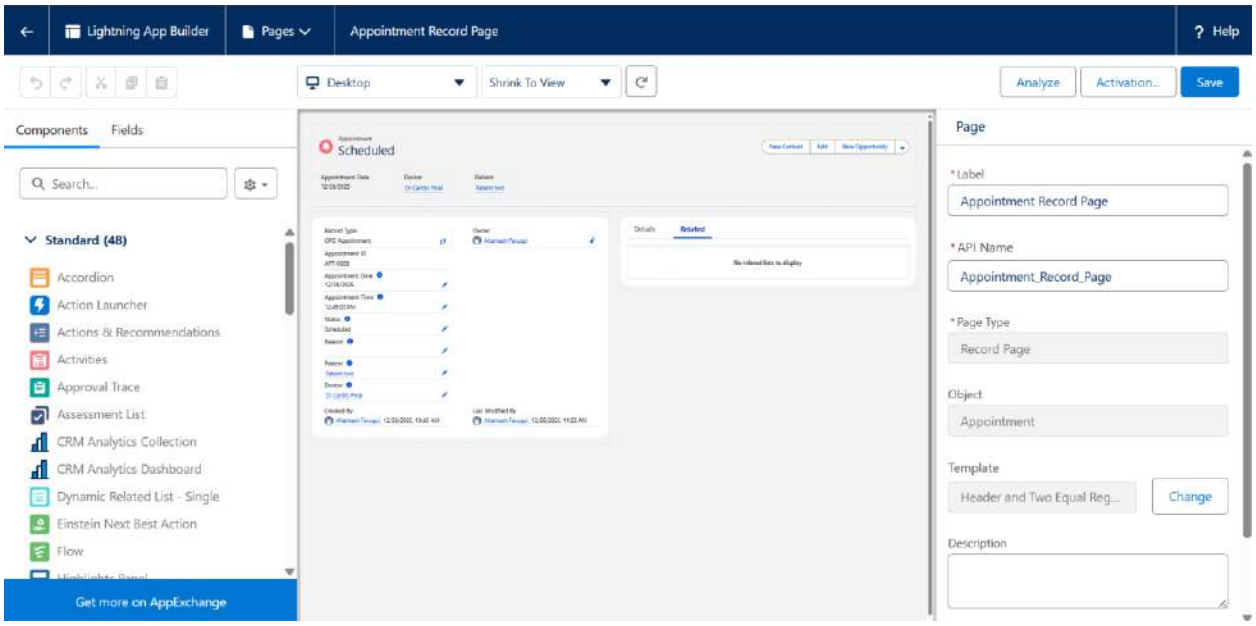
- Related Lists
    - Doctor
    - Patient

🔔 Optional Automation Component

- Add **Path** (Status)
    - Scheduled → Completed → Cancelled

Activate for:

- Receptionist
- Hospital Manager



Label: Appointment Record Page
API Name: Appointment_Record_Page
Page Type: Record Page
Object: Appointment
Template: Header and Two Equal Reg...
Description:

◇ STEP 6.5 — Home Page (Hospital Dashboard Lite)

Navigation

Setup → Lightning App Builder → New → Home Page

Components to Add

Top Section

- Assistant

- Rich Text:

“Welcome to Hospital Management System”

📈 *Middle Section*

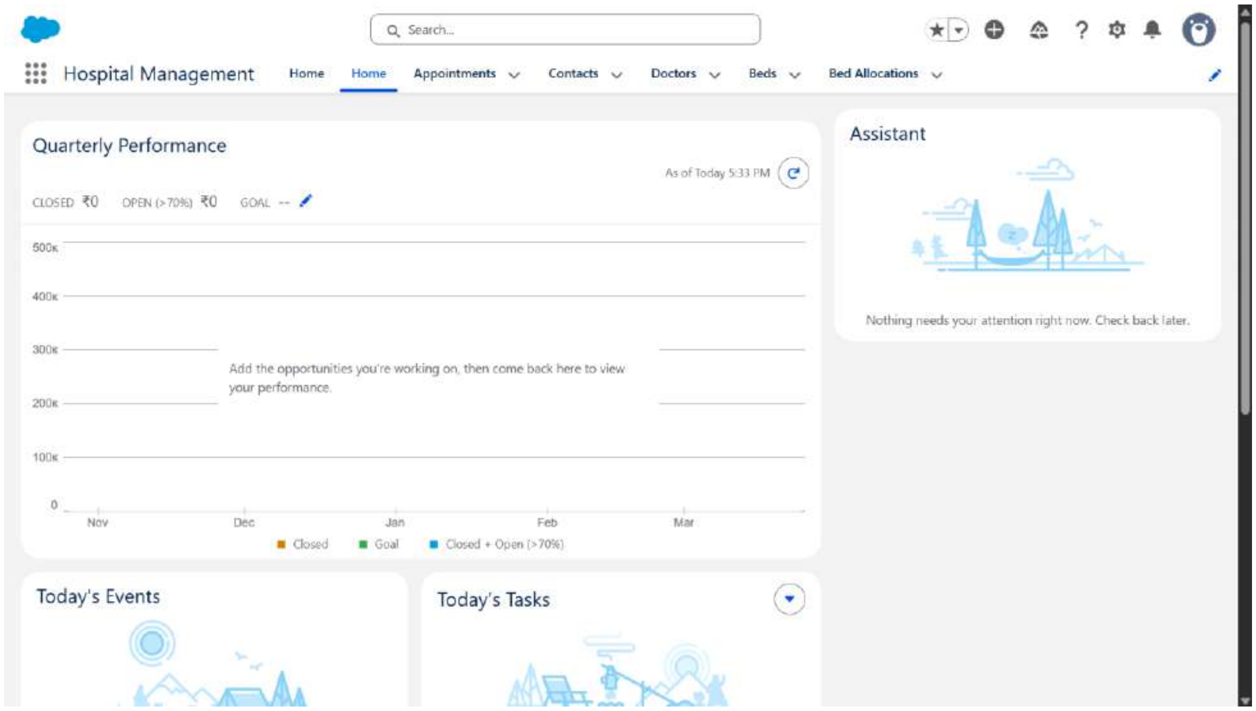
- Report Chart:
  - Today’s Appointments
  - Available Beds

🗒 *Bottom Section*

- To-Do List
- Recent Appointments

👥 Assign Home Page

- App: Hospital Management
- Profiles:
  - Receptionist
  - Hospital Manager



◇ STEP 6.6 — Utility Bar (POWER FEATURE)

Navigation

App Manager → Hospital Management → Edit → Utility Items

Add Utilities

1 Notes

Quick patient notes

2 Recent Items

Fast access to Doctors/Appointments

6 Lightning Web Components (LWC)

Purpose

Build **reusable**, **responsive**, and **high-performance** UI components in Salesforce.

Structure

File	Description
`.js`	Component logic
`.html`	UI template
`.js-meta.xml`	Metadata & exposure settings

Steps

1. Open **VS Code** → SFDX Project
2. Create **Lightning Web Component**
3. Implement business logic & styling
4. Deploy using **SFDX: Deploy**

Notes

- LWCs can be used in:
  - Lightning App Builder
  - Record Pages
  - Tabs
  - Utility Bar

7 Apex with LWC

Purpose

Fetch or manipulate **Salesforce data** using Apex from LWC.

🛠 Steps

1. Write Apex methods using `@AuraEnabled`
2. Call Apex from LWC using:
    a. **Wire Adapter** → Reactive data
    b. **Imperative Call** → User-triggered actions
3. Handle **responses & errors** in LWC JavaScript

8️⃣ Events in LWC

🎯 Purpose

Enable **communication between components**.

🔁 Types

Event Type	Use Case
Custom Events	Child → Parent communication
Lightning Message Service	Cross-component communication
Standard DOM Events	Click, Change, Input

🛠 Steps

1. Create and dispatch `CustomEvent` in child component
2. Listen in parent using `on<eventname>` attribute

9️⃣ Wire Adapters

🎯 Purpose

Retrieve Salesforce data **reactively**.

Examples

```
@wire(getRecord, { recordId, fields })
@wire(getObjectInfo, { objectApiName })
```

Notes

- Automatically refreshes UI when data changes
- Best suited for **read-only operations**

Imperative Apex Calls

Purpose

Call Apex methods **on demand** (user actions).

Steps

1. Import Apex method in LWC JS
2. Call method inside a function (e.g., button click)
3. Handle Promise:

```
myApexMethod({ param1: value })
  .then(result => {
    // handle result
  })
  .catch(error => {
    // handle error
  });
```

Notes

- Best for **user-triggered actions**
- Provides **explicit control** over execution

☑ Best Practices

✓ Keep LWCs **modular and reusable**

✓ Always handle **errors** in Apex calls

✓ Use **Profiles / Permission Sets** for access control

✓ Test Home & Record Pages across multiple profiles

✓ Deploy using **Change Sets or SFDX** for version control

★ Main Focus Areas

I mainly concentrated on:

- **Lightning App Builder**
- **Record Pages**
- **Tabs**
- **Home Page Layouts**
- **Utility Bar**

Phase 8: Data Management & Deployment

This phase focuses on Salesforce data management and deployment strategies.

The **primary emphasis is on the Data Import Wizard**, which we used to upload **Doctor records** for your project.

Other data tools are summarized for awareness.

1️⃣ Data Import Wizard (Main Focus)

Purpose:

Import data into Salesforce using a guided interface—ideal for uploading your **Doctor records CSV file** for custom objects referencing the Doctor object.

Steps to Import Doctor Records:

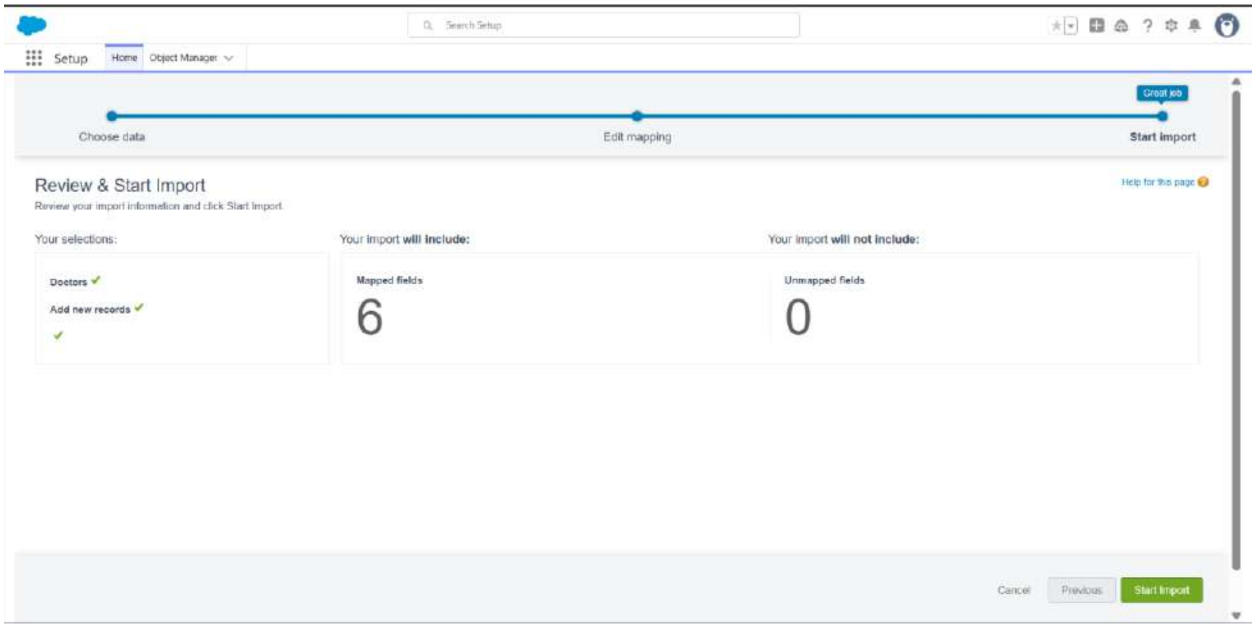
1. Navigate to **Setup → Data Import Wizard**
2. Select your **Doctor custom object**
3. Click **Upload CSV** and select the file (`doctor_records_updated.csv`)
4. **Map CSV fields** to Salesforce fields
    a. *Example:* `Doctor Name → Doctor_Name__c`
5. Click **Start Import** and monitor progress until complete

Supported Fields in Your Doctor Object

CSV Header	Salesforce Field (example)
Availability Status	Availability_Status__c
Consultation Fee	Consultation_Fee__c
Doctor Name	Doctor_Name__c
Experience (Years)	Experience_Years__c
Phone Number	Phone_Number__c
Specialization	Specialization__c

Notes

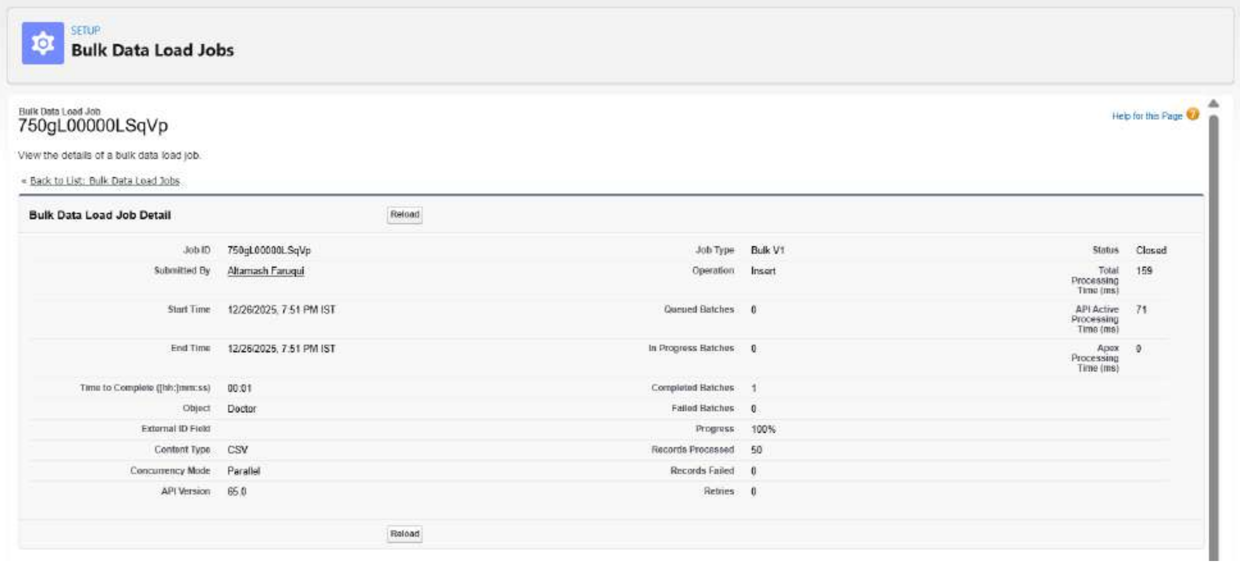
- Supports both **standard and custom objects**
- Best suited for **small-to-medium data volumes**
- Provides **automatic field matching** when headers are correct
- Can **prevent duplicate uploads** when duplicate rules are active



Review & Start Import

Review your import information and click Start Import.

Your selections:	Your import **will include:**	Your import **will not include:**
Doctors ✓	Mapped fields	Unmapped fields
Add new records ✓	6	0
✓		



Bulk Data Load Jobs

Bulk Data Load Job

750gL00000LSqVp

View the details of a bulk data load job.

Bulk Data Load Job Detail

Job ID	750gL00000LSqVp	Job Type	Bulk V1	Status	Closed
Submitted By	Altamash Faruqui	Operation	Insert	Total Processing Time (ms)	159
Start Time	12/26/2025, 7:51 PM IST	Queued Batches	0	API Active Processing Time (ms)	71
End Time	12/26/2025, 7:51 PM IST	In Progress Batches	0	Apex Processing Time (ms)	0
Time to Complete ([hh:]mm:ss)	00:01	Completed Batches	1		
Object	Doctor	Failed Batches	0		
External ID Field		Progress	100%		
Content Type	CSV	Records Processed	50		
Concurrency Mode	Parallel	Records Failed	0		
API Version	65.0	Retries	0		

2️⃣ Data Loader

Purpose:

Bulk data import/export tool for **large data volumes**

Quick Points

- Requires **installation** and Salesforce login
- Operations: **Insert, Update, Upsert, Delete, Export**
- Supports **millions of records**
- Requires **API access** and correct CSV mappings

3️⃣ Duplicate Rules

Purpose:

Prevent duplicate records—important when importing Doctors to avoid duplicate entries

Quick Points

- Setup path: **Duplicate Management → Duplicate Rules**
- Define **object, matching fields, and actions (Alert/Block)**
- Can use **custom Matching Rules**
- Maintains **clean, high-quality data**

4️⃣ Data Export & Backup

Purpose:

Backup Salesforce data regularly

Quick Points

- Go to **Setup → Data Export**
- Schedule backups **weekly or monthly**
- Downloads provided as **ZIP files containing CSV data**
- Third-party automation tools available

5️⃣ Change Sets

Purpose:

Deploy metadata between orgs (e.g., Sandbox → Production)

Quick Points

- Add components to **Outbound Change Set**
- Upload to target org → deploy via **Inbound Change Set**
- **Validate before activation**
- Works only between **connected orgs**

6️⃣ Unmanaged vs Managed Packages

Purpose:

Package and distribute Salesforce metadata

Quick Points

Unmanaged Package	Managed Package
Code editable	Code protected
One-time deployment	Designed for distribution
Not updateable	Updateable & versioned
Not AppExchange friendly	Required for AppExchange

7️⃣ ANT Migration Tool

Purpose:

Command-line utility for metadata deployment/retrieval

Quick Points

- Requires **build.xml** + **package.xml**
- Commands: `ant retrieve`, `ant deploy`
- Logs help troubleshoot success/failure
- Useful in **automation / CI-CD**

8️⃣ VS Code & SFDX

Purpose:

Salesforce development & deployment via **VS Code + CLI**

Quick Points

- Install **VS Code + Salesforce Extension Pack**
- Authenticate orgs via **SFDX CLI**
- Execute metadata commands using **SFDX**
- Supports **Apex, LWC, version control, and automation**

Phase 9 — Reports & Dashboards

Phase Objective

Phase 9 focuses on building **operational intelligence** for the Hospital Management System using **Salesforce Reports and Dashboards**. This phase converts raw hospital data (Doctors, Appointments, Beds) into **visual, decision-ready insights** for admins and hospital staff.

Key Outcomes

- Real-time visibility of **doctor availability**
- Monitoring of **daily appointments**
- Live tracking of **bed availability & occupancy**
- Centralized **Hospital Overview Dashboard**

Reports Created

1 Available Doctors Report

Purpose: Identify doctors currently available for appointments.

Report Type: Doctor

Filters Applied:

- Availability Status = Available

Fields Used:

- Doctor Name
- Specialization
- Availability Status

Usage:

- Feeds the *Available Doctors by Specialization* chart
- Helps front desk assign doctors efficiently

2️⃣ Unavailable Doctors Report

Purpose: Track doctors who are not currently available.

Report Type: Doctor

Filters Applied:

- Availability Status = Unavailable

Usage:

- Operational awareness
- Staff planning and shift management

3️⃣ New Appointments Report

Purpose: Display all appointments scheduled for today.

Report Type: Appointment

Filters Applied:

- Appointment Date = TODAY

Fields Used:

- Appointment ID
- Appointment Date
- Patient
- Doctor
- Status

Usage:

- Powers Today’s Appointment Status table
- Used for daily hospital operations

4️⃣ Available Beds Report

Purpose: Show count of beds currently available.

Report Type: Bed

Filters Applied:

- Availability Status = Available

Usage:

- Feeds Available Beds Gauge chart
- Helps emergency & admission teams

5️⃣ Occupied Beds Report

Purpose: Track beds currently occupied by patients.

Report Type: Bed

Filters Applied:

- Availability Status = Occupied

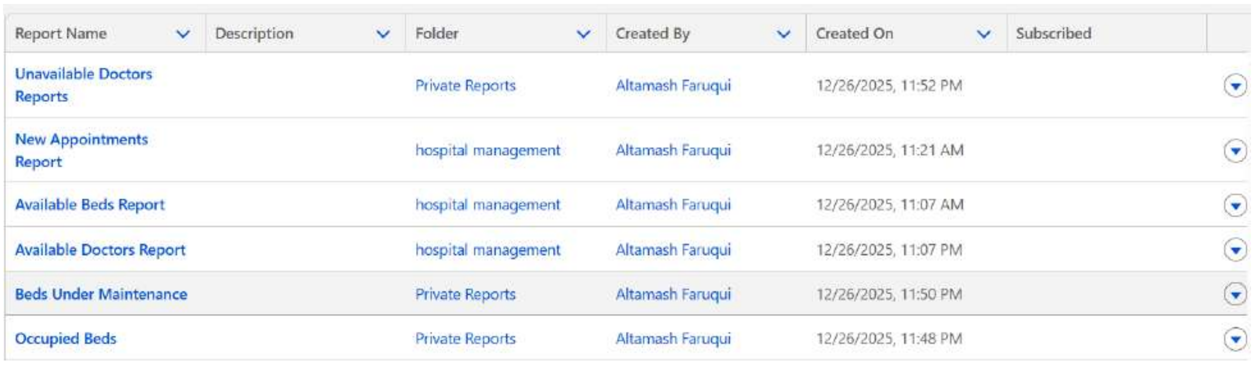
6️⃣ Beds Under Maintenance Report

Purpose: Identify beds unavailable due to maintenance.

Report Type: Bed

Filters Applied:

- Availability Status = Maintenance



Report Name	Description	Folder	Created By	Created On	Subscribed
Unavailable Doctors Reports		Private Reports	Altamash Faruqui	12/26/2025, 11:52 PM	
New Appointments Report		hospital management	Altamash Faruqui	12/26/2025, 11:21 AM	
Available Beds Report		hospital management	Altamash Faruqui	12/26/2025, 11:07 AM	
Available Doctors Report		hospital management	Altamash Faruqui	12/26/2025, 11:07 PM	
Beds Under Maintenance		Private Reports	Altamash Faruqui	12/26/2025, 11:50 PM	
Occupied Beds		Private Reports	Altamash Faruqui	12/26/2025, 11:48 PM	

📊 Dashboards Created

🧩 Hospital Overview Dashboard

Dashboard Name: Hospital Overview Dashboard

Description:

Provides a centralized, real-time snapshot of hospital operations including doctors, appointments, and beds.

📈 Dashboard Components

1️⃣ Available Doctors by Specialization

- **Component Type:** Donut Chart
- **Source Report:** Available Doctors Report
- **Grouped By:** Specialization

Insight Provided:

- Shows doctor distribution across departments

- Identifies specialization shortages instantly

2️⃣ Today's Appointments Status

- **Component Type:** Table
- **Source Report:** New Appointments Report

Displayed Columns:

- Appointment ID
- Appointment Date
- Patient
- Doctor

Insight Provided:

- Live appointment tracking
- Quick access to today's schedule

3️⃣ Today Appointment Count

- **Component Type:** Metric
- **Source Report:** New Appointments Report

Insight Provided:

- Total appointments for the day
- Helps workload planning

4️⃣ Available Beds

- **Component Type:** Gauge Chart
- **Source Report:** Available Beds Report

Gauge Ranges:

- Red: Low availability
- Yellow: Medium availability
- Green: Healthy availability

Insight Provided:

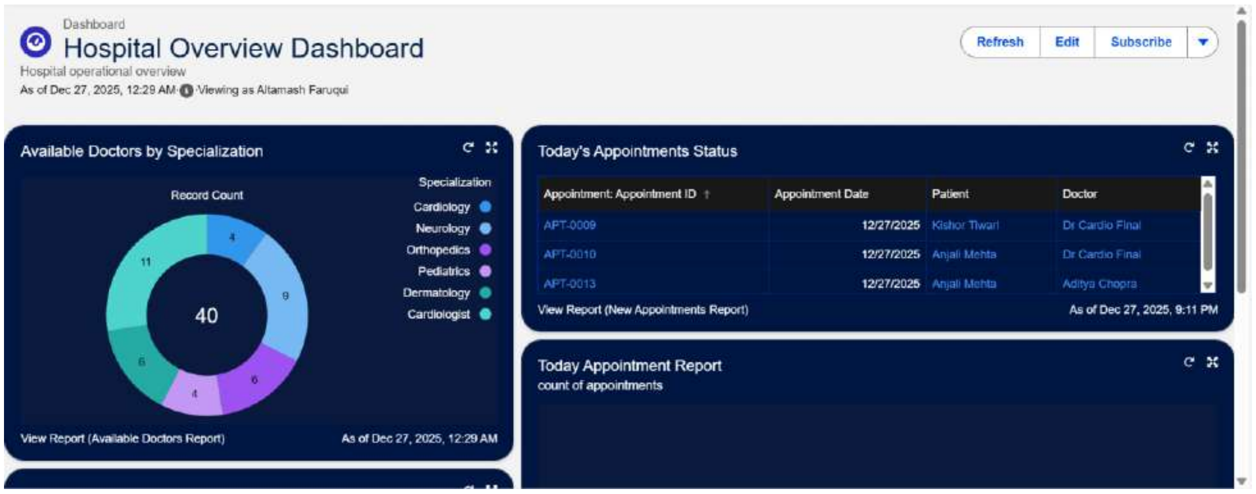
- Immediate understanding of bed capacity
- Critical for emergency response

Design Considerations

- Dark theme for **visual clarity & focus**
- KPI-driven layout
- Real-time data refresh
- Report-driven dashboard architecture

Security & Access

- Reports stored in **Hospital Management** folder
- Private reports used where needed
- Dashboard visibility controlled via Salesforce sharing


Dashboard

Hospital Overview Dashboard

Hospital operational overview

As of Dec 27, 2025, 12:29 AM · Viewing as Altamash Faruqui

Available Doctors by Specialization

View Report (Available Doctors Report) As of Dec 27, 2025, 12:29 AM

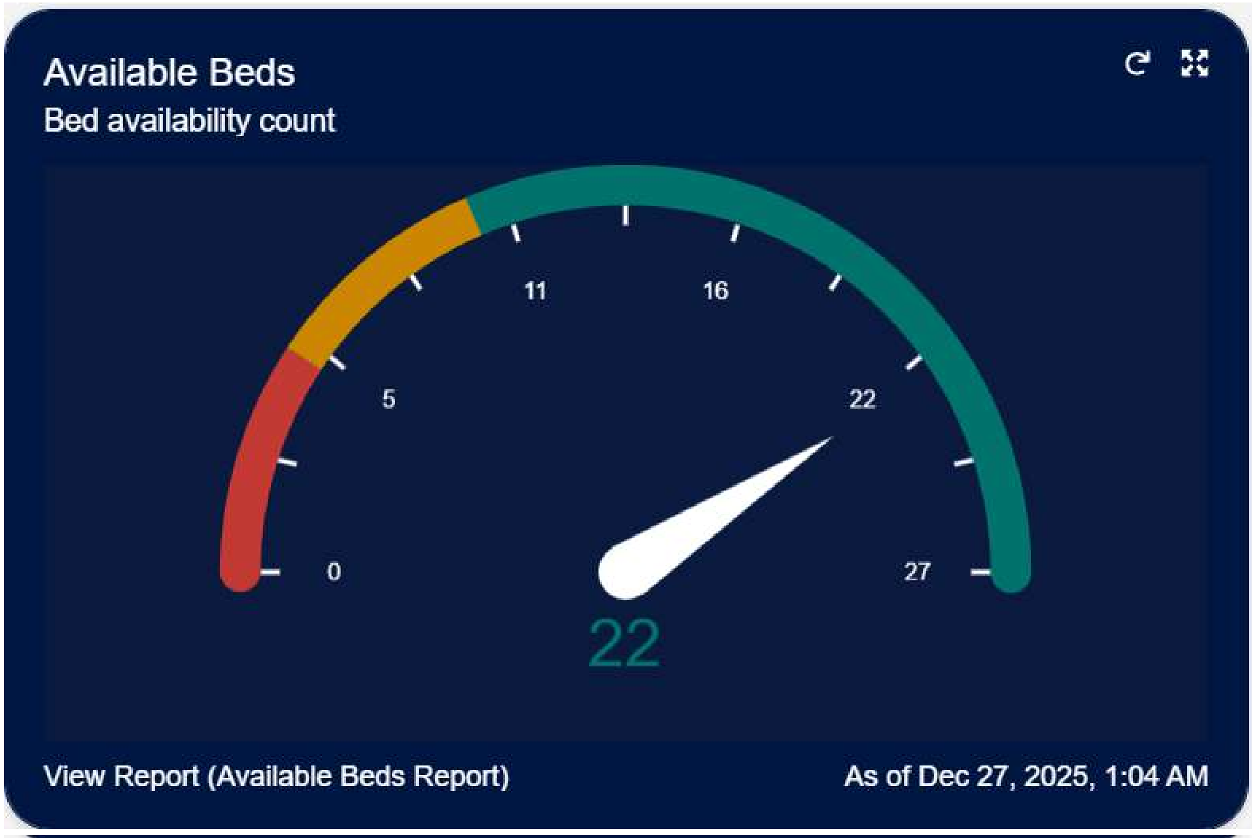
Today's Appointments Status

Appointment: Appointment ID ↑	Appointment Date	Patient	Doctor
APT-0009	12/27/2025	Kishor Tiwari	Dr Cardio Final
APT-0010	12/27/2025	Anjali Mehta	Dr Cardio Final
APT-0013	12/27/2025	Anjali Mehta	Aditya Chopra

View Report (New Appointments Report) As of Dec 27, 2025, 9:11 PM

Today Appointment Report

count of appointments

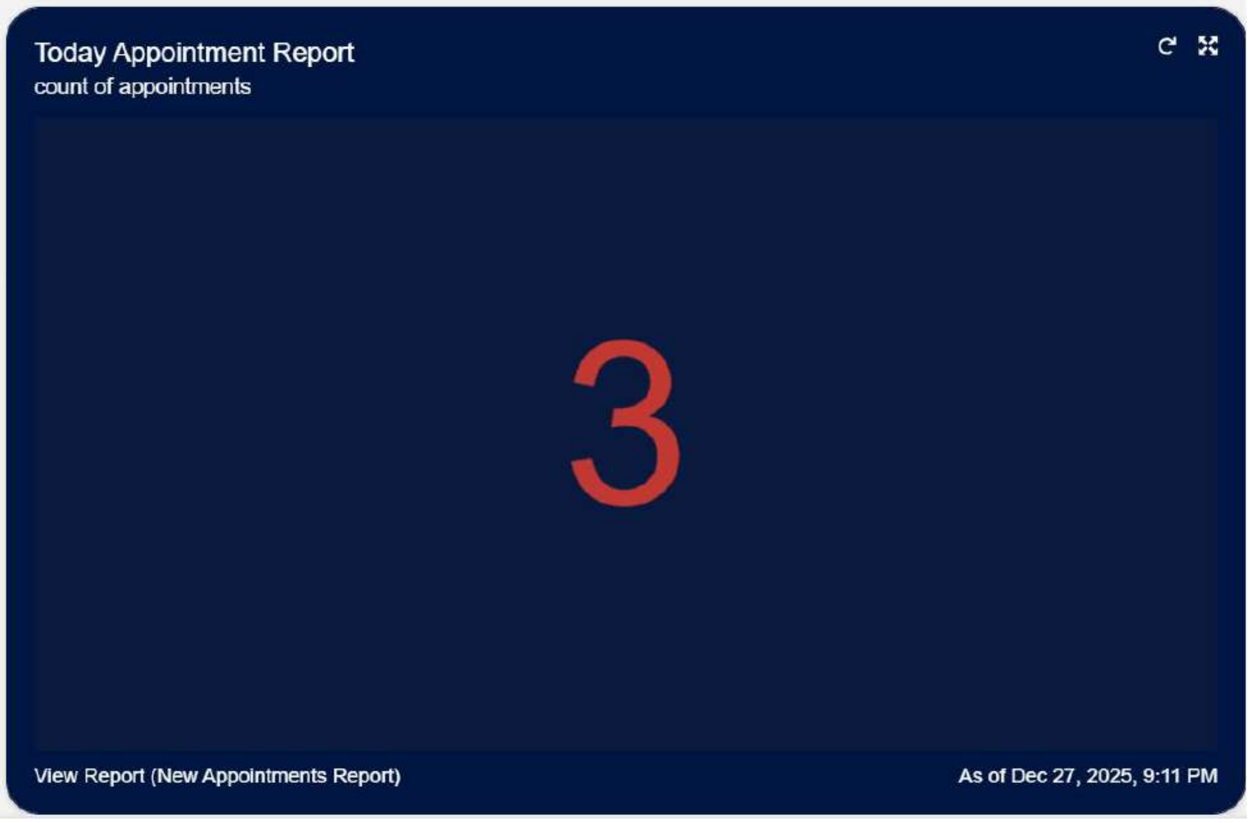


Available Beds

Bed availability count

View Report (Available Beds Report)

As of Dec 27, 2025, 1:04 AM



Today Appointment Report

count of appointments

View Report (New Appointments Report)

As of Dec 27, 2025, 9:11 PM

☑ Phase 9 Completion Status

✓ All core reports created

✓ Dashboard components configured correctly

✓ Data reflecting real-time records

✓ Ready for final submission & demo