

Salesforce Project Phase-1

Healthcare Patient Management and Telemedicine CRM

Phase 1: Problem Understanding & Industry Analysis

❖ Requirement Gathering:

- **Patients:**
 1. Easy registration & onboarding
 2. Online/offline appointment booking
 3. Telemedicine consultation
 4. Access to prescriptions & medical history
- **Doctors:**
 1. Patient record management
 2. Appointment scheduling
 3. Add or update prescriptions
 4. View consultation history
- **Hospital Admins:**
 1. Manage doctors, patients, and staff
 2. Billing and reports
 3. Security and compliance with medical standards

❖ Stakeholder Analysis:

- **Primary Stakeholders:** Patients, Doctors, Hospital Admin
- **Secondary Stakeholders:** Receptionists or office staff, Pharmacy, Insurance providers

❖ Business Process Mapping:

Flow chart: Patient Registration -> Appointment Booking -> Doctor Consultation
-> Prescription Generation -> Follow-up and Billing -> Reports and Analytics

❖ Industry-Specific Use Case Analysis:

Problem in Healthcare:

- Hospitals struggle with fragmented patient records, making it hard for doctors to see a complete medical history.
- Long waiting times for appointments due to manual scheduling.
- Lack of centralized telemedicine support, forcing patients to use external apps.
- No real-time hospital bed availability tracking, especially during emergencies

In Salesforce how can we solve it:

- Patient 360 View → A unified record for each patient with history, prescriptions, and lab reports.
- Automated Appointment Scheduling → Flows & Process Builder to reduce waiting time.
- Telemedicine Integration → Embed Zoom or Teams calls within Salesforce for online consultations.
- Bed Availability Management → Real-time updates via dashboards so doctors/admins see availability instantly.

❖ **AppExchange Exploration:**

- **Health Cloud:** inspiration for patient management
- **Zoom or Teams Integration Apps:** for telemedicine video calls
- **DocuSign for Salesforce:** for digital prescriptions & approvals
- **Payment Gateway Apps:** for billing and insurance payments

Salesforce Project Phase-2

Health Patient Management and Telemedicine CRM

Phase 2: Org setup & Configuration

❖ Salesforce Editions:

- Login into the existing salesforce org.
- Click on the Setup.
- In the Quick Find box, type Company Information and click on it.
- Look for the field Organization Edition like Developer Edition.

❖ Company Profile Setup:

- Login to existing Salesforce org.
- Click on the Setup.
- In the Quick Find box, type Company Information and click on it.
- On the Company Information page, click on edit.
- Fill the details as below:
 1. Organization Name: HealthCare Plus Telemedicine Pvt. Ltd
 2. Primary Contact: Anusha
 3. Street: Hitech City
 4. City: Hyderabad
 5. State: Telangana
 6. Country: India
 7. Default Locale: English (India)
 8. Default Currency: INR
 9. Default Time Zone: (GMT+05:30) India Standard Time (Asia/Kolkata)
 10. Click Save.

The screenshot shows the Salesforce Setup interface. On the left, the navigation sidebar is open, showing categories like Objects and Fields, Custom Code, Lightning Components, Company Settings, and Company Information (which is selected). The main content area is titled "Company Information" and displays the profile for "HealthCare Plus Telemedicine Pvt.Ltd". It includes sections for Organization Detail, Business Hours, and various system statistics. The organization's address is listed as "Hitech City, Hyderabad, Telangana, India". The business hours start in April. System statistics include 396 KB used data space and 17 KB used file space.

❖ Business Hours & Holidays Setup:

1. Business Hours:

- Login to existing Salesforce org.
- Click on Setup.
- In the Quick Find box, type Business Hours and click on the Business Hours.
- Click New.
- Fill the details:
 1. Business Hours Name: Hospital 24*7 Hours
 2. Time Zone: (GMT+05:30) India Standard Time (Asia/Kolkata)
 3. Enable the Active.
 4. Set to all days that is Monday to Sunday.
 5. Click Save.

The screenshot shows the Salesforce Setup interface. The navigation sidebar is open, showing categories like Objects and Fields, Custom Code, Lightning Components, Company Settings (with Business Hours selected), and Company Information. The main content area is titled "Business Hours" and displays the "Organization Business Hours" section. It lists two business hours entries: "Default" (Active, Pacific Daylight Time (America/Los_Angeles)) and "Hospital 24*7 Hours" (Active, (GMT+05:30) India Standard Time (Asia/Kolkata)).

2. Holidays:

- Login to existing Salesforce org.
- Click on Setup.
- In the Quick Find box, type Holidays and click on the Holidays.
- Click New.
- Fill the details for 5 holidays:
 1. Holiday Name: New Year
Description: only for emergency cases
Date: 1/1/2025
Enable the Recurrence
Business Hours: All Days
Click Save
 2. Holiday Name: Republic Day
Description: only for emergency cases
Date: 1/26/2025
Enable the Recurrence
Business Hours: All Days
Click Save
 3. Holiday Name: Independence Day
Description: only for emergency cases
Date: 8/15/2025
Enable the Recurrence
Business Hours: All Days
Click Save
 4. Holiday Name: Gandhi Jayanti
Description: only for emergency cases
Date: 10/2/2025
Enable the Recurrence
Business Hours: All Days
Click Save
 5. Holiday Name: Christmas Day
Description: only for emergency cases
Date: 12/25/2025
Enable the Recurrence
Business Hours: All Days
Click Save

Holidays

Holidays are dates and times at which business hours are suspended. Business hours are the days and hours that your support team is available.

Action	Holiday Name	Description	Date and Time
Edit Del	Christmas Day	Only emergency services available	12/25/2025 All Day
Edit Del	Gandhi Jayanti	Only emergency services available	10/2/2025 All Day
Edit Del	Independence Day	Only emergency services available	8/15/2025 All Day
Edit Del	New year	Only emergency services available	1/1/2026 All Day
Edit Del	Republic Day	Only emergency services available	1/26/2026 All Day

Elapsed Holidays

No records to display

❖ Fiscal Year Settings:

- Login to existing salesforce org.
- Click on the Setup.
- In the Quick Find box, type Fiscal Year and click on it.
- Fiscal Year Start Month: April
- Select the starting month.
- Click Save.

Fiscal Year

Organization Fiscal Year Edit: HealthCare Plus Telemedicine Pvt.Ltd

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.

Change Fiscal Year Period

Name: HealthCare Plus Telemedicine Pvt Ltd

Fiscal Year Start Month: **April**

Fiscal Year Is Based On: The starting month

❖ **User Setup & Licenses:**

1. Check how many Licenses:

- In the Quick Find box, type Company Information and click it.
- Look under the User Licenses.

2. User Setup:

Create Roles:

1. In the Quick Find box, type Roles and click it.

2. Click on Setup Roles.

3. Click on Add Role to create a new Role.

4. Fill the details:

- Label: HealthCare Admin
Role Name: HealthCare_Admin
Click on save
- Label: Doctor/Specialist
Role Name: Doctor_Specialist
Click on save
- Label: Patient Coordinator
Role Name: Patient_Coordinator
Click on Save

Create Users:

1. In the Quick Find box, type Users and click it.

2. Click on New user.

3. Fill the details:

- First Name: Anusha
Last Name: Peetha
Alias: anu
Email: anusha1901@gmail.com
License: Salesforce
Profile: System Admin
Click on save and new
- First Name: Dr. Ramesh
Last Name: Shaik
Alias: Ramesh
Email: ramesh1902@gmail.com

License: Salesforce Integration

Profile: Standard User

Click on save and new

- First Name: Priya
Last Name: Kamala
Alias: fiya
Email: priya1903@gmail.com
License: Salesforce Integration
Profile: Standard User
Click on save

❖ **Profiles:**

- Go to Setup.
- In the Quick Find box, type the Profiles and then click on it.
- HealthCare Admin Profile:
 - Find the System Administrator and click on clone.
 - Profile Name: HealthCare Admin Profile
 - Click save.
- Doctor Profile:
 - Find the Standard User and click on clone.
 - Profile Name: Doctor Profile
 - Click save.
 - Click on edit.
 - In the Object Permissions give access to Patients, Appointments, Telemedicine (Read, Create, Edit).
 - Click save.
- Patient Coordinator Profile:
 - Find the Standard User and click on clone.
 - Profile Name: Patient Coordinator Profile
 - Click save.
 - Click on edit.
 - In the Object Permissions give access to Patients, Appointments, Bed Availability (Read, Create, Edit), Telemedicine Records (Read).
 - Click on save.

❖ **Roles:**

- In the Quick Find box, type Roles and click it.

- Click on Setup Roles.
- Click on Add Role to create a new Role.
- Fill the details:
 - Label: HealthCare Admin
Role Name: HealthCare_Admin
Click on save
 - Label: Doctor/Specialist
Role Name: Doctor_Specialist
Click on save
 - Label: Patient Coordinator
Role Name: Patient_Coordinator
- Click on Save

❖ Permission Sets:

- Click on the Setup.
- In the Quick Find box, type the Permission sets and click on it.
- Click on New.
- Fill the details:
 - Label: Doctor Extra Access
 - API Name: Doctor_Extra_Access
 - User License: Salesforce
- Click on save.
- Now, open the Permission Set just created.
- Go to the Object Setting and select the object in the drop box Patients, Appointments, Telemedicine Records, Bed Availability.
- Click on edit.
- In object settings, enable the
 - Patients: Read, Create, Edit
 - Appointments: Read, Create, Edit
 - Telemedicine Records: Read, Create, Edit
 - Bed Availability: Read
- Click save.
- Now, click on the Manage Assignments and Add Assignments.

- Select Anusha Peetha and click Assign and then Done.

The screenshot shows the Salesforce Setup interface with the 'Permission Sets' section selected. A specific permission set named 'Doctor Extra Access' is being viewed. The 'Current Assignments' table displays one assignment for the user 'Anusha Peetha'. The assignment details are as follows:

Full Name	Active	Role	Profile	User License	Expires On
Anusha Peetha	✓	HealthCare Admin	System Administrator	Salesforce	2023-12-31

❖ OWD:

- In the Quick Find box, type sharing settings and click on it.
- Scroll down to the Organization Wide Section and click on edit.
- Change the default internal access for the following objects:
Patients - Private, Appointments – Private, Bed Availability – Public Read only,
Telemedicine Records – Private, Payments - Private
- Click save.

The screenshot shows the Salesforce Setup interface with the 'Sharing Settings' section selected under 'Security'. The 'Sharing Settings' table lists sharing rules for various objects:

Object	Sharing Rule Type	Access Level	Control By
Appointments	Sharing Rule	Public Read/Write	Private
Bed Availability	Sharing Rule	Public Read/Write	Private
Mentor	Sharing Rule	Public Read/Write	Private
orderDetail	Sharing Rule	Public Read/Write	Private
Patients	Sharing Rule	Public Read/Write	Private
Student	Sharing Rule	Controlled by Parent	Controlled by Parent
Telemedicine Records	Sharing Rule	Public Read/Write	Private

The 'Other Settings' section contains the following configuration:

- Manager Groups: []
- Secure guest user record access: ✓ []
- Require permission to view record names in lookup fields: []

Sharing Rules sections for Lead, Account, and Opportunity are shown with no rules specified.

❖ **Sharing Rules:**

- Click on the Setup.
- In the Quick Find box, type the Sharing settings and click on it.
- Scroll down to the Sharing Rules.
- Patients:
 - Label: Doctors Access
 - Rule Name: Doctor_Access
 - Select your rule Type: Based on Role
 - Select which records to be shared: Roles, Doctor
 - Select the users to share with: Roles, Doctor
 - Select the level of access for the users: Read/write
 - Click save.
- Appointments:
 - First you need create Appointment_status_c for that you need to go to object manager and then click on the appointments.
 - Now, then click on the Fields & Relationships.
 - Click on New and select Text and then next.
 - Field Label: Appointment_status_c, Length: 10, Field Name: Appointment_status_c, click next and again next and then Save.
 - Label: Coordinator Access
 - Rule Name: Coordinator_Access
 - Select your rule Type: Based on criteria
 - Select which records to be shared: Appointment_status_c, equals, Scheduled
 - Select the users to share with: Roles, Patient Coordinator
 - Select the level of access for the users: Read only
 - Click Save.
- Bed Availability:
 - First you need create bed_status_c for that you need to go to object manager and then click on the Bed Availability.
 - Now, then click on the Fields & Relationships.
 - Click on New and select Text and then next.
 - Field Label: bed_status_c, Length: 10, Field Name: bed_status_c, click next and again next and then Save.
 - Label: Beds_share_staff
 - Rule Name: Beds_share_staff
 - Select your rule Type: Based on criteria
 - Select which records to be shared: Appointment_status_c, equals, Available

- Select the users to share with: Roles, Patient Coordinator
 - Select the level of access for the users: Read only
 - Click save.
- Telemedicine Records:
- First, you need to create tele_status_c for that you need to go to object manager and then click on the appointments.
 - Now, then click on the Fields & Relationships.
 - Click on New and select Text and then next.
 - Field Label: tele_status_c, Length: 10, Field Name: tele_status_c, click next and again next and then Save.
 - Second, you need to create public groups for that you need to go to public groups by typing in Quick Find box.
 - Now enter the details: Label: Doctors& Coordinators, Group Name: Doctors_Coordinator, Search: Roles, Available Members: Doctors and Patient Coordinator and then save.
 - Label: tele_coordinator
 - Rule Name: tele_coordinator
 - Select your rule Type: Based on criteria
 - Select which records to be shared: tele_status_c, equals, Scheduled
 - Select the users to share with: public groups, Doctors& Coordinators
 - Select the level of access for the users: Read/write
 - Click save.

❖ **Login Access Policies:**

- Click on the Setup.
- In the Quick Find box, type Login Access Policies and click on it.
- Check the box “Administrations can log in as any user”.
- Click save.
- Test as HealthCare Project users:
 - Go to Setup.
 - In Quick Find box, type the Users and click it.
 - Find the user you want to test: Doctor, Patient Coordinator, Patient, Bed availability
- Click login next to that user.
- You are now logged in as that user.
- Test their tabs, records, and permission.

❖ **Dev Org Setup:**

We created a Developer Org and set it up for our Healthcare Management project. We configured custom objects like Patients, Doctors, Appointments, Telemedicine Records,

and Bed Availability. Then we created a Healthcare Lightning App, assigned profiles and users, set Org-Wide Defaults, created sharing rules, and tested login access policies to ensure the right visibility and permissions.

❖ **Sandbox Usage:**

- HealthCare Patient Management and Telemedicine CRM project, since we are working with a Salesforce Developer Org Sandboxes are not available.
- Instead, the Developer Org itself acts as both Development and Testing environment.

❖ **Deployment Basics:**

- Sandbox is not available, so all development and testing are done directly in the main org.
- Custom objects like Patients, Doctors, Appointments, Telemedicine Records, Bed Availability are created and configured here.
- Profiles, Roles, OWD, Sharing Rules, and Login Access Policies are applied to simulate real-world hospital access control.
- Test Users are created to verify permissions, visibility, and workflow automation.

Salesforce Project Phase-3

Health Patient Management and Telemedicine CRM

Phase 3: Data Modeling & Relationships

❖ Standard and Custom Objects:

All the required standard and custom objects for the project have already been created in the previous phase.

Standard Objects: User

Custom Objects: Patients, Appointments, Telemedicine Records, Bed Availability, Payments

❖ Fields:

- Go to Setup.
- Click on the Object Manager, select the Patients.
- Now click on the Fields & Relationships and click on New.
- Patients:

Data Type	Field Name	Steps
Auto Number	Patient_ID	Auto Number -> Next -> Starts with :1 -> Next-> Next-> Next -> Next -> Save
Text	First_Name	Text -> Next -> Length: 50 -> Next-> Next -> Next -> Next -> Save
Text	Last_Name	Text -> Next -> Length: 50 -> Next-> Next -> Next -> Next -> Save
Date	Date_of_Birth	Date -> Next -> Next-> Next -> Next -> Next -> Save
Picklist	Gender	Picklist -> Next -> Enter values (Male, Female,

		Other)-> Next-> Next -> Next -> Next -> Save
Phone	Contact_Number	Phone -> Next -> Next-> Next -> Next -> Next-> Save
Email	Email	Email -> Next -> Length: 255 -> Next-> Next -> Next -> Next -> Save
Text Area	Address	Text Area -> Next -> Length: 255 -> Next-> Next -> Next -> Next -> Save
Long Text Area	Medical_History	Text -> Next -> Length: 500 -> Next-> Next -> Next -> Next -> Save

The screenshot shows the Salesforce Object Manager interface for the 'Patients' object. The left sidebar contains navigation links for Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The main content area is titled 'Fields & Relationships' and displays 13 items, sorted by Field Label. The listed fields include:

- Date_of_Birth
- Email
- First_Name
- Gender
- Last_Modified_By
- Last_Name
- Medical_History
- Owner
- Patient_ID
- Patients_ID

Each field is shown with its name, field label, and data type (e.g., Date, Email, Text(50), Picklist, Lookup(User), Long Text Area(500), Auto Number, Text(80)).

➤ Appointments Object:

Data Type	Field Name	Steps
Auto Number	Appointment_ID	Auto Number -> Next -> Starts with :1 -> Next-> Next-> Next -> Next -> Save
Lookup	Patient	Lookup-> Next -> Related to Patients ->

		Next-> Next -> Next -> Next -> Save
Lookup	Doctor	Lookup-> Next -> Related to User -> Next-> Next -> Next -> Next -> Save
Date/Time	Appointment__Date	Date/Time -> Next -> Next-> Next -> Next -> Next -> Save
Picklist	Status	Picklist -> Next -> Enter values (Scheduled, Completed, Cancelled)-> Next-> Next -> Next -> Next -> Save
Long Text Area	Notes	Text Area (Long) -> Next -> Length: 500-> Next-> Next -> Next -> Next -> Save

The screenshot shows the Salesforce Setup interface for the Appointments object. The left sidebar lists various configuration options: Details, Fields & Relationships (selected), Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The main content area displays the 'Fields & Relationships' section, which lists 12 items sorted by field label. Each item includes the field name, its type, and a dropdown menu for further actions.

Field Label	Type	Description
Appointment__Date	Date/Time	
Appointment_ID	Auto Number	
Appointment_Status	Picklist	
Appointment_Status_c	Text(10)	
Created_By	Lookup(User)	
Last_Modified_By	Lookup(User)	
Notes	Long Text Area(500)	
Owner	OwnerId	Lookup(User,Group)
Patients	Patient__c	Lookup(Patients)
Status	Status__c	Picklist
User	Doctor__c	Lookup(User)

➤ Telemedicine Records object:

Data Type	Field Name	Steps
Auto Number	Record__ID	Auto Number -> Next -> Starts with :1 -> Next->Next-> Next -> Next -> Save

Lookup	Patient	Lookup-> Next -> Related to Patients -> Next-> Next -> Next -> Next -> Save
Lookup	Doctor	Lookup-> Next -> Related to User -> Next-> Next -> Next -> Next -> Save
Date/Time	Consultation _Date	Date/Time -> Next -> Next-> Next -> Next -> Next -> Save
Long Text Area	Diagnosis	Text Area (Long) -> Next -> Length: 500 ->Next-> Next -> Next -> Next-> Save
Long Text Area	Prescription	Text Area (Long) -> Next -> Length: 500 ->Next-> Next -> Next -> Next-> Save
Long Text Area	Notes	Text Area (Long) -> Next -> Length: 500 ->Next-> Next -> Next -> Next-> Save

The screenshot shows the Salesforce Setup interface, specifically the Object Manager for the 'Telemedicine Records' object. The left sidebar contains navigation links for Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The main content area is titled 'Fields & Relationships' and displays a list of 12 items, sorted by Field Label. Each item includes the field name, its type, and a description. The fields listed are:

- Diagnosis: Diagnosis__c, Long Text Area(500)
- Last Modified By: LastModifiedById, Lookup(User)
- Notes: Notes__c, Long Text Area(500)
- Owner: OwnerId, Lookup(User,Group)
- Patients: Patient__c, Lookup(Patients)
- Prescription: Prescription__c, Long Text Area(500)
- Record ID: Name, Text(80)
- tele_status: tele_status__c, Picklist (Multi-Select)
- tele_status_c: tele_status_c__c, Text(10)
- User: Doctor__c, Lookup(User)

➤ Bed Availability object:

Data Type	Field Name	Steps
Auto Number	Bed_ID	Auto Number -> Next -> Starts with :1 -> Next->Next-> Next -> Next -> Save
Text	Ward	Text -> Next -> Length: 50 ->Next-> Next -> Next -> Next-> Save
Text	Bed_Number	Text -> Next -> Length: 10 ->Next-> Next -> Next -> Next-> Save
Picklist	Status	Picklist -> Next -> Enter values (Available, occupied, Maintenance)-> Next-> Next -> Next -> Next -> Save
Lookup	Patient_Assigned	Lookup-> Next -> Related to Patients -> Next-> Next -> Next -> Next -> Save
Long Text Area	Notes	Text Area (Long) -> Next -> Length: 500 ->Next-> Next -> Next -> Next-> Save

The screenshot shows the Salesforce Object Manager interface for the 'Bed Availability' object. The left sidebar lists various setup options like Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The main content area is titled 'Fields & Relationships' and displays 12 items, sorted by Field Label. The fields listed are:

Field Label	API Name	Type
Bed_ID	Bed_ID_c	Auto Number
Bed_Number	Bed_Number_c	Text[10]
bed_status	bed_status_c	Picklist
bed_status_c	bed_status_c_c	Text[10]
Created By	CreatedById	Lookup(User)
Last Modified By	LastModifiedById	Lookup(User)
Notes	Notes_c	Long Text Area[500]
Owner	OwnerId	Lookup(User/Group)
Patients	Patient_Assigned_c	Lookup(Patients)
Status	Status_c	Picklist

➤ Payments:

Data Type	Field Name	Steps
Lookup	Patient	Lookup -> Next -> Related to Patients -> Next->Next-> Next -> Next -> Save
Lookup	Appointment	Lookup -> Next -> Related to Appointments -> Next->Next-> Next -> Next -> Save
Currency	Amount	Text -> Next -> Length: 16, Decimal: 2->Next-> Next -> Next -> Next-> Save
Date	Payment Date	Next-> Next -> Next -> Save
Picklist	Payment Method	Picklist -> Next -> Enter values (Cash, Card, UPI, Insurance)-> Next-> Next -> Next -> Next -> Save
Text Area	Notes	Text Area -> Next ->Next-> Next -> Next -> Next-> Save
Picklist	Status	Picklist -> Next -> Enter values (Pending, Completed, Failed)-> Next-> Next -> Next -> Next -> Save

❖ Record Types:

- Go to Setup.
- Click On Object Manager and click on required object.
- Click on Record Type and then click New.

➤ Patients:

1. Inpatient:

- Record Type: Master
- Record Type Label: Inpatient
- Record Type Name: Inpatient
- Description: Patients admitted in hospital

- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

2. Outpatient:

- Record Type: Master
- Record Type Label: outpatient
- Record Type Name: outpatient
- Description: Patients visiting for consultation
- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

RECORD TYPE LABEL	DESCRIPTION	ACTIVE	MODIFIED BY
Inpatient	Patients admitted in hospital	✓	Anusha Peetha, 9/21/2025, 12:02 PM
Outpatient	Patients visiting for consultation	✓	Anusha Peetha, 9/21/2025, 12:04 PM

➤ Appointments:

1. Consultation:

- Record Type: Master
- Record Type Label: Consultation
- Record Type Name: Consultation
- Description: Appointment for consultation
- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

2. Follow-up:

- Record Type: Master
- Record Type Label: Follow-up
- Record Type Name: Follow-up
- Description: Follow-up appointments
- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

RECORD TYPE LABEL	DESCRIPTION	ACTIVE	MODIFIED BY
Consultation	Appointment for consultation	✓	Anusha Peetha, 9/21/2025, 12:05 PM
Follow_up	Follow-up appointments	✓	Anusha Peetha, 9/21/2025, 12:06 PM

➤ Telemedicine Records:

1. Video consultation:

- Record Type: Master
- Record Type Label: Video Consultation
- Record Type Name: Video _Consultation
- Description: Telemedicine via video call
- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

2. Phone Consultation:

- Record Type: Master
- Record Type Label: Phone Consultation
- Record Type Name: Phone_ Consultation

- Description: Telemedicine via phone call
- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

The screenshot shows the Salesforce Setup interface under the Object Manager. The left sidebar lists various configuration options like Details, Fields & Relationships, Page Layouts, etc., with 'Record Types' selected. The main content area is titled 'Record Types' and shows a table with two items:

RECORD TYPE LABEL	DESCRIPTION	ACTIVE	MODIFIED BY
Phone Consultation	Telemedicine via phone	✓	Anusha Peetha, 9/21/2025, 12:08 PM
Video Consultation	Telemedicine via video call	✓	Anusha Peetha, 9/21/2025, 12:07 PM

➤ Bed Availability object:

1. ICU:

- Record Type: Master
- Record Type Label: ICU
- Record Type Name: ICU
- Description: ICU beds
- Description: Telemedicine via video call
- Active: enable
- Select, Apply a different layout for each profile
- Click on save.

2. General Ward:

- Record Type: Master
- Record Type Label: General Ward

- Record Type Name: General_Ward
- Description: Regular ward beds
- Active: enable
- Select, Apply a different layout for each profile
- Click on save

RECORD TYPE LABEL	DESCRIPTION	ACTIVE	MODIFIED BY
General Ward	Regular ward beds	✓	Anusha Peetha, 9/21/2025, 12:10 PM
ICU	ICU beds	✓	Anusha Peetha, 9/21/2025, 12:09 PM

❖ Page Layouts:

- Go to Setup.
- Click on the Object Manager.
- Find the required object Patients, Appointments, Bed availability, Telemedicine Records, Payments and click on it.
- Find the Page Layouts in the left panel and click on it.
- Now click on edit on the default Page Layout.
- See that all the Fields are in the Information Section.
- If not in the Information Section drag the fields into the Information Section.

❖ Compact Layouts:

- Go to Setup.
- Click on the Object Manager.

- Find the required object Patients, Appointments, Bed availability, Telemedicine Records and click on it.
- Find the Compact Layouts in the left panel and click on it.
- Now click on New.
- Patients:
 - Label: Patients Compact Layout
 - Name: Patients _Compact _Layout
 - Fields to display: Patient _ID, Record Type, first _Name, Last _Name, Gender, Date _of_ Birth, Contact _Number, Email
 - Click on Save.
 - Now click on Compact Layout Assignment and click on edit.
 - In the two drop down, select the Patient Compact Layout.
 - Click on save.

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
Patients Compact Layout	Patients.Compact.Layout	✓	Anusha Peetha	9/21/2025, 7:15 PM
System Default	SYSTEM			

- Appointments:
 - Label: Appointment Compact Layout
 - Name: Appointment _Compact _Layout
 - Fields to display: Appointment _ID, Record Type, Appointment _Date, Appointment _status _c, User, status
 - Click on Save.
 - Now click on Compact Layout Assignment and click on edit.
 - In the two drop down, select the Appointment Compact Layout.
 - Click on save.

Label	API Name	Primary	Modified By	Last Modified
Appointment Compact Layout	Appointment_Compact_Layout	✓	Anusha Peetha	9/21/2025, 7:18 PM
System Default	SYSTEM			

➤ Bed Availability:

- Label: Bed Availability Compact Layout
- Name: Bed_Availability_Compact_Layout
- Fields to display: Bed_ID, Record Type, Bed_Number, bed_status_c, Patients, status, ward
- Click on Save.
- Now click on Compact Layout Assignment and click on edit.
- In the two drop down, select the Bed Availability Compact Layout.
- Click on save.

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
Bed Availability Compact Layout	Bed.Availability_Compact_Layout	✓	Anusha Peetha	9/21/2025, 7:21 PM
System Default	SYSTEM			

➤ Telemedicine Records:

- Label: Telemedicine Compact Layout
- Name: Telemedicine _Compact _Layout
- Fields to display: Record ID, Record Type, Consultation _Date, tele _status _c, User
- Click on Save.
- Now click on Compact Layout Assignment and click on edit.
- In the two drop down, select the Telemedicine Compact Layout.
- Click on save.

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
System Default	SYSTEM			
Telemedicine Compact Layout	Telemedicine_Compact_Layout	✓	Anusha Peetha	9/21/2025, 7:23 PM

➤ Payments:

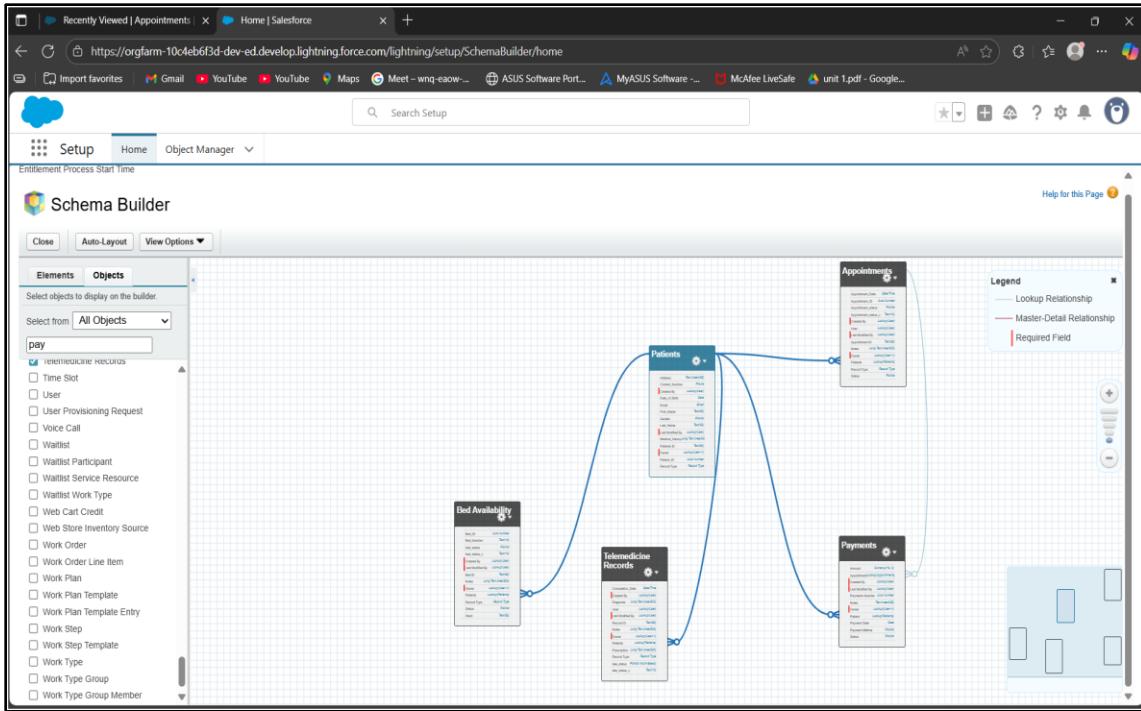
- Label: Payment Compact Layout
- Name: Payment _Compact _Layout
- Fields to display: Payment Number, Patient, Amount, Status
- Click on Save.
- Now click on Compact Layout Assignment and click on edit.
- In the two drop down, select the Payment Compact Layout.
- Click on save.

The screenshot shows the Salesforce Setup interface with the 'Object Manager' tab selected. Under the 'Payments' object, the 'Compact Layouts' section is displayed. The table shows two items:

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
Payments Compact Layout	Payments_Compact_Layout	✓	Anusha Peetha	9/22/2025, 12:59 AM
System Default	SYSTEM			

❖ Schema Builder:

- Go to Setup.
- In the Quick Find box, type the Schema Builder and click on it.
- In the left panel, enable the Objects:
Patients, Doctors, Appointments, Telemedicine Records, Bed Availability and Payments.
- Click on the Auto – layout.



❖ Lookup vs Master-Detail vs Hierarchical Relationships:

➤ Lookup Relationships:

- Patients → Appointments
Each patient can have multiple appointments. An appointment record looks up to the patient.
- Doctors → Appointments
Each appointment looks up to a doctor, but if the doctor record is deleted, the appointment can still exist.
- Patients → Telemedicine Reports
Telemedicine reports are linked to patients using a lookup relationship.

➤ Master – Detail Relationships:

- Appointments → Payments
A payment record depends on the appointment. If the appointment is deleted, the related payment record is also deleted.

➤ Hierarchical Relationship:

- User Object
A senior doctor can supervise junior doctors or staff members. This relationship type is only available on the User object and helps in approvals and hospital reporting structures.

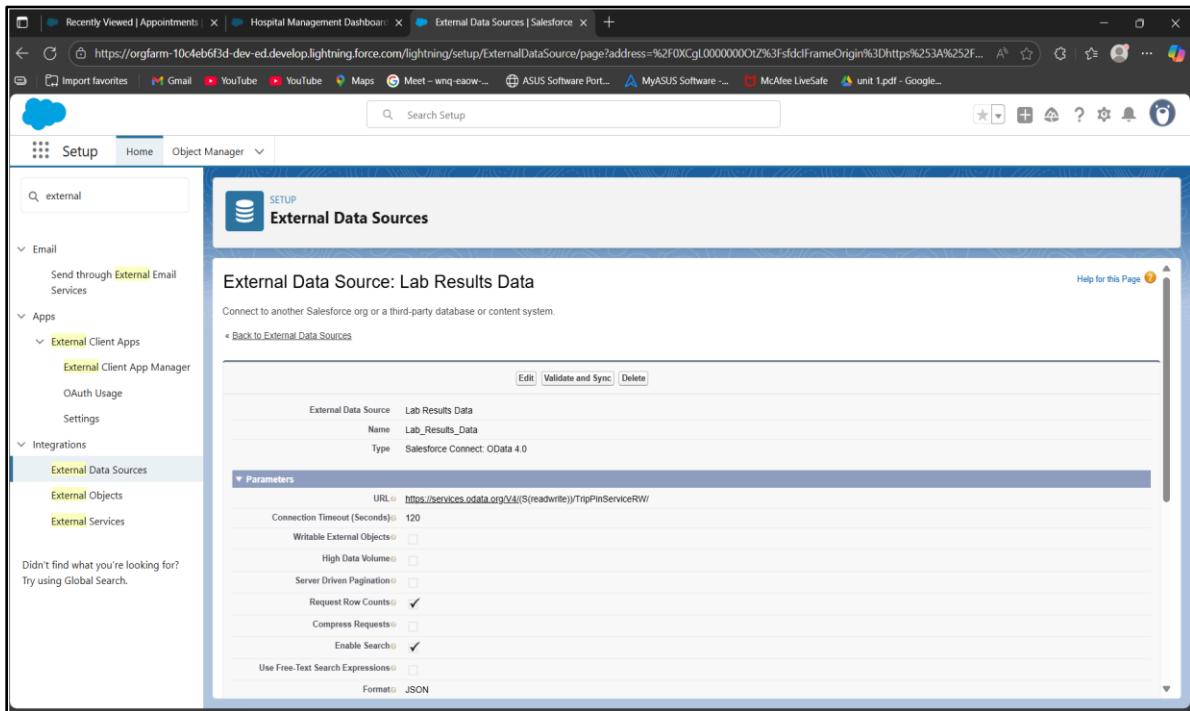
❖ **Junction Objects:**

- Go to Setup.
- Click on the Object Manager and click on create.
- Select the Custom Object.
- Fill the details:
 - Label: Patient Doctor Assignment
 - Object Name: Patient _Doctor _Assignment _c
 - Enable Allow Reports, Allow Activities and track reports
- Click on Save.
- Now in the Patient Doctor Assignment Object go to Fields & Relationships and click on New.
- Select the master – detail relationship and click Next
- Fill the details:
 - Field Label: Patient
 - Field Name: Patient _c
- Click on save.
- Again, go to Fields & Relationships and click on New
- Select the master – detail relationship and click Next
- Fill the details:
 - Field Label: Doctor
 - Field Name: Doctor _c
- Click on Save
- Click on the Assign Page Layout
- In the drop box, select the Patient Doctor Assignment Layout.
- Click save.
- Click on the App Launcher, select Reports
- Click on New Report
- Select the Patient Doctor Assignment
- Click start Report
- Drag the Patient and Doctor to the report columns
- Click on Save & Run
- Name the report: Patient-Doctor Assignments
- Create and choose the HealthCare report and then save
- Click on the App Launcher, select the Dashboards
- Click on New Dashboard
- Enter Name: Hospital Management Dashboard and then create
- Click on +widget and choose table
- Select the Patient-Doctor Assignments report
- Click on save and then Done.

❖ **External Objects:**

- Go to Setup.
- In the Quick Find box, type the External Data Sources click on it.
- Click on the New External Data Sources.
- Fill the details:
 - Label: Lab Results Data

- Name: Lab _ Results _ Data
- Type: OData 4.0
- URL: [https://services.odata.org/V4/\(S\(readwrite\)\)/TripPinServiceRW/](https://services.odata.org/V4/(S(readwrite))/TripPinServiceRW/)
- Identity Type: Named Principal
- Authentication Protocol: No Authentication
- Click on Save.
- Click on Validate and Sync
- Select people and click on sync.



Salesforce Project Phase-4

Health Patient Management and Telemedicine CRM

Phase 4: Process Automation (Admin)

❖ Validation Sets:

- Go to Setup.
- Click on the Object Manager.
- Find the required objects: Patients, Appointments, Telemedicine Reports, Bed Availability
- Click on Validation Sets and click on New.
- Patients:
 - Rule Name: Validate_Patient_Contact
 - Error Condition Formula: contact_c = '10' and click check syntax
 - Error Message: Please enter a valid contact number
 - Error Location: Field and then Contact_Number

The screenshot shows the Salesforce Setup interface for the Object Manager. The left sidebar lists various object settings like Details, Fields & Relationships, Page Layouts, etc. The main content area is titled 'Validation Rules' under the 'Patients' object. A single rule is listed:

Rule Name	Error Location	Error Message	Active	Modified By
Validate_Patient_Contact	Contact_Number	please enter a valid contact number	✓	Anusha Peetha, 9/22/2025, 5:04 AM

➤ Appointments:

1. Appointment _ Date:

- Rule Name: Validate _Appointment _Date
- Error Condition Formula: Appointment _Date __c < NOW () and click check syntax
- Error Message: Appointment date cannot be in the past
- Error Location: Field and then Appointment Date

2. Status:

- Rule Name: Validate _Appointment _Status
- Error Condition Formula: ISPICKVAL (Status __c, "Cancelled") and click check syntax
- Error Message: Status cannot be blank
- Error Location: Field and then Status

The screenshot shows the Salesforce Setup interface under the Object Manager section for the 'Appointments' object. On the left, there is a sidebar with various tabs like Details, Fields & Relationships, Page Layouts, etc. The 'Validation Rules' tab is selected. A table lists two validation rules:

Rule Name	Error Location	Error Message	Active	Modified By
Validate_Appointment_Date	Appointment_Date	Appointment date cannot be in the past	✓	Anusha Peetha, 9/22/2025, 5:10 AM
Validate_Appointment_Status	Status	Status cannot be blank	✓	Anusha Peetha, 9/22/2025, 5:12 AM

➤ Bed Availability:

- Rule Name: Validate _Bed _Status
- Error Condition Formula: ISPICKVAL (Bed _Status __c, "Occupied") and click check syntax
- Error Message: Cannot assign patient. Bed is already occupied.
- Error Location: Field and then bed _status

SETUP > OBJECT MANAGER
Bed Availability

Validation Rules
1 items, Sorted by Rule Name

Rule Name	Error Location	Error Message	Active	Modified By
Validate_Bed_Status	bed_status.c	Cannot assign patient. Bed is already occupied.	✓	Anusha Peetha, 9/22/2025, 5:17 AM

➤ Telemedicine Records:

- Rule Name: Validate_Telemedicine_Fields
- Error Condition Formula: ISBLANK(Patient__c) || ISBLANK(Doctor__c) and click check syntax
- Error Message: Patient and Doctor must be selected before saving a Telemedicine session.
- Error Location: Top of page

SETUP > OBJECT MANAGER
Telemedicine Records

Validation Rules
1 items, Sorted by Rule Name

Rule Name	Error Location	Error Message	Active	Modified By
Validate_Telemedicine_Fields	Top of Page	Patient and Doctor must be selected before saving a Telemedicine session.	✓	Anusha Peetha, 9/22/2025, 5:23 AM

➤ Payments:

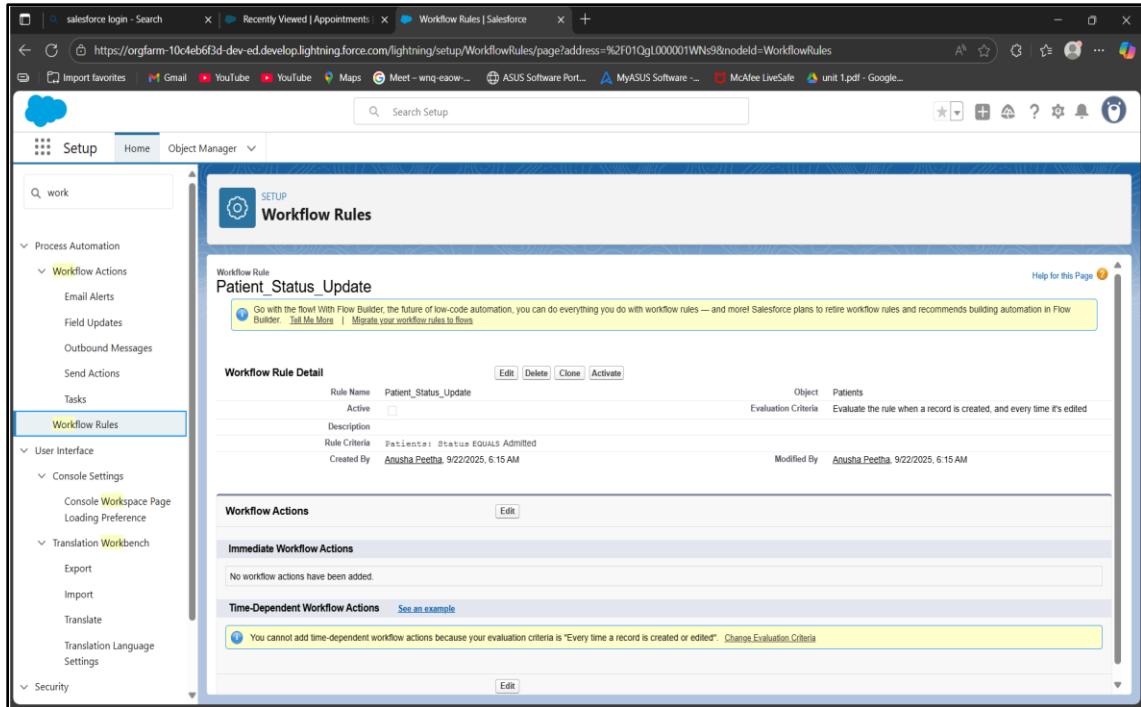
- Rule Name: Validate _Payment _Amount
- Error Condition Formula: Amount _c <= 0 and click check syntax
- Error Message: Payment amount must be greater than zero
- Error Location: Top of page

The screenshot shows the Salesforce Object Manager interface for the 'Payments' object. On the left, a sidebar lists various setup options like Details, Fields & Relationships, Page Layouts, etc. The main area displays a table titled 'Validation Rules' with one item listed:

RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY
Validate_Payment_Amount	Amount	Payment amount must be greater than zero	✓	Anusha Peetha, 9/22/2025, 5:58 AM

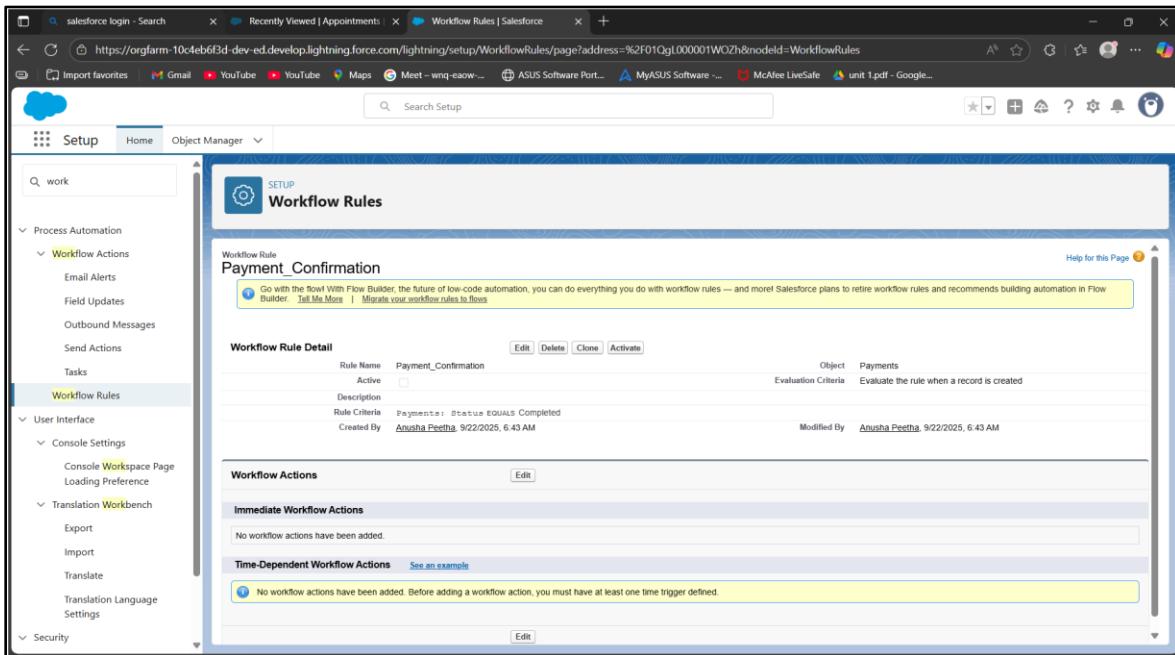
❖ Workflow Rules:

- Go to Setup.
- In the Quick Find box, type the Workflow Rules and click on it.
- Click continue.
- Now click on the New Rule and click on Continue on the Workflow Rule.
- Select the objects: Patients, Payments, Telemedicine Records,
- Click Next.
- Patients:
 - Rule Name: Patient_Status_Update
 - Evaluate Rule: created, and every time it's edited
 - Rule Criteria:
Field: Status _c, Operator: =, Value: 'Admitted'
 - Click Save & Next and Done



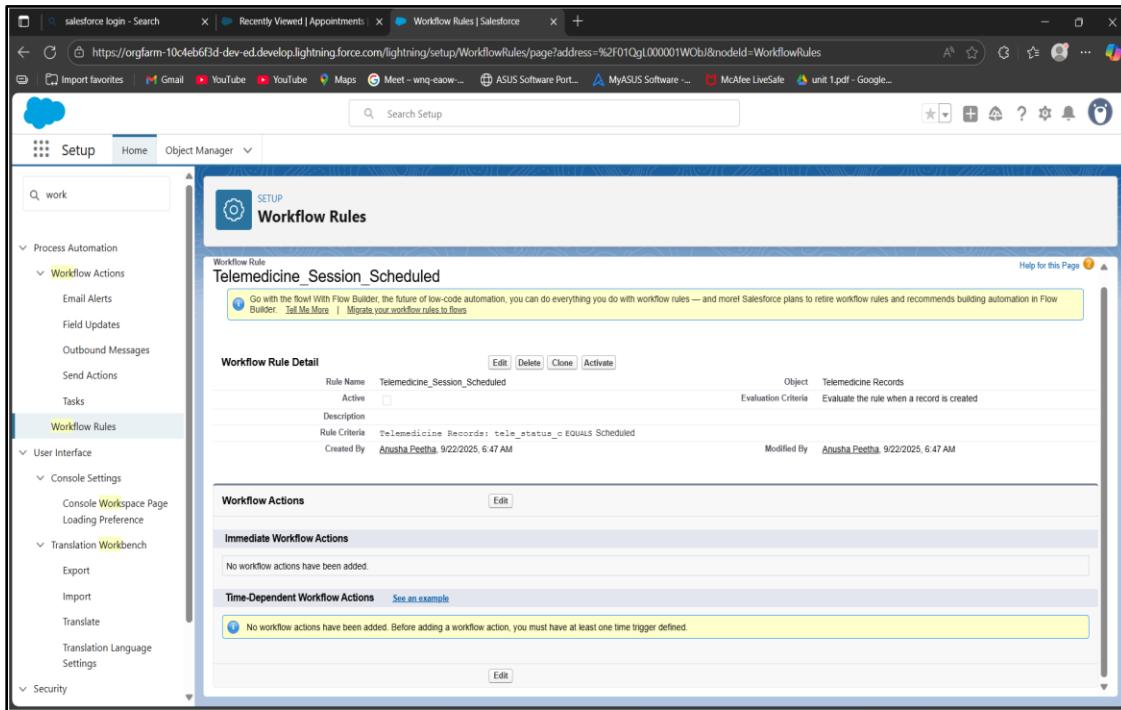
➤ Payments:

- Rule Name: Payment_Confirmation
- Evaluate Rule: Created
- Rule Criteria:
Fields: status_c, Operator: =, Value: Completed
- Click Save & Next and then Done



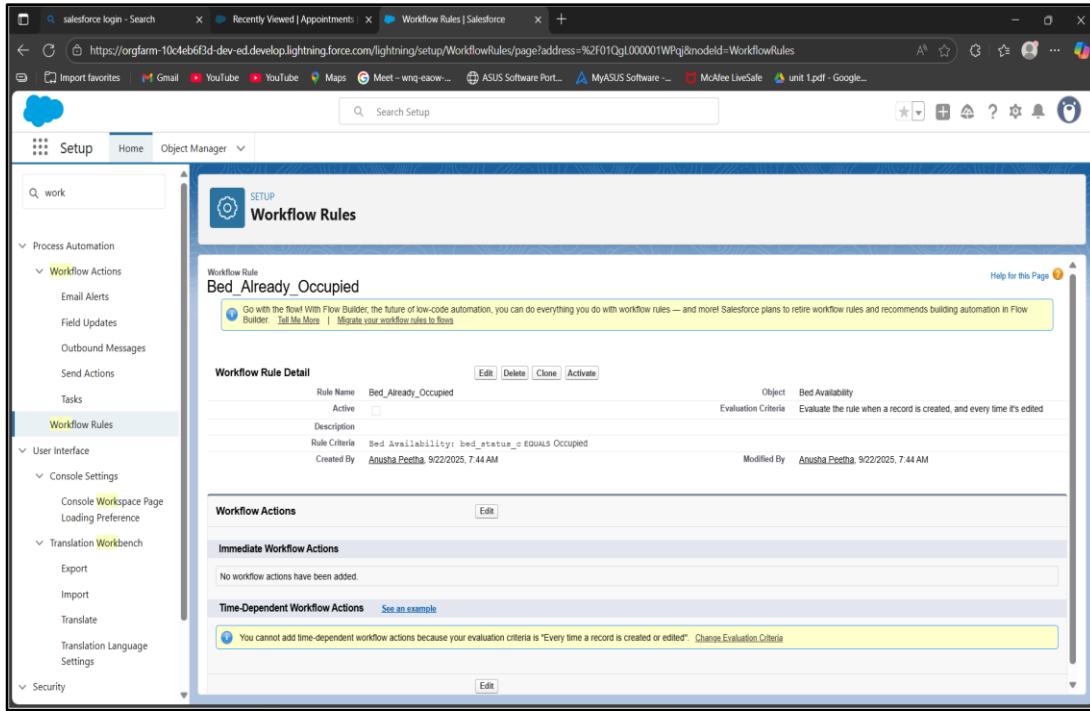
➤ Telemedicine Records:

- Rule Name: Telemedicine_session_scheduled
- Evaluate Rule: created
- Rule Criteria:
Field: status_c, Operator: =, Value: Scheduled
- Click on Save & Next and then click Done



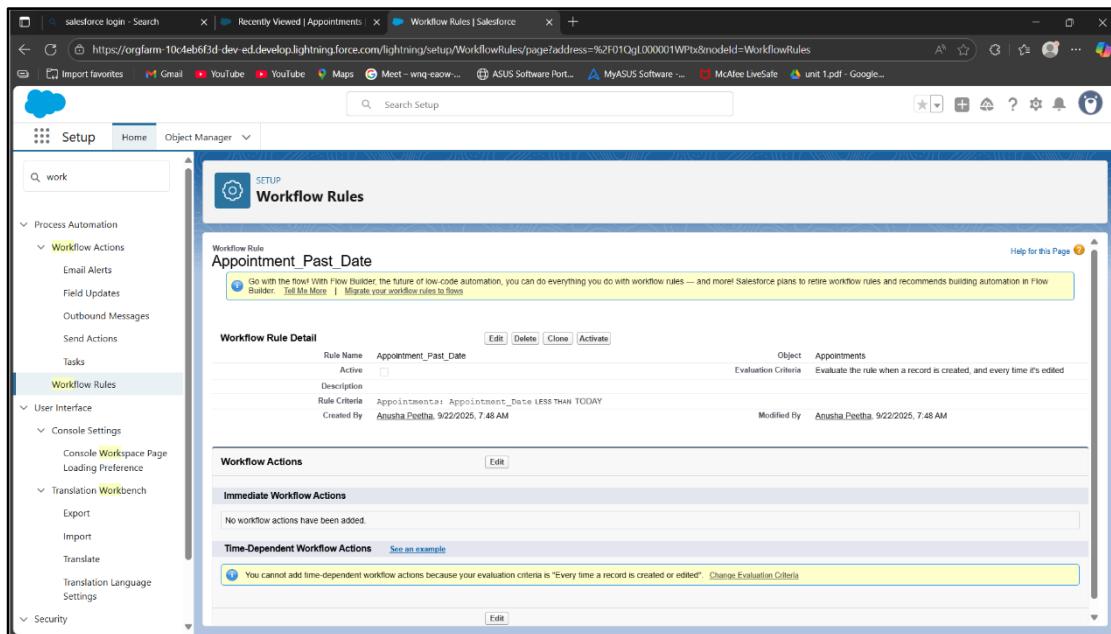
➤ Bed Availability:

- Rule Name: Bed_Already_Occupied
- Evaluation Criteria: created, and every time it's edited
- Rule Criteria:
Field: bed_status_c, Operator: =, Value: Occupied
- Click on Save & Next and then click Done



➤ Appointments:

- Rule Name: Is_Past_Appointment
- Evaluation Criteria: created, and every time it's edited
- Rule Criteria:
Field: Appointment_Date, Operator: <, Value: TODAY
- Click on Save & Next and then click Done

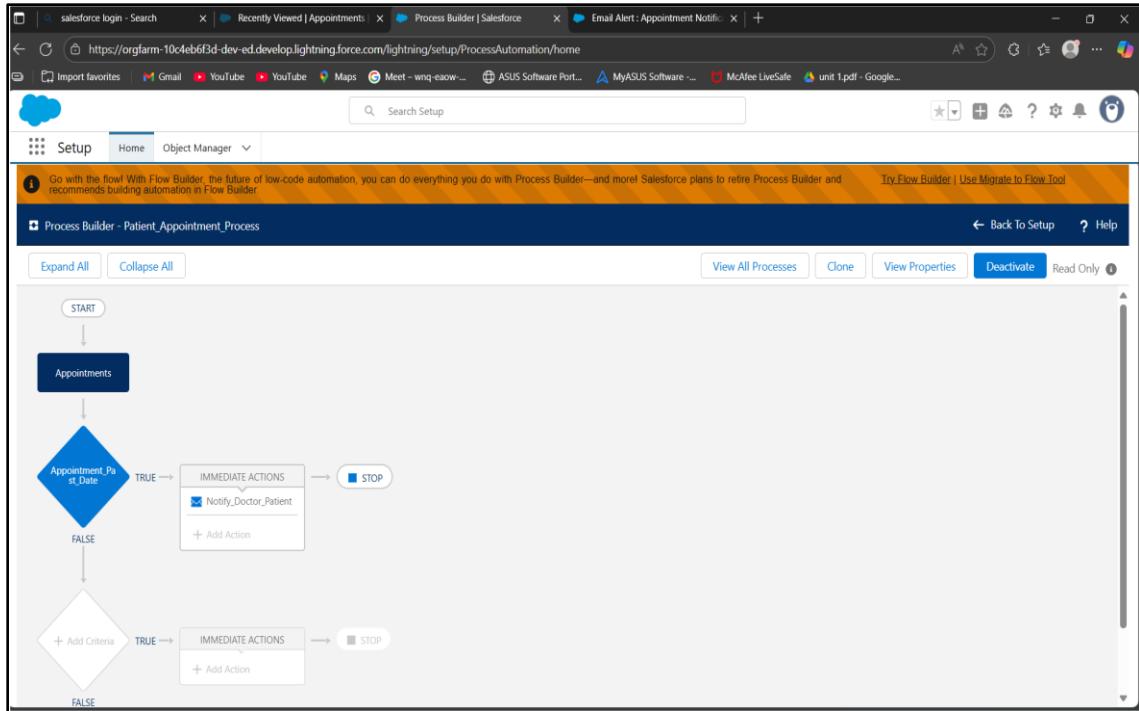


- After creating Workflow Rules for Objects click on the Activate.

Action	Rule Name	Description	Object	Active
Edit Del Deactivate	Appointment_Past_Date		Appointments	✓
Edit Del Deactivate	Bed_Already_Occupied		Bed Availability	✓
Edit Del Deactivate	Patient_Status_Update		Patients	✓
Edit Del Deactivate	Payment_Confirmation		Payments	✓
Edit Del Deactivate	Telemedicine_Session_Scheduled		Telemedicine Records	✓

❖ Process Builders:

- Go to Setup.
- In the Quick Find box, type the Process Builder and click on it.
- Click on New.
- Fill the details:
 - Process Name: Patient_Appointment_Process
 - API Name: Patient_Appointment_Process
 - The Process starts when: A record changes and then click save.
 - Click on +Add Object
 - Object: Appointments
 - Start the Process: when a record is created or edited and then click save.
 - Click +Add Criteria
 - Name: Appointment_Past_Date
 - Criteria: the field reference
 - Click +Add Action and Action Type: Email Alert
 - Action Name: Notify_Doctor_Patient
 - Appointment Notification and click on save
 - Click Activate and then Confirm



❖ Approval Process:

- Go to Setup.
- In the Quick Find box, type the Approval Settings and click on it.
- Select Appointments.
- Select the Standard Wizard.

Process Definition Detail	
Process Name	Appointment Approval Process
Unique Name	Appointment_Approval_Process
Description	
Entry Criteria	Appointments: Status EQUALS Scheduled
Record Editability	Administrator ONLY
Approval Assignment Email Template	Appointment for Unauthenticated User using Appointment Types - For Amazon Chime.
Initial Submitters	Appointments Owner, Role: HealthCare Admin
Created By	Anusha Peetha, 9/22/2025, 9:08 AM
Modified By	Anusha Peetha, 9/22/2025, 9:14 AM

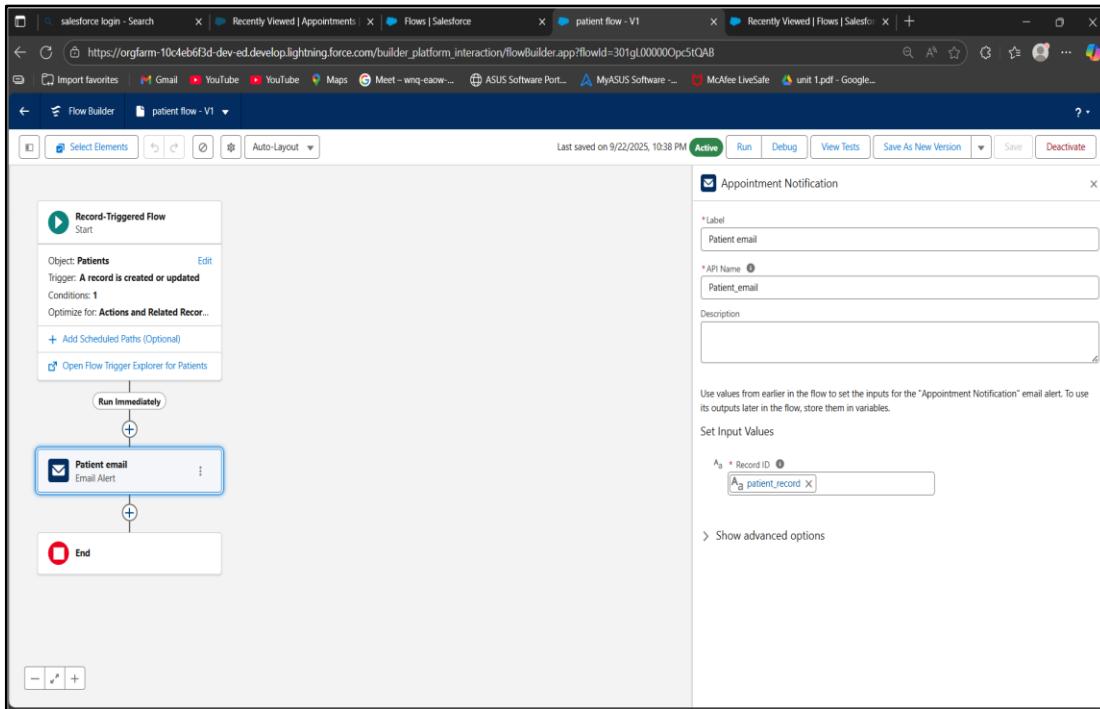
Initial Submission Actions	
Action Type	Add Existing Add New
Record Lock	Lock the record from being edited

Approval Steps						
Action	Step Number	Name	Description	Criteria	Assigned Approver	Reject Behavior
Show Actions Edit	1	Appointment Approval Process		Appointments: Status EQUALS Completed , else Approve	Manually Chosen	Final Rejection

- Process Name: Appointment Approval Process
- Field: Appointment status, Operator: =, Value: Scheduled
- Next and then Appointment Notification
- Select Appointment ID, Appointment Status, Patients, Record Type, Appoint Date, User
- Next, Save and then Click on Activate.

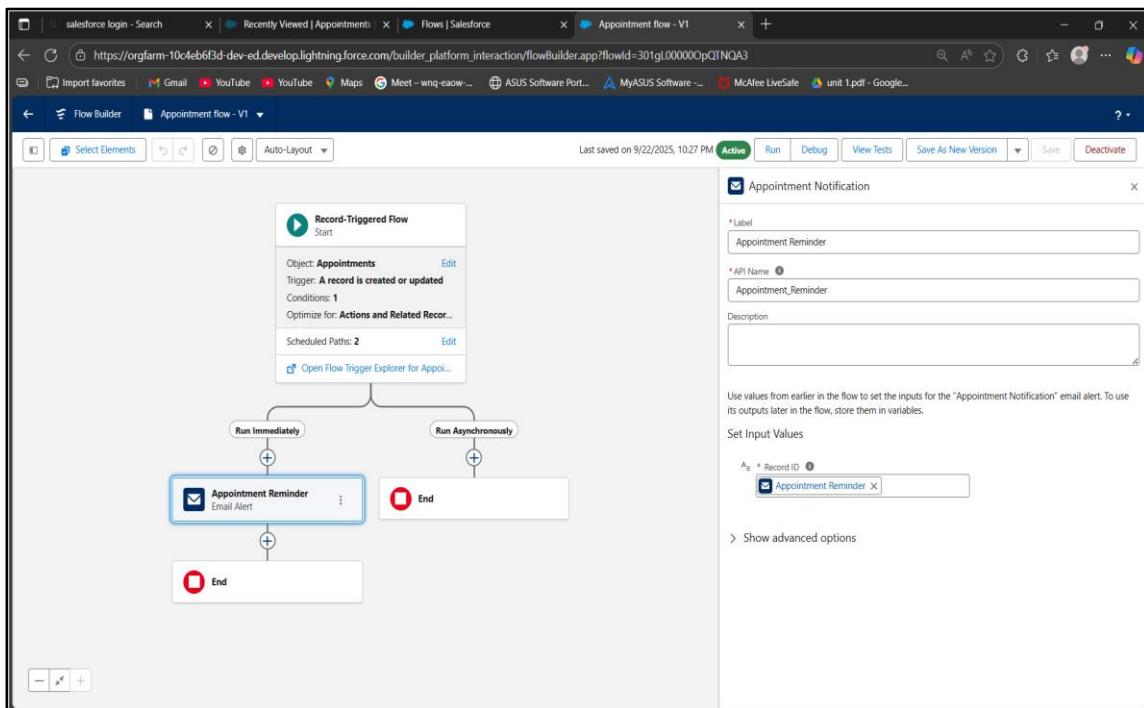
❖ **Flow Builder (Screen, Record-Triggered, Scheduled, Auto-launched):**

- Go to the Setup.
- In the Quick Find box, type the Flow Builder then click on it.
- Click on New Flow.
- Patients:
 - Click on the Record – Triggered Flow
 - Object: Patient _c
 - Trigger: A record is created or updated
 - Select the Condition is met
Field: status, Operator: =, Value: Confirmed
 - Add element and click on the send email
 - Select the existing one.
 - Select the record on existing one.
 - Name: Patient flow and then save.
 - Click save and then activate.



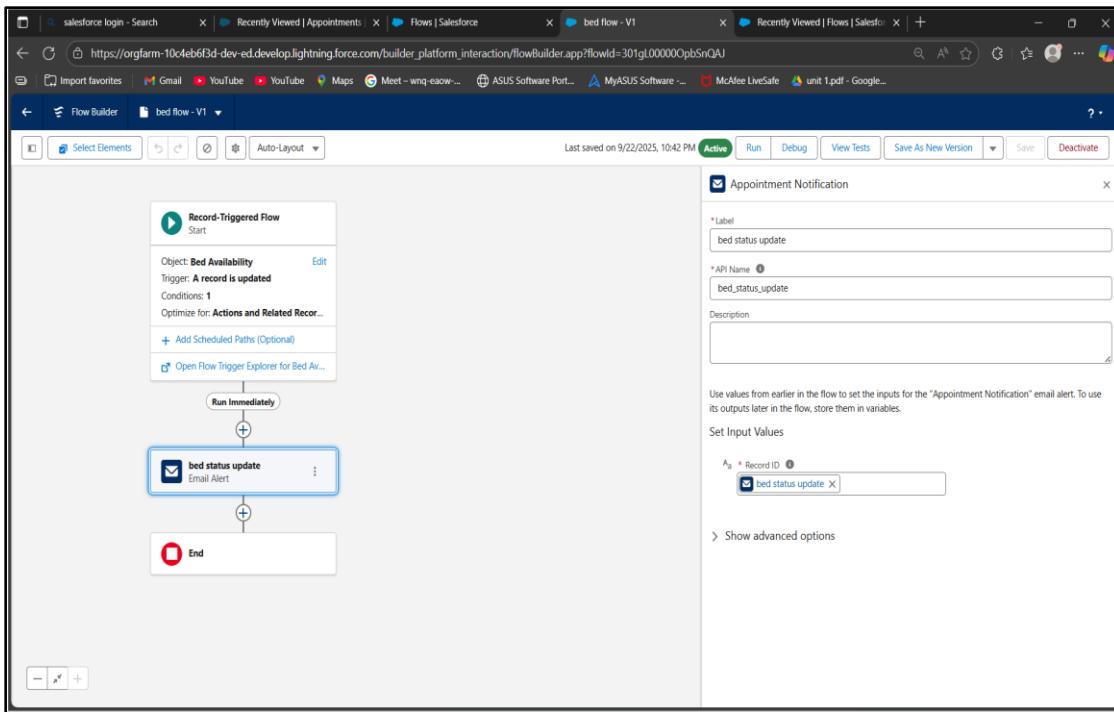
➤ Appointments:

- Click on the Record- Triggered Flow
- Object: Appointment _c
- Trigger: A record is created or updated
- Select the Condition is met
Field: Appointment _status, Operator: =, Value: Scheduled
- Add element and click on the send email
- Select the existing one.
- Select the record on existing one.
- Name: Appointment flow and then save.
- Click save and then activate



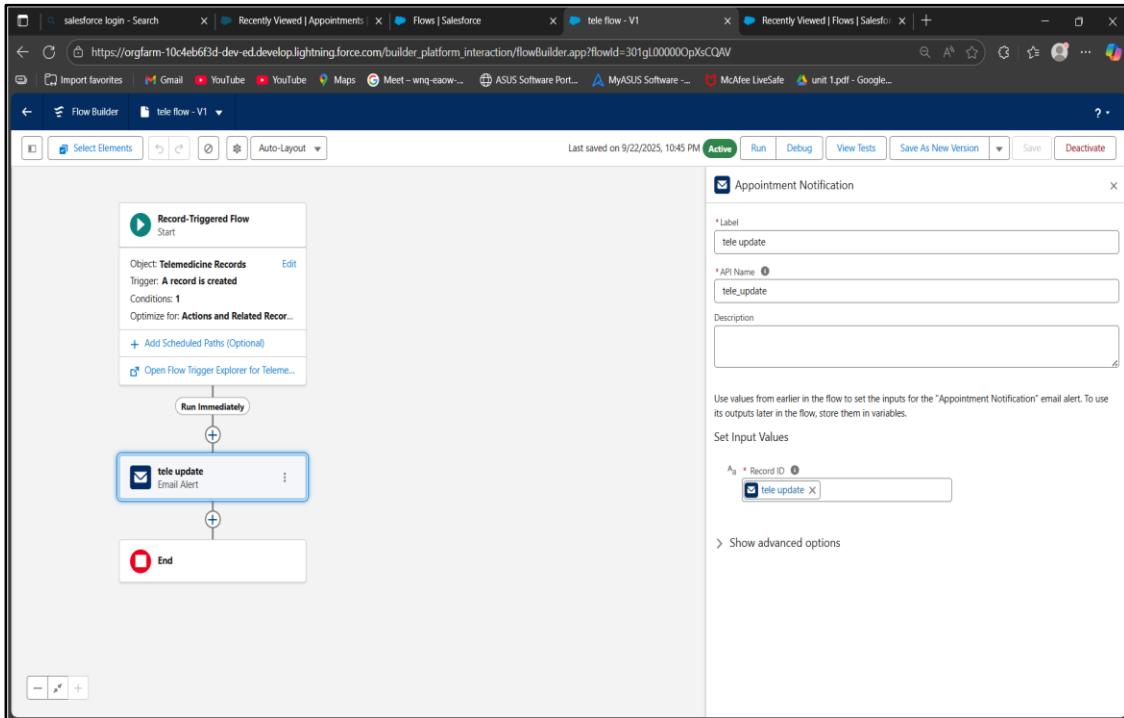
➤ Bed Availability:

- Click on the Record – Triggered Flow
- Object: Bed _Availability _c
- Trigger: A record is updated
- Select the Condition is met
Field: status, Operator: =, Value: Available
- Add element and click on the send email
- Select the existing one.
- Select the record on existing one.
- Name: Bed flow and then save.
- Click save and then activate



➤ Telemedicine Records:

- Click on the Record – Triggered Flow
- Object: Telemedicine _c
- Trigger: A record is created
- Select the Condition is met
Field: tele _status, Operator: =, Value: Scheduled
- Add element and click on the send email
- Select the existing one.
- Select the record on existing one.
- Name: tele flow and then save.
- Click save and then activate.



❖ Email Alerts:

- Go to Setup.
- In the Quick Find box, type the Email Alerts and click on it.
- If you don't have the Email template first you need to create the Email template by typing the Classic email template in Quick Find box.
- Create New Template.
- Appointments:
 - Description: Appointment Confirmation Alert – Patient
 - Object: Appointment_c
 - Email Template: Patient_Appointment_Confirmation
 - Recipients: HealthCare Admin, Patient Coordinator, Doctor
- Bed Allocation:
 - Description: Bed Allocation Alert – Patient
 - Object: Bed Availability_c
 - Email Template: Patient_Bed_Allocation_Alert
 - Recipients: HealthCare Admin, Patient Coordinator, Doctor
- Telemedicine Reports:
 - Description: Telemedicine Appointment Alert – Patient
 - Object: Telemedicine_c
 - Email Template: Patient_Telemedicine_Confirmation
 - Recipients: HealthCare Admin, Patient Coordinator, Doctor
- Payments:
 - Description: Payment Status Alert -Patient
 - Object: Payment_c

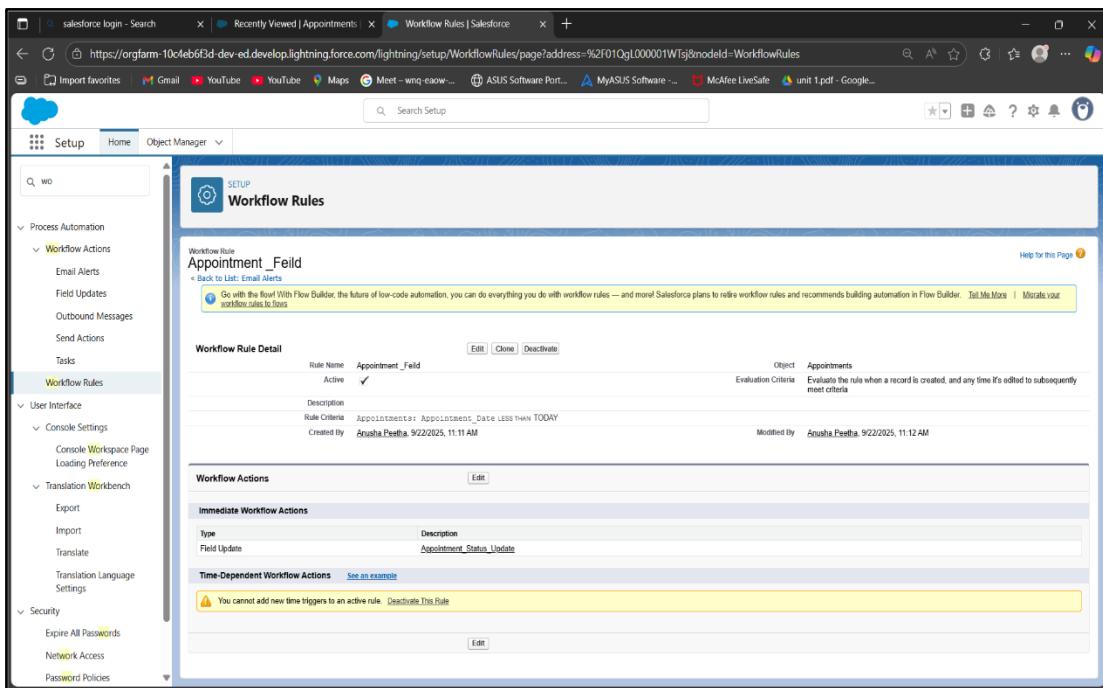
- Email Template: Patient _ Payment _ Status _ Alert
- Recipients: HealthCare Admin, Patient Coordinator, Doctor

Action	Description	Email Template Name	Object	Last Modified Date
Edit Del	Appointment Confirmation Alert - Patient	Patient_Appointment_Confirmation	Appointments	9/22/2025
Edit Del	Appointment Notification	Appointment for Unauthenticated User using Appointment Types - For Amazon China.	Appointments	9/22/2025
Edit Del	Bed Allocation Alert - Patient	Bed_Allocation_Alert	Bed Availability	9/22/2025
Edit Del	Payment Status Alert - Patient	Patient_Payment_Status_Alert	Payments	9/22/2025
Edit Del	Telemedicine Appointment Alert - Patient	Patient_Telemedicine_Confirmation	Telemedicine Records	9/22/2025

❖ Fields Updates:

- Go to Setup.
- In the Quick Find box, type the Workflow Layout and click it.
- Click on New Rule.
- Select Object: Patient _c
 - Rule Criteria:
Fields: Patient, Operator: =, Value: Inpatient

- Click on Save
 - Click on Add immediate Action and then select New Field Update
 - Name: Patient_Status_Update
 - Field to update: Status_c
 - New value: Admitted
 - Click on Save and then Activate
- Select Object: Appointment_c
- Rule Criteria:
Fields: Appointment_Date, Operator: less than, Value: TODAY
 - Click on Save
 - Click on Add immediate Action and then select New Field Update
 - Name: Appointment_Status_Update
 - Field to update: Status_c
 - New value: Completed
 - Click on Save and then Activate



- Select Object: Bed_Availability_c
- Rule Criteria:
Fields: Bed_status, Operator: =, Value: Available
 - Click on Save
 - Click on Add immediate Action and then select New Field Update
 - Name: Bed_Status_Update
 - Field to update: Status_c
 - New value: Available
 - Click on Save and then Activate

The screenshot shows the Salesforce Setup interface with the 'Workflow Rules' page open. The left sidebar is collapsed, and the main area displays a 'Workflow Rule' named 'Bed Field'. The rule is active and evaluates the 'Bed Availability' field when a record is created or edited. It has one immediate workflow action: a 'Field Update' for the 'Bed_Status__c' field. A note at the bottom states: 'You cannot add new time triggers to an active rule. Deactivate This Rule'.

➤ Select Object: Telemedicine Reports

- Rule Criteria:
Fields: Telemedicine_Status, Operator: =, Value: Completed
- Click on Save
- Click on Add immediate Action and then select New Field Update
- Name: Tele_Status_Update
- Field to update: Status_c
- New value: Completed

The screenshot shows the Salesforce Setup interface with the 'Workflow Rules' page open. The left sidebar is collapsed, and the main area displays a 'Workflow Rule' named 'Tele field'. The rule is active and evaluates the 'Telemedicine Records: tele_Status__c equals Scheduled' field. It has one immediate workflow action: a 'Field Update' for the 'Tele_Status__c' field. A note at the bottom states: 'You cannot add new time triggers to an active rule. Deactivate This Rule'.

- Click on Save and then Activate

❖ **Tasks:**

Task setup was finished in the previous step. No additional changes are needed for tasks.

❖ **Custom Notifications:**

- Go to Setup.
- In the Quick Find box, type the Custom Notifications and click on it.
- Click on New.
- Appointment:
 - Name: Appointment _Notification
 - Supported Channels: Desktop, Mobile
- Bed Availability:
 - Name: Bed _Allocation _Notification
 - Supported Channels: Desktop, Mobile
- Telemedicine Reports:
 - Name: Telemedicine _Notification
 - Supported Channels: Desktop, Mobile
- Payments:
 - Name: Payment _Notification
 - Supported Channels: Desktop, Mobile
- Click on Save.

NOTIFICATION NAME	API NAME	NAMESPACE	DESKTOP	MOBILE
Appointment_Notification	Appointment_Notification		✓	✓
Bed_Allocation_Notification	Bed_Allocation_Notification		✓	✓
enablement_coaching_feedback_ready	enablement_coaching_feedback_ready		✓	✓
Payment_Notification	Payment_Notification		✓	✓
Telemedicine_Notification	Telemedicine_Notification		✓	✓

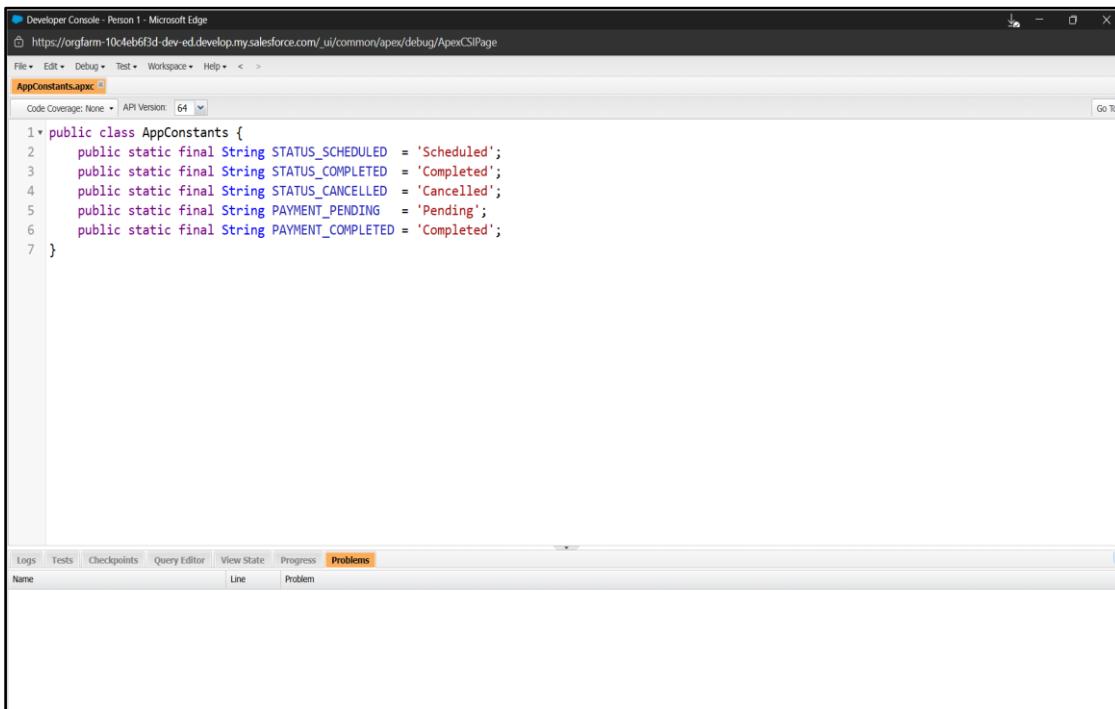
Salesforce Project Phase-5

Health Patient Management and Telemedicine CRM

Phase 5: Apex Programming (Developer)

❖ Classes & Objects:

- Go to the Developer Console.
- Click on File and select the New and then click on Apex class
- Click ok.
- AppConstants.apxc

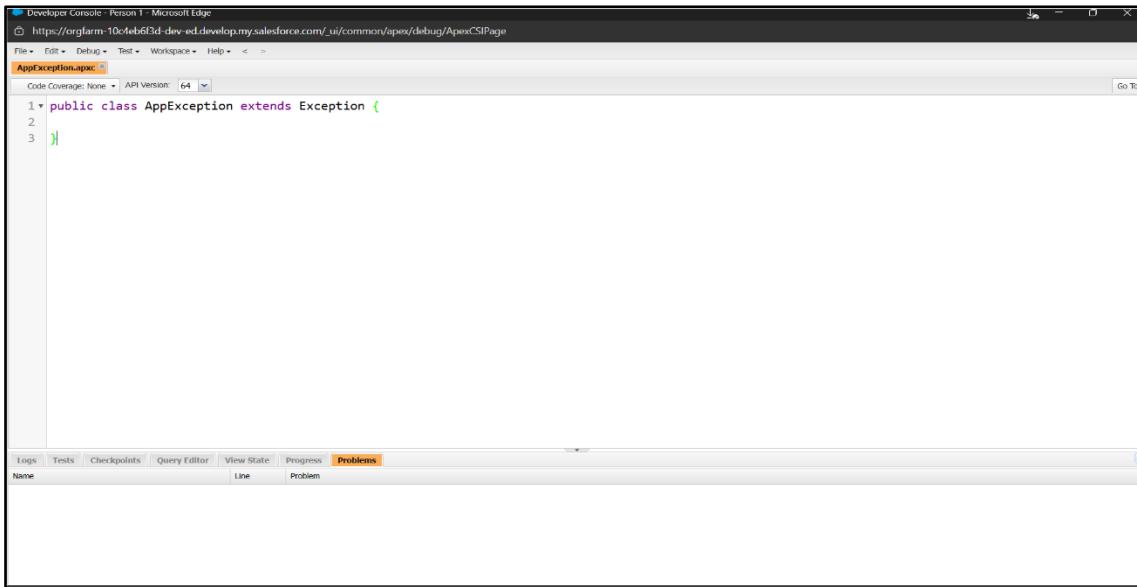


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb63d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCIPage". The menu bar includes File, Edit, Debug, Test, Workspace, Help, and a back/forward button. The main area is titled "AppConstants.apxc" and contains the following Apex code:

```
1 public class AppConstants {
2     public static final String STATUS_SCHEDULED = 'Scheduled';
3     public static final String STATUS_COMPLETED = 'Completed';
4     public static final String STATUS_CANCELLED = 'Cancelled';
5     public static final String PAYMENT_PENDING = 'Pending';
6     public static final String PAYMENT_COMPLETED = 'Completed';
7 }
```

Below the code editor, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, ViewState, Progress, and Problems. The Problems tab is currently selected. A status bar at the bottom shows "Name", "Line", and "Problem".

➤ AppException.apxc

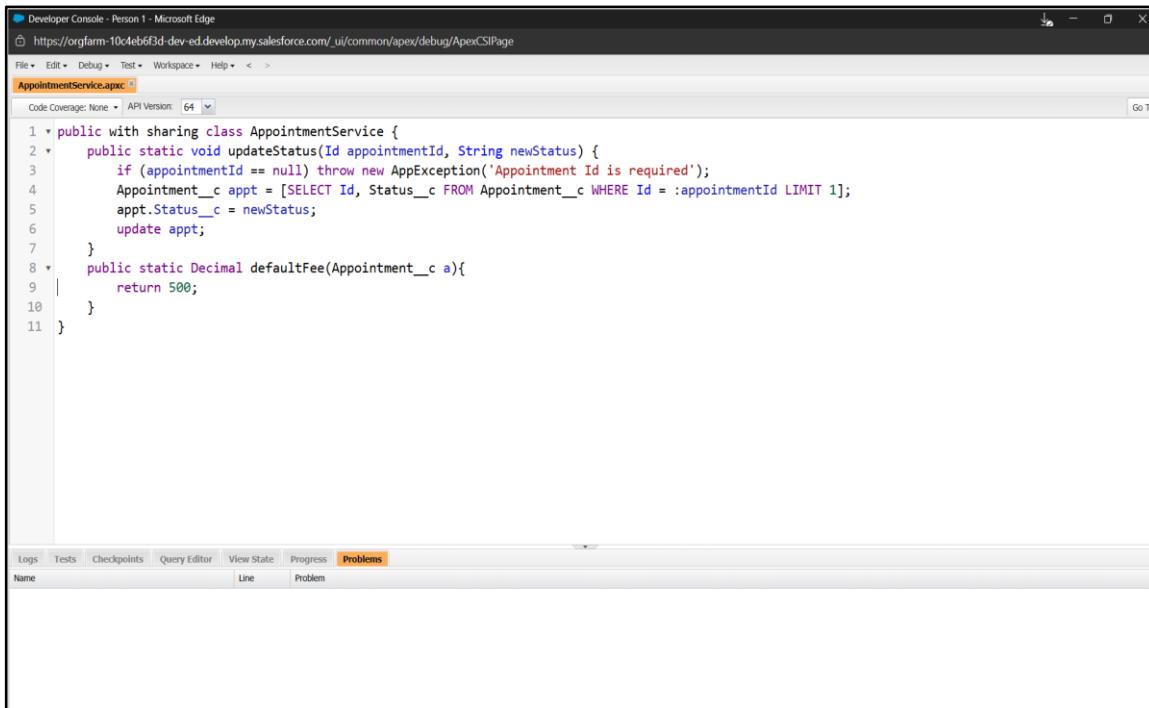


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar at the top has "AppException.apxc" selected. Below the tabs, there are buttons for "Code Coverage: None" and "API Version: 64". The main editor area contains the following Apex code:

```
1 public class AppException extends Exception {  
2  
3 }
```

Below the editor, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is highlighted. A table below the tabs lists problems, with columns for Name, Line, and Problem. The table is currently empty.

➤ AppointmentService.apxc

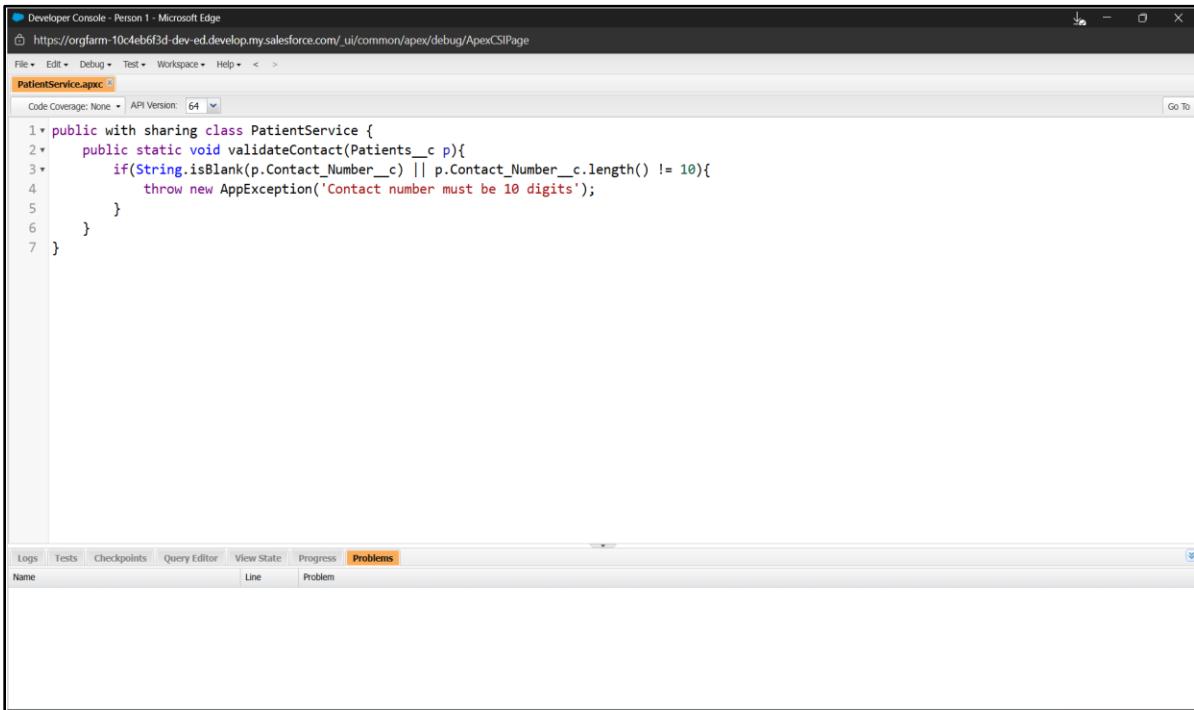


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar at the top has "AppointmentService.apxc" selected. Below the tabs, there are buttons for "Code Coverage: None" and "API Version: 64". The main editor area contains the following Apex code:

```
1 public with sharing class AppointmentService {  
2     public static void updateStatus(Id appointmentId, String newStatus) {  
3         if (appointmentId == null) throw new AppException('Appointment Id is required');  
4         Appointment__c appt = [SELECT Id, Status__c FROM Appointment__c WHERE Id = :appointmentId LIMIT 1];  
5         appt.Status__c = newStatus;  
6         update appt;  
7     }  
8     public static Decimal defaultFee(Appointment__c a){  
9         return 500;  
10    }  
11 }
```

Below the editor, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is highlighted. A table below the tabs lists problems, with columns for Name, Line, and Problem. The table is currently empty.

➤ PatientService.apxc

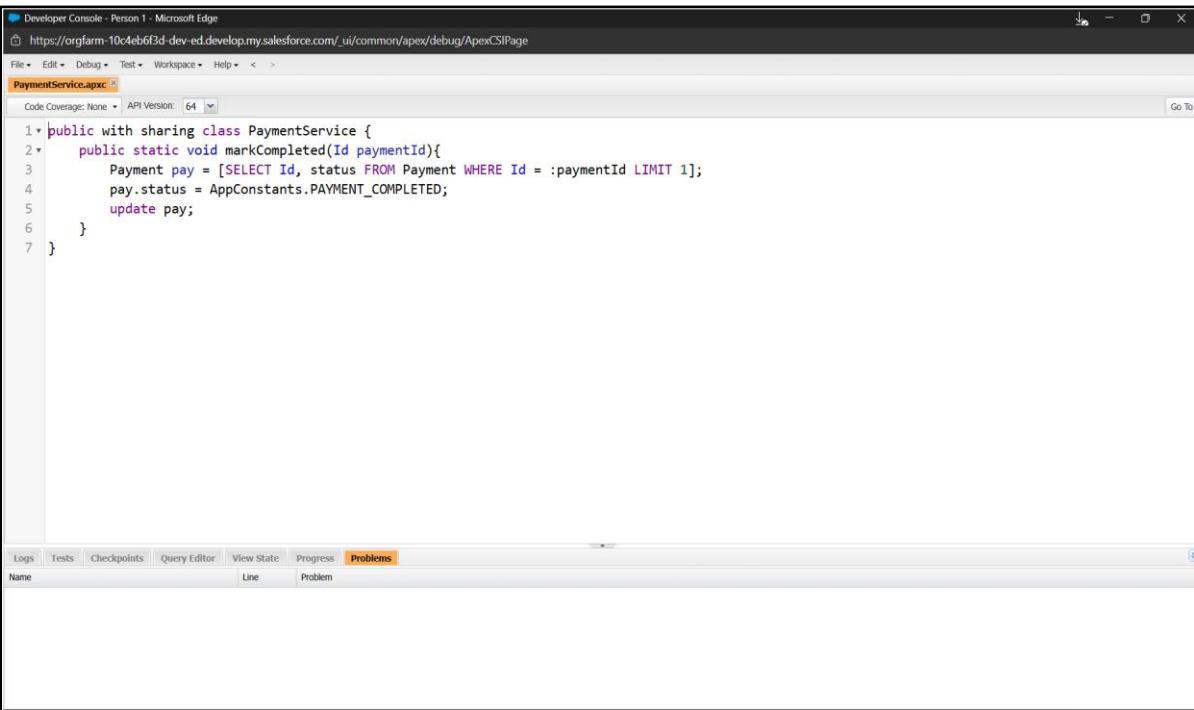


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage". The tab bar at the top has "PatientService.apxc" selected. Below the tabs, there are buttons for "Code Coverage: None" and "API Version: 64". The main area contains the following Apex code:

```
1 * public with sharing class PatientService {
2 *     public static void validateContact(Patients__c p){
3 *         if(String.isBlank(p.Contact_Number__c) || p.Contact_Number__c.length() != 10){
4 *             throw new AppException('Contact number must be 10 digits');
5 *         }
6 *     }
7 }
```

At the bottom of the developer console, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected. Below the tabs, there are three columns: Name, Line, and Problem.

➤ PaymentService.apxc



The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage". The tab bar at the top has "PaymentService.apxc" selected. Below the tabs, there are buttons for "Code Coverage: None" and "API Version: 64". The main area contains the following Apex code:

```
1 * public with sharing class PaymentService {
2 *     public static void markCompleted(Id paymentId){
3 *         Payment pay = [SELECT Id, status FROM Payment WHERE Id = :paymentId LIMIT 1];
4 *         pay.status = AppConstants.PAYMENT_COMPLETED;
5 *         update pay;
6 *     }
7 }
```

At the bottom of the developer console, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected. Below the tabs, there are three columns: Name, Line, and Problem.

➤ BedAvailabilityService.apxc

The screenshot shows the Salesforce Developer Console in Microsoft Edge. The URL is https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The tab title is "BedAvailabilityService.apxc". The code editor contains the following Apex class:

```
1 public class BedAvailabilityService {  
2  
3 }
```

The console interface includes a navigation bar with File, Edit, Debug, Test, Workspace, Help, and a Go To button. Below the code editor is a tabs bar with Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected, showing an empty list.

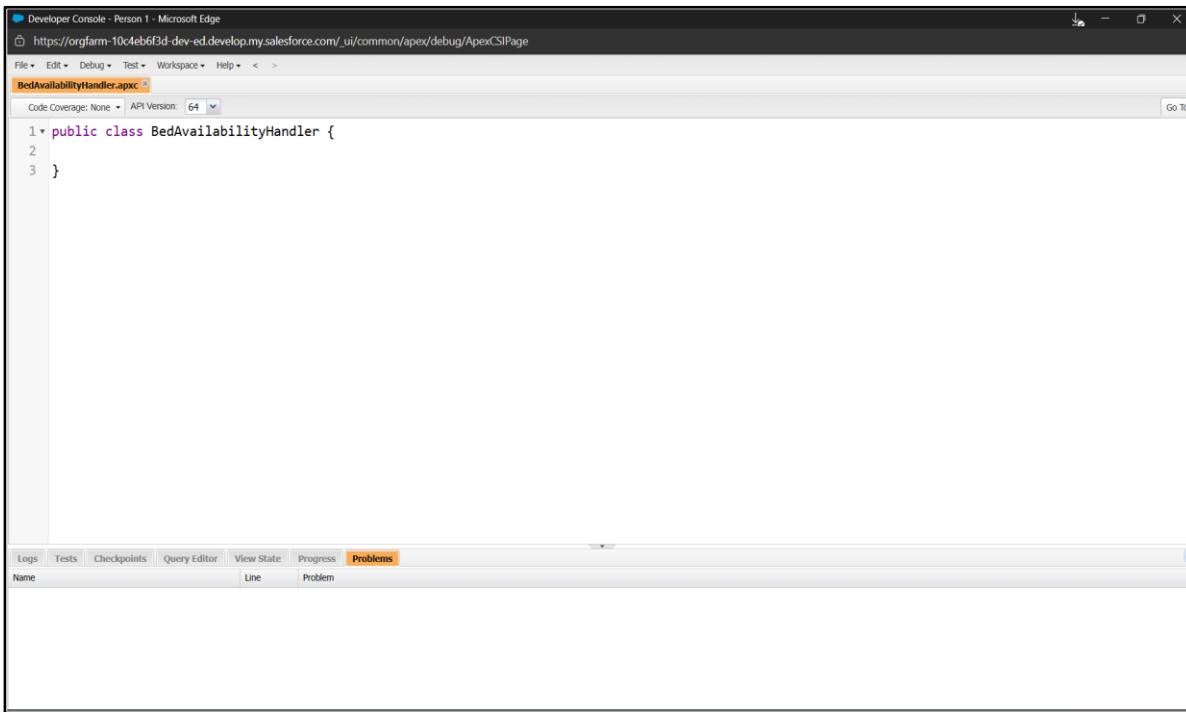
➤ AppointmentHandler.apxc

The screenshot shows the Salesforce Developer Console in Microsoft Edge. The URL is https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The tab title is "AppointmentHandler.apxc". The code editor contains the following Apex class:

```
1 public with sharing class AppointmentHandler {  
2     public static void beforeInsert(List<Appointment__c> newList) {  
3         for (Appointment__c a : newList) {  
4             if (a.Appointment_Date__c != null && a.Appointment_Date__c < System.now()) {  
5                 a.addError('Appointment date cannot be in the past.'); } } }  
6     public static void afterUpdate(List<Appointment__c> newList,Map<Id, Appointment__c> oldMap) {  
7         List<SObject> paymentsToInsert = new List<SObject>();  
8         Map<String, Schema.SObjectType> gd = Schema.getGlobalDescribe();  
9         String paymentApiName = null;  
10        if (gd.containsKey('Payment__c')) paymentApiName = 'Payment__c';  
11        else if (gd.containsKey('Payment')) paymentApiName = 'Payment';  
12        else if (gd.containsKey('Payments__c')) paymentApiName = 'Payments__c';  
13        else if (gd.containsKey('Payments')) paymentApiName = 'Payments';  
14        if (paymentApiName == null) {  
15            System.debug('AppointmentHandler: Could not find a Payment object in this org. Please verify API name.');//  
16            return; }  
17        Schema.SObjectType paymentType = gd.get(paymentApiName);  
18        Map<String, Schema.SObjectField> paymentFields = paymentType.getDescribe().fields.getMap();  
19        for (Appointment__c a : newList) {  
20            Appointment__c oldRec = oldMap.get(a.Id);  
21            Boolean nowCompleted = (a.Status__c == AppConstants.STATUS_COMPLETED);  
22            Boolean wasCompleted = (oldRec != null && oldRec.Status__c == AppConstants.STATUS_COMPLETED);
```

The console interface includes a navigation bar with File, Edit, Debug, Test, Workspace, Help, and a Go To button. Below the code editor is a tabs bar with Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected, showing an empty list.

➤ BedAvailabilityHandler.apxc

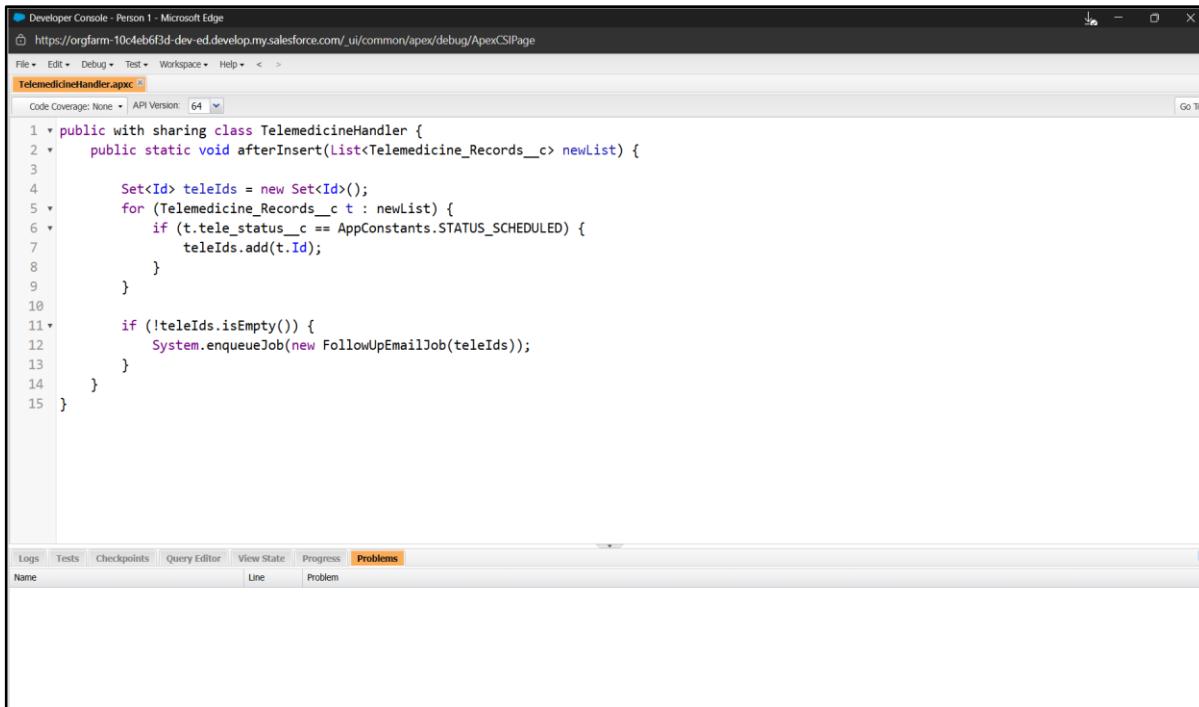


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar at the top has "BedAvailabilityHandler.apxc" selected. The main area displays the following Apex code:

```
1 public class BedAvailabilityHandler {  
2 }  
3 }
```

Below the code editor, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected. The status bar at the bottom shows "Name" and "Line Problem".

➤ TelemedicineHandler.apxc



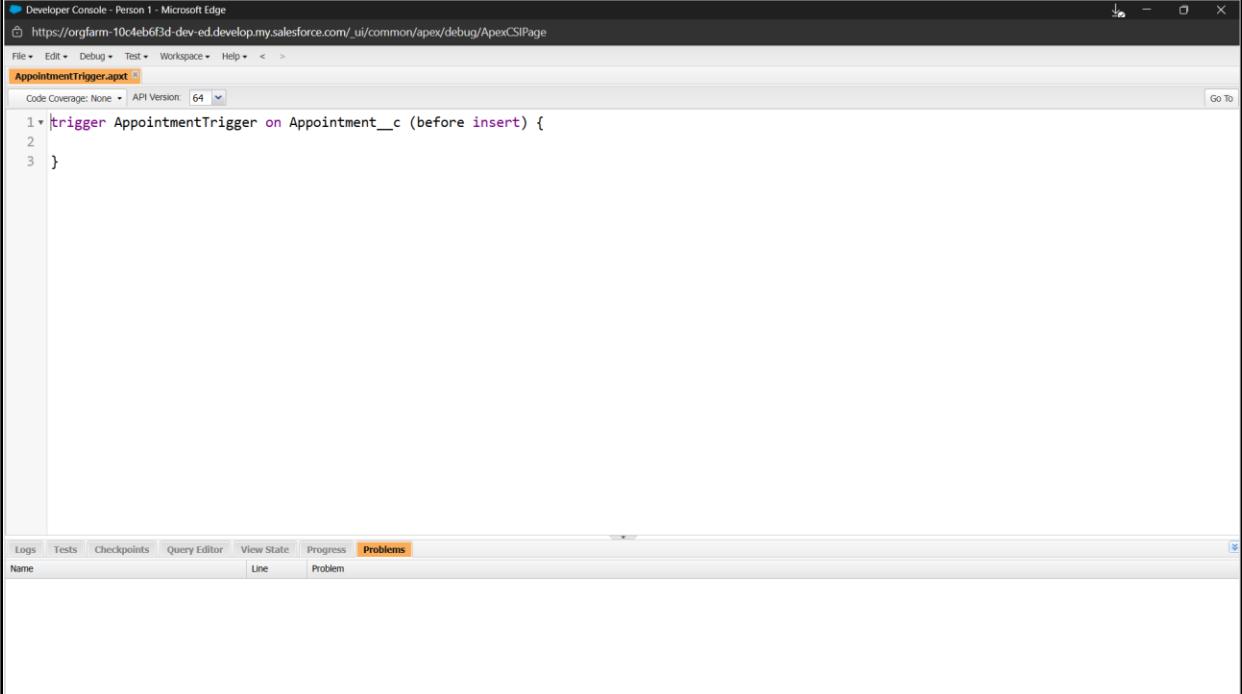
The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar at the top has "TelemedicineHandler.apxc" selected. The main area displays the following Apex code:

```
1 public with sharing class TelemedicineHandler {  
2     public static void afterInsert(List<Telemedicine_Records__c> newList) {  
3         Set<Id> teleIds = new Set<Id>();  
4         for (Telemedicine_Records__c t : newList) {  
5             if (t.tele_status__c == AppConstants.STATUS_SCHEDULED) {  
6                 teleIds.add(t.Id);  
7             }  
8         }  
9         if (!teleIds.isEmpty()) {  
10             System.enqueueJob(new FollowUpEmailJob(teleIds));  
11         }  
12     }  
13 }  
14 }  
15 }
```

Below the code editor, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected. The status bar at the bottom shows "Name" and "Line Problem".

❖ Apex Triggers (before/after insert/update/delete):

➤ AppointmentTrigger.apxt

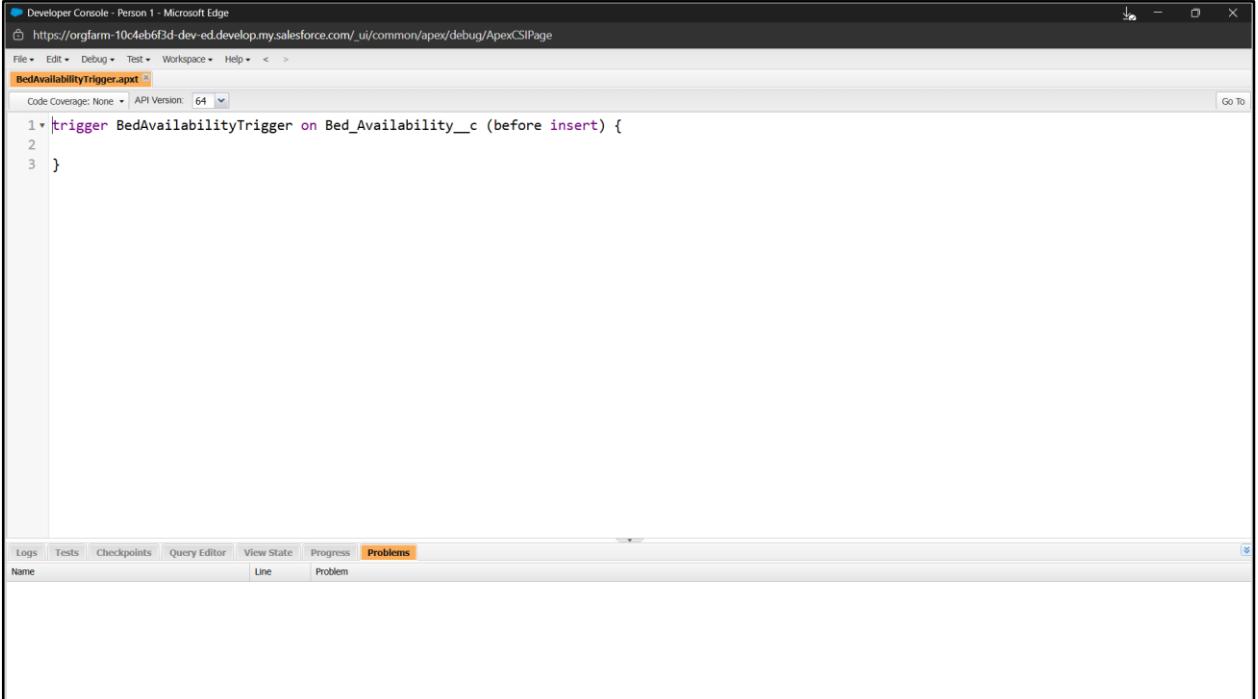


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge". The URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar at the top has "AppointmentTrigger.apxt" selected. Below the tabs, there are dropdown menus for "File", "Edit", "Debug", "Test", "Workspace", and "Help". The main editor area contains the following Apex trigger code:

```
trigger AppointmentTrigger on Appointment__c (before insert) {  
}  
}
```

The status bar at the bottom shows "Code Coverage: None" and "API Version: 64". The "Logs", "Tests", "Checkpoints", "Query Editor", "View State", "Progress", and "Problems" tabs are present, with "Problems" being the active tab. The "Problems" section is currently empty.

➤ BedAvailabilityTrigger.apxt

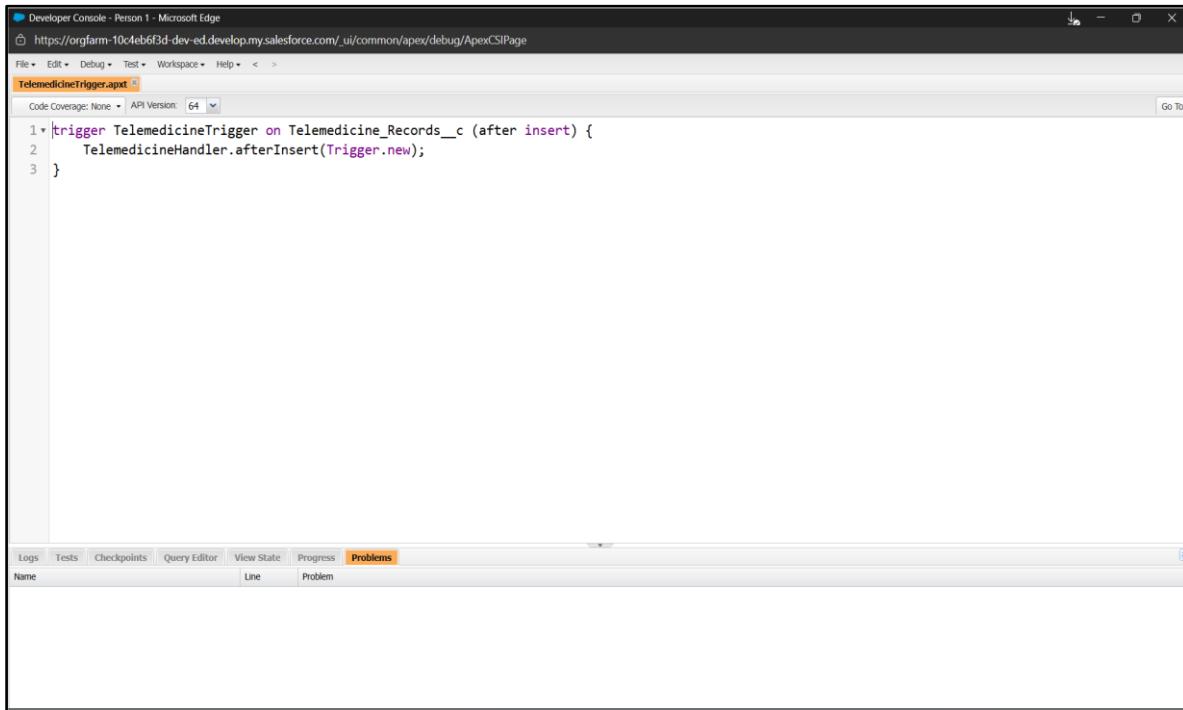


The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge". The URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar at the top has "BedAvailabilityTrigger.apxt" selected. Below the tabs, there are dropdown menus for "File", "Edit", "Debug", "Test", "Workspace", and "Help". The main editor area contains the following Apex trigger code:

```
trigger BedAvailabilityTrigger on Bed_Availability__c (before insert) {  
}  
}
```

The status bar at the bottom shows "Code Coverage: None" and "API Version: 64". The "Logs", "Tests", "Checkpoints", "Query Editor", "View State", "Progress", and "Problems" tabs are present, with "Problems" being the active tab. The "Problems" section is currently empty.

➤ TelemedicineTrigger.apxt



The screenshot shows the Salesforce Developer Console in Microsoft Edge. The URL is https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The tab title is 'TelemedicineTrigger.apxt'. The code editor contains the following Apex trigger:

```
trigger TelemedicineTrigger on Telemedicine__Records_c (after insert) {
    TelemedicineHandler.afterInsert(Trigger.new);
}
```

The console interface includes a toolbar with File, Edit, Debug, Test, Workspace, Help, and a Go To button. Below the toolbar is a dropdown for Code Coverage (None) and API Version (64). The bottom navigation bar has tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems, with Problems selected.

❖ Trigger Design Pattern:

We are already done this in the previous steps. So, no need of doing this.

❖ SOQL & SOSL:

- Go to the Developer Console.
- Click on the Debug and then select the Execute Anonymous Window.
- Type the code and Execute
- Patients:

```
List<Patients__c> patients = [SELECT Id, First_Name__c, Last_Name__c,  
Email__c, Date_of_Birth__c  
FROM Patients__c];  
System.debug('--- Patients ---');  
for(Patients__c p : patients){  
    System.debug('Patient: ' + p.First_Name__c + ' ' + p.Last_Name__c + ',  
Email: ' + p.Email__c + ', DOB: ' + p.Date_of_Birth__c);  
}
```

The screenshot shows the Salesforce Developer Console interface. At the top, there are tabs for 'TelemedicineTrigger.apxc' and 'Log executeAnonymous @9/23/2025, 11:48:31 PM'. The main area displays the 'Execution Log' and the 'Logs' table.

Execution Log

Timestamp	Event	Details
23:48:31:016	USER_DEBUG	[3]DEBUG--- Patients ---
23:48:31:016	USER_DEBUG	[5]DEBUG Patient: Alekya Kali, Email: alekhyakali@gmail.com, DOB: 2005-10-20 00:00:00
23:48:31:016	USER_DEBUG	[5]DEBUG Patient: Ashok N, Email: ashokn@gmail.com, DOB: 2004-09-15 00:00:00
23:48:31:016	USER_DEBUG	[5]DEBUG Patient: Mohit ch, Email: mohit1892@gmail.com, DOB: 2009-02-11 19:00:00
23:48:31:017	USER_DEBUG	[5]DEBUG Patient: Anu P, Email: apeetha0119@gmail.com, DOB: 2002-11-19 00:00:00
23:48:31:017	USER_DEBUG	[5]DEBUG Patient: Gireesha Adari, Email: adarigireesha@gmail.com, DOB: 2003-02-19 00:00:00
23:48:31:017	USER_DEBUG	[5]DEBUG Patient: Avi P, Email: avinash@gmail.com, DOB: 2011-01-11 00:00:00

Logs

User	Application	Operation	Time	Status	Read	Size
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:48:31 PM	Success		8.02 KB
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:45:00 PM	Success		2.69 KB
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:37:05 PM	Success		2.22 KB

➤ Appointments:

```
List<Appointment__c> appointments = [SELECT Id, Patient__c,
Appointment_Date__c, Status__c
FROM Appointment__c];
System.debug('--- Appointments ---');
for(Appointment__c a : appointments){
    System.debug('Appointment Id: ' + a.Id + ', Patient Id: ' + a.Patient__c +
    ' Date: ' + a.Appointment_Date__c + ', Status: ' + a.Status__c);
}
```

The screenshot shows the Salesforce Developer Console interface. At the top, there are tabs for 'Log executeAnonymous @9/23/2025, 11:48:31 PM' and 'Log executeAnonymous @9/24/2025, 12:32:17 AM'. The main area displays the 'Execution Log' and the 'Logs' table.

Execution Log

Timestamp	Event	Details
00:32:17:016	USER_DEBUG	[3]DEBUG--- Appointments ---
00:32:17:016	USER_DEBUG	[5]DEBUG Appointment Id: a04gL000009nPGDQA2, Patient Id: a03gJ00000C0xhNQAV, Date: 2025-09-26 19:00:00, Status: Scheduled
00:32:17:017	USER_DEBUG	[5]DEBUG Appointment Id: a04gL000009nPfQ4M, Patient Id: a03gJ00000C0GhQQAQ, Date: 2025-09-24 19:00:00, Status: Completed
00:32:17:017	USER_DEBUG	[5]DEBUG Appointment Id: a04gL000009nPfQ4M, Patient Id: a03gJ00000C0R7QAN, Date: 2025-09-27 19:00:00, Status: Scheduled

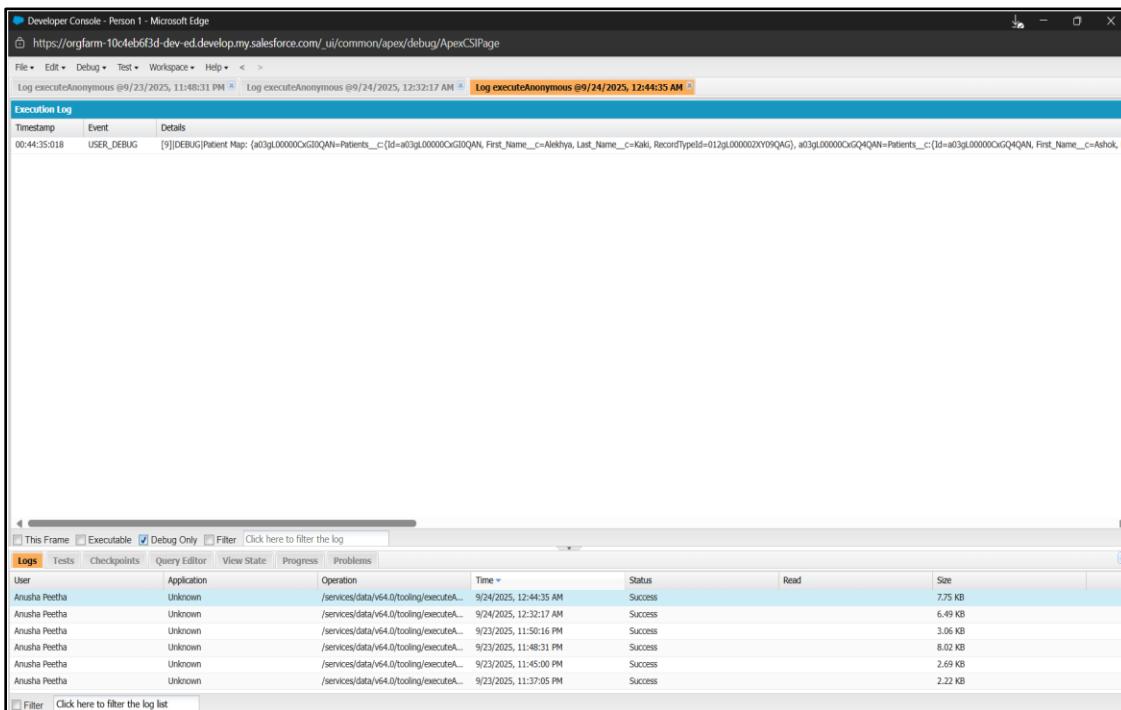
Logs

User	Application	Operation	Time	Status	Read	Size
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/24/2025, 12:32:17 AM	Success		6.49 KB
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:50:16 PM	Success		3.06 KB
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:48:31 PM	Success		8.02 KB
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:45:00 PM	Success		2.69 KB
Anusha Peetha	Unknown	/services/data/v64.0/tooling/executeA...	9/23/2025, 11:37:05 PM	Success		2.22 KB

❖ Collections: List, Set, Map

- Go to Developer Console.
- Click on the Debug and then select the Execute Anonymous Window.
- Type the code and Execute

```
List<Patients__c> patientList = [SELECT Id, First_Name__c, Last_Name__c  
FROM Patients__c];  
Set<Id> patientIds = new Set<Id>();  
Map<Id, Patients__c> patientMap = new Map<Id, Patients__c>();  
  
for(Patients__c p : patientList){  
    patientIds.add(p.Id);  
    patientMap.put(p.Id, p);  
}  
System.debug('Patient Map: ' + patientMap);
```



❖ Control Statements:

- Go to Developer Console.
- Click on the Debug and then select the Execute Anonymous Window.
- Type the code and Execute

```
for(Appointment__c a : [SELECT Id, Status__c FROM Appointment__c]){|  
    if(a.Status__c == 'Scheduled'){|  
        System.debug('Send reminder for appointment: ' + a.Id);  
    } else {
```

```

        System.debug('No action needed: ' + a.Id);
    }
}

```

The screenshot shows the Salesforce Developer Console interface. At the top, there's a navigation bar with links like File, Edit, Debug, Test, Workspace, Help, and a back/forward button. Below the navigation bar is a status bar showing the URL https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage and the log message "Log executeAnonymous @9/24/2025, 12:50:00 AM".

The main area contains two tabs: "Execution Log" and "Logs". The "Execution Log" tab shows a table of log entries:

Timestamp	Event	Details
00:50:00:020	USER_DEBUG	[3] DEBUG Send reminder for appointment: a04gL000009nPGDQA2
00:50:00:020	USER_DEBUG	[5] DEBUG No action needed: a04gL000009nPhQAM
00:50:00:020	USER_DEBUG	[3] DEBUG Send reminder for appointment: a04gL000009nPmQAM

The "Logs" tab shows a table of logs from a user named Anusha Peetha:

User	Application	Operation	Time	Status	Read	Size
Anusha Peetha	Unknown	/services/data/v44.0/tooling/executeA...	9/24/2025, 12:50:00 AM	Success		6.74 kB
Anusha Peetha	Unknown	/services/data/v44.0/tooling/executeA...	9/24/2025, 12:44:35 AM	Success		7.75 kB
Anusha Peetha	Unknown	/services/data/v44.0/tooling/executeA...	9/24/2025, 12:32:17 AM	Success		6.49 kB
Anusha Peetha	Unknown	/services/data/v44.0/tooling/executeA...	9/23/2025, 11:50:16 PM	Success		3.06 kB
Anusha Peetha	Unknown	/services/data/v44.0/tooling/executeA...	9/23/2025, 11:48:31 PM	Success		8.02 kB
Anusha Peetha	Unknown	/services/data/v44.0/tooling/executeA...	9/23/2025, 11:45:00 PM	Success		2.69 kB

❖ Batch Apex:

- Go to Developer Console
- Click on the File and select the New and then Apex class.

The screenshot shows the Salesforce Developer Console with the code editor open. The file name is "PatientBatch.apxc". The code implements the Database.Batchable<SObject> interface:

```

1  public class PatientBatch implements Database.Batchable<SObject> {
2      public Database.QueryLocator start(Database.BatchableContext BC) {
3          return Database.getQueryLocator([SELECT Id, First_Name__c, Last_Name__c, Date_of_Birth__c, Medical_History__c FROM Patients__c]);
4      }
5      public void execute(Database.BatchableContext BC, List<Patients__c> scope) {
6          for(Patients__c p : scope){
7              Integer age = PatientHandler.calculateAge(p.Date_of_Birth__c);
8              if(age != null){
9                  p.Medical_History__c = (p.Medical_History__c != null ? p.Medical_History__c + ' | ' : '') + 'Age: ' + age;
10             }
11         }
12         PatientHandler.updatePatients(scope);
13     }
14     public void finish(Database.BatchableContext BC) {
15         System.debug('Patient batch processing completed.');
16     }
17 }
18

```

The code editor has tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is currently selected, showing the message "Name".

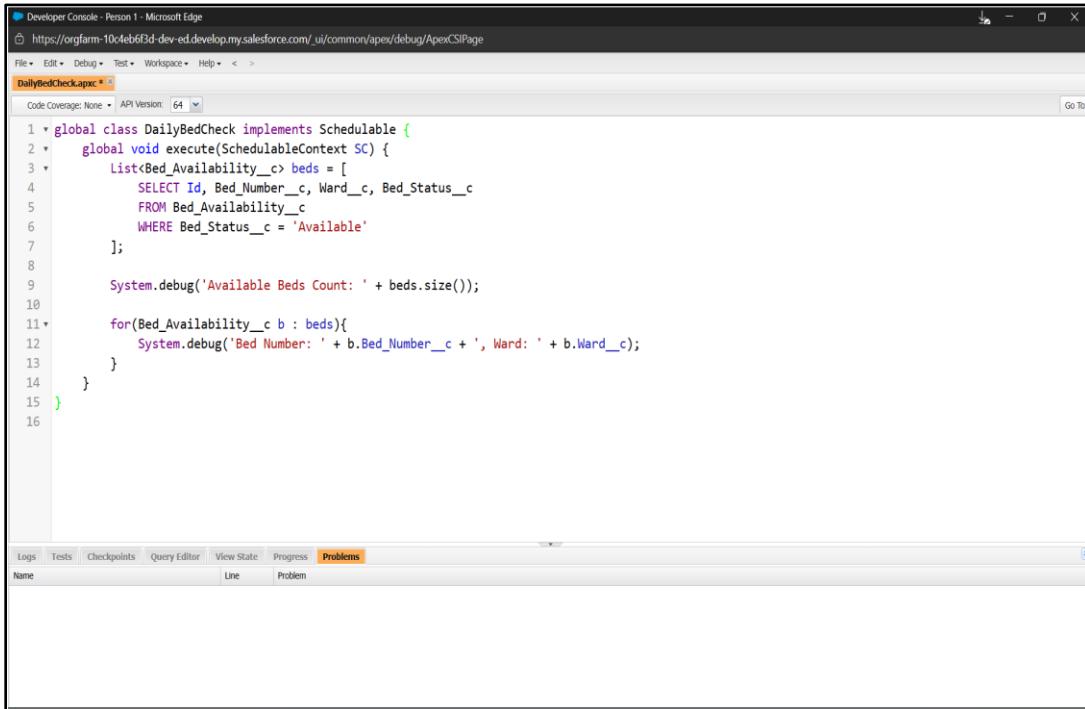
- Click on the Debug and then select the Execute Anonymous Window.

- Type the code and then Execute.

```
Database.executeBatch(new PatientBatch(), 200);
```

❖ **Scheduled Apex:**

- Go to Developer Console.
- Click on File and New and then select the Apex class.
- Enter the File Name DailyBedCheck
- Type the code,



The screenshot shows the Salesforce Developer Console in Microsoft Edge. The URL is https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage. The tab title is "Developer Console - Person 1 - Microsoft Edge". The code editor contains the following Apex class:

```
1 * global class DailyBedCheck implements Schedulable {
2 *     global void execute(SchedulableContext SC) {
3 *         List<Bed_Availability__c> beds = [
4 *             SELECT Id, Bed_Number__c, Ward__c, Bed_Status__c
5 *             FROM Bed_Availability__c
6 *             WHERE Bed_Status__c = 'Available'
7 *         ];
8 *
9 *         System.debug('Available Beds Count: ' + beds.size());
10 *
11 *         for(Bed_Availability__c b : beds){
12 *             System.debug('Bed Number: ' + b.Bed_Number__c + ', Ward: ' + b.Ward__c);
13 *         }
14 *     }
15 * }
16 *
```

The code implements the Schedulable interface and defines an execute method that queries available beds and logs their details. The code editor has syntax highlighting and line numbers. Below the code editor is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected, showing a table with columns: Name, Line, and Problem. The table is currently empty.

- Save the file.
- Now click on the Debug and select the Execute Anonymous Window.
- Type the code and then Execute.

```
System.schedule('Daily Bed Check', '0 0 8 * * ?', new DailyBedCheck());
```

❖ **Future Methods:**

- Go to Developer Console.
- Click on the File and New and then select the Apex class.
- Enter the File Name PaymentNotification.
- Type the code,

```

1 public class PaymentNotification {
2
3     @Future
4     public static void sendAlerts(Set<Id> patientIds){
5         System.debug('⚡ Future method triggered with Patient Ids: ' + patientIds);
6
7         List<Payments_c__c> payments = [
8             SELECT Id, Patient__c, Status__c, Amount__c
9             FROM Payments_c__c
10            WHERE Patient__c IN :patientIds
11        ];
12
13        System.debug('⚡ Payments fetched: ' + payments.size());
14
15        for(Payments_c__c p : payments){
16            System.debug('Send payment alert → Patient Id: ' + p.Patient__c +
17                ', Amount: ' + p.Amount__c +
18                ', Status: ' + p.Status__c);
19        }
20    }
21 }

```

- Save the file.
- Now click on the Debug and select the Execute Anonymous Window.
- Type the code and Execute

```

List<Patients__c> patientList = [SELECT Id FROM Patients__c LIMIT 5];
Set<Id> patientIds = new Set<Id>();
for(Patients__c p : patientList){
    patientIds.add(p.Id);
}
PaymentNotification.sendAlerts(patientIds);

```

❖ Exception Handling:

- Go to Developer Console
- Click on Debug and select the Execute Anonymous Window.
- Type the code and Execute

Patients:

```

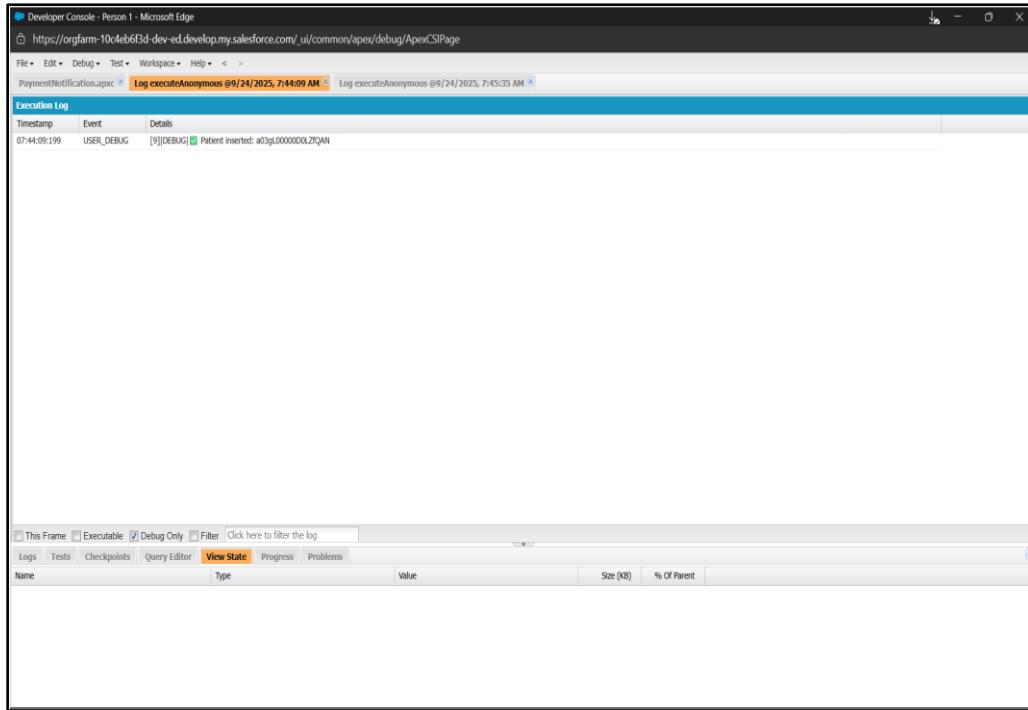
try {
    Patients__c p = new Patients__c(
        First_Name__c = 'John'
    );

```

```

        insert p;
        System.debug('✓ Patient inserted: ' + p.Id);
    } catch(DmlException e) {
        System.debug('⚠ Error inserting patient: ' + e.getMessage());
    }

```

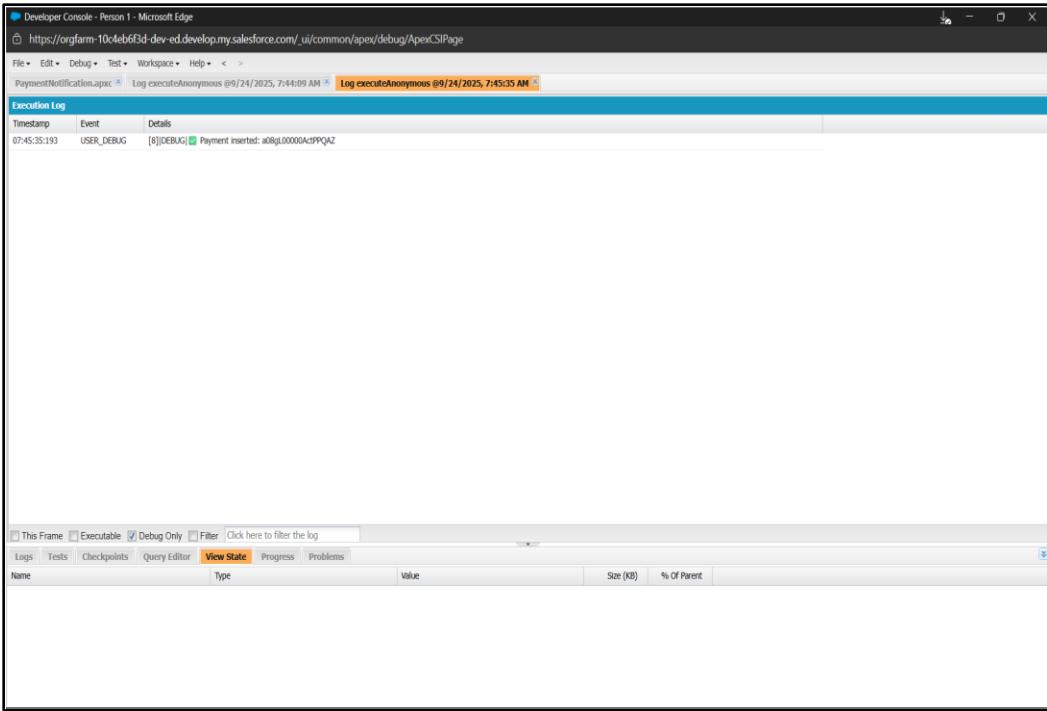


Payments:

```

try {
    Payments__c pay = new Payments__c(
        Patient__c = null,
        Amount__c = 1000
    );
    insert pay;
    System.debug('✓ Payment inserted: ' + pay.Id);
} catch(DmlException e) {
    System.debug('⚠ Error inserting payment: ' + e.getMessage());
}

```



❖ Test Classes:

- Go to Debug Console.
- Click on File and New and then select the Execute Anonymous Window.
- Enter the File Name HealthCareProjectTest
- Type the code and save it.

The screenshot shows the Salesforce Developer Console in Microsoft Edge. The URL is https://orgfarm-104eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The title bar says 'Developer Console - Person 1 - Microsoft Edge'. The main area displays the code editor for the file 'HealthCareProjectTest.apxc'. The code implements a test class with methods for inserting a patient and an appointment, and asserting their correctness. The code editor includes syntax highlighting and a status bar at the bottom.

```

1  // Test Class for HealthCareProject
2
3  public class HealthCareProjectTest {
4
5      // Test Data
6      private static Patient__c p;
7      private static Appointment__c a;
8
9      // Test Method: Insert Patient and Appointment
10     @IsTest
11     public static void insertPatientAndAppointment() {
12         // Create Patient
13         p = new Patient__c();
14         p.First_Name__c = 'Test';
15         p.Last_Name__c = 'User';
16         insert p;
17
18         // Create Appointment
19         a = new Appointment__c();
20         a.Patient__c = p.Id;
21         a.Status__c = 'Scheduled';
22         insert a;
23
24         // Assertions
25         System.assert(p.Id != null, 'Patient inserted successfully');
26         System.assert(a.Id != null, 'Appointment inserted successfully');
27
28         // Query Appointment
29         Appointment__c queriedAppointment = [
30             SELECT Id, Patient__c, Status__c
31             FROM Appointment__c
32             WHERE Id = :a.Id];
33
34         // Assertions
35         System.assertEquals(p.Id, queriedAppointment.Patient__c, 'Patient is correctly linked');
36         System.assertEquals('Scheduled', queriedAppointment.Status__c, 'Appointment status is correct');
37     }
38 }

```

- Click on the Test and select the New Run.
- The output will be Success.

Salesforce Project Phase-6

Health Patient Management and Telemedicine CRM

Phase 6: User Interface Development

❖ Lightning App Builder and Record Pages:

- Go to Setup.
- In the Quick Find box, type the Lightning App Builder and click on it.
- Click on New and select Record Page.
- Patient:
 - Page Name: Patient Record Page
 - Object: Patient _c
 - Next
 - Choose Grouped header Template
 - Drag components into Layout: Record detail, Field Section, Related List – Single – Appointment _c, Related List – Single – Bed Availability _c
 - Select Set Dynamic Visibility: Field: Status, Operator: =, Value: Admitted
 - Save and then Activate and close
- Appointment:
 - Page Name: Appointment Record Page
 - Object: Appointment _c
 - Next
 - Choose Grouped header Template
 - Drag components into Layout: Record detail, Field Section, Related Record
 - Save and then Activate and close
- Hospital Dashboard App Page:
 - Instead of selecting Record Page select the App page
 - Page Name: Hospital Dashboard
 - Next
 - Choose 2 Regions
 - Drag components into Layout:

1. Left column:
Report Column, List View
2. Right column:
List View
Save and then Activate and close

The screenshot shows the Salesforce Lightning App Builder interface. On the left, there's a navigation sidebar with various setup options like Setup Home, Service Setup Assistant, and Field Service Setup Home (Beta). The main area is titled 'Lightning App Builder' and displays a table of 'Lightning Pages'. The table has columns for Action, Label, Name, Namespace Prefix, Description, Type, Created By, and Last Modified By. There are three entries in the table:

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
Edit Clone Del	Appointment Record Page	Appointment_Record_Page			Record Page	anu 9/23/2025, 11:28 PM	anu 9/23/2025, 11:31 PM
Edit Clone Del	Hospital Dashboard	Hospital_Dashboard			App Page	anu 9/23/2025, 11:35 PM	anu 9/23/2025, 11:35 PM
Edit Clone Del	Patient Record Page	Patient_Record_Page			Record Page	anu 9/23/2025, 11:19 PM	anu 9/23/2025, 11:20 PM

❖ Tabs:

- We already created the required tabs in the previous steps.
- Now we have to add to the Lightning Builder.
- Go to the Setup.
- In the Quick Find box, type the App Manager.
- Near HealthCare in the drop menu click on Edit.
- Go to Navigation Items
- Select the Dashboard to the selected items and arrange them in required way.
- Click on save.

❖ Home Page Layouts:

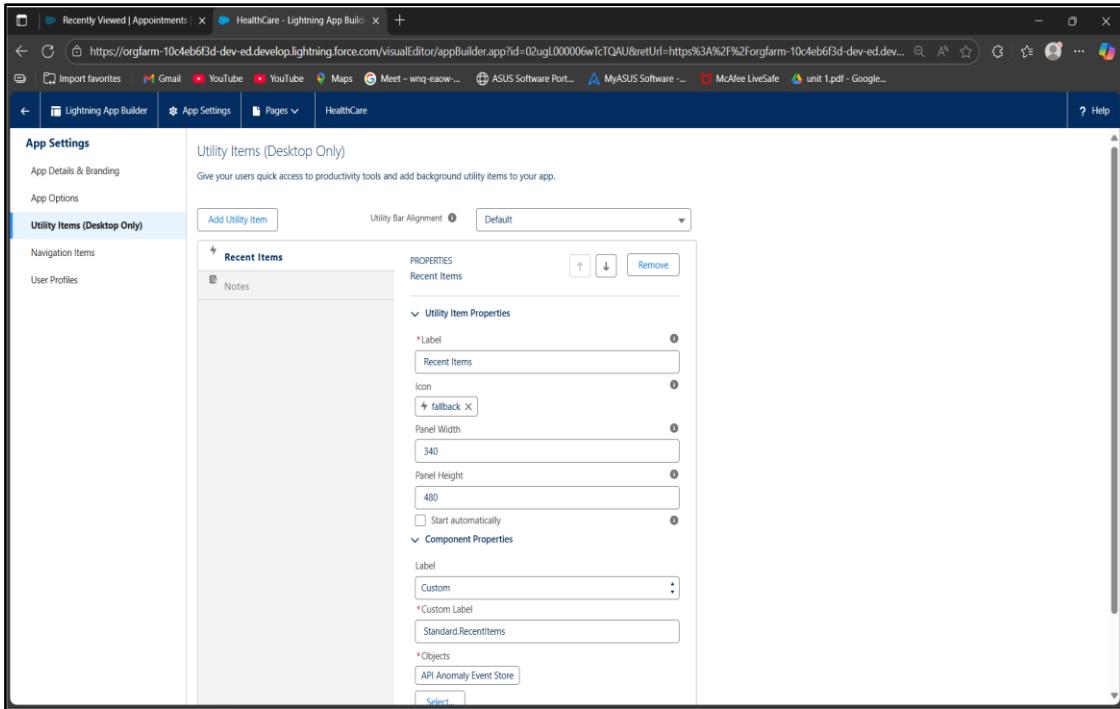
- Go to Setup.
- In the Quick Find box, type the lightning app Builder and click it.
- Click on New.
- Select the Home Page and click Next.
- Label: HealthCare staff Home
- Choose template: Header and three regions
- Click Next.
- Drag the components,
Left: List View, Middle: Report chart, Right: List View
- Click on save and Activate
- Select the App Default and then Assign Apps
- Select the HealthCare and save.

The screenshot shows the Salesforce Lightning App Builder interface. On the left, there's a sidebar with various categories like Lightning Bolt, Feature Settings, Objects and Fields, User Interface, and Custom Code. Under User Interface, 'Lightning App Builder' is selected and highlighted in blue. The main area is titled 'Lightning App Builder' and contains a brief description of what it does. Below that is a table titled 'Lightning Pages' with columns for Action, Label, Name, Namespace Prefix, Description, Type, Created By, and Last Modified By. The table lists several pages, including 'Appointment Record Page', 'Bed_Availability_Record_Page', 'HealthCare_Staff_Home', 'Hospital_Dashboard', 'Patient_Record_Page', and 'Patient_Record_Page1'. At the bottom of the page, there's a search bar and a note: 'Didn't find what you're looking for? Try using Global Search.'

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
Edit Clone Del	Appointment Record Page	Appointment_Record_Page			Record Page	anu. 9/23/2025, 11:28 PM	anu. 9/23/2025, 11:31 PM
Edit Clone Del	Appointment Record Page	Appointment_Record_Page1			Record Page	anu. 9/24/2025, 11:14 PM	anu. 9/24/2025, 11:14 PM
Edit Clone Del	Bed Availability Record Page	Bed_Availability_Record_Page			Record Page	anu. 9/24/2025, 11:16 PM	anu. 9/24/2025, 11:16 PM
Edit Clone Del	HealthCare Staff Home	HealthCare_Staff_Home			Home Page	anu. 9/24/2025, 11:56 PM	anu. 9/24/2025, 11:56 PM
Edit Clone Del	Hospital Dashboard	Hospital_Dashboard			App Page	anu. 9/23/2025, 11:35 PM	anu. 9/23/2025, 11:35 PM
Edit Clone Del	Patient Record Page	Patient_Record_Page			Record Page	anu. 9/23/2025, 11:19 PM	anu. 9/23/2025, 11:20 PM
Edit Clone Del	Patient Record Page	Patient_Record_Page1			Record Page	anu. 9/24/2025, 10:53 PM	anu. 9/24/2025, 10:53 PM

❖ Utility Bar:

- Go to Setup.
- In the Quick Find box, type the App Manager and click it.
- Search the HealthCare and click on edit.
- Click the Utility Items.
- Click on Add Utility Items and select the Notes and Recent Items.
- Click on save.



❖ **LWC (Lightning Web Components):**

- Open VS code with Salesforce Extension Pack.
- Press ctrl + shift + P and select the SFDX: Create Lightning Web Component
- Name it as patientQuickSearch
- Right click on the file select the SFDX: Deploy this source to org
- Repeat the steps for AppointmentReminder and BedAvailabilityChart

❖ **Apex with LWC:**

- Open the VS code with Salesforce Extension Pack
- Press ctrl + shift + P and select SFDX: Create Apex class
- Name the class PatientController
- Type the code:

```

force-app > main > default > classes > PatientController.cls ...
1  public with sharing class PatientController {
2    @AuraEnabled(cacheable=true)
3    public static (List<Patients__c> > searchClients(String searchKey){
4      searchKey = '%' + searchKey + '%';
5      return [SELECT Id, First_Name__c, Last_Name__c, Status__c, Assign_Docor__c
6              FROM Patients__c
7              WHERE First_Name__c LIKE :searchKey OR Last_Name__c LIKE :searchKey
8              LIMIT 50];
9    }
10 }
11

```

OUTPUT DEBUG CONSOLE TERMINAL PORTS HISTORY PROJECT PATH

STATE FULL NAME TYPE PROJECT PATH

Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.html
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js-meta.xml

13:11:23.493 Ended SFDX: Deploy This Source to Org
 13:21:44.392 Starting SFDX: Deploy This Source to Org

==== Deployed Source
 STATE FULL NAME TYPE PROJECT PATH

Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.html
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js-meta.xml

13:21:47.577 Ended SFDX: Deploy This Source to Org

Do you want to install the recommended extensions from Red Hat, Prettier and others for this repository? [Install](#) [Show Recommendations](#)

- Right click on the file and Select the SFDX: Deploy this source to Org
- Repeat for the AppointmentController and BedController

```

force-app > main > default > classes > AppointmentController.cls ...
1  public with sharing class AppointmentController {
2    @AuraEnabled(cacheable=true)
3    public static (List<Appointment__c> > getTodayAppointments() {
4      return [SELECT Id, Patient__c, Status__c, Appointment_Date__c
5              FROM Appointment__c
6              WHERE Appointment_Date__c = TODAY];
7    }
8  }
9

```

OUTPUT DEBUG CONSOLE TERMINAL PORTS HISTORY PROJECT PATH

STATE FULL NAME TYPE PROJECT PATH

Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.html
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js-meta.xml

13:11:23.493 Ended SFDX: Deploy This Source to Org
 13:21:44.392 Starting SFDX: Deploy This Source to Org

==== Deployed Source
 STATE FULL NAME TYPE PROJECT PATH

Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.html
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js
 Changed patientQuickSearch LightningComponentBundle force-app\main\default\lwc\patientQuickSearch\patientQuickSearch.js-meta.xml

13:21:47.577 Ended SFDX: Deploy This Source to Org

Do you want to install the recommended extensions from Red Hat, Prettier and others for this repository? [Install](#) [Show Recommendations](#)

```

force-app > main > default > classes > BedController.cls > ...
1 public with sharing class BedController {
2     @AuraEnabled(cacheable=true)
3     public static List<Bed_Availability__c> getBeds() {
4         return [SELECT Id, Bed_Number__c, Bed_Status__c FROM Bed_Availability__c];
5     }
6 }
7

```

➤ Connect Apex to the LWC:

Update the codes for patientQuickSearch.js, patientQuickSearch.html, patientQuicksearch.js-meta.xml and right click on the file and select the SFDX: Deploy this source to Org

```

force-app > main > default > lwc > patientQuickSearch > JS patientQuickSearch.js > ...
1 import { LightningElement, track } from 'lwc';
2 import searchPatients from '@salesforce/apex/PatientController.searchPatients';
3
4 export default class PatientQuickSearch extends LightningElement {
5     @track searchKey = '';
6     @track patients = [];
7
8     columns = [
9         { label: 'First Name', fieldName: 'First_Name__c' },
10        { label: 'Last Name', fieldName: 'Last_Name__c' },
11        { label: 'Status', fieldName: 'Status__c' },
12        { label: 'Assigned Doctor', fieldName: 'Assign_Doctor__c' }
13    ];
14
15    handleKeyChange(event) {
16        this.searchKey = event.target.value;
17    }
18
19    handleSearch() {
20        searchPatients({ searchKey: this.searchKey })
21            .then(result => { this.patients = result; })
22            .catch(error => { console.error(error); });
23    }
24 }
25

```

```

<?xml version="1.0" encoding="UTF-8"?>
<lightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>64.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning_RecordPage</target>
        <target>lightning_HomePage</target>
        <target>lightning_AppPage</target>
    </targets>
    <targetConfigs>
        <targetConfig targets="lightning_RecordPage">
            <objects>
                <object>Patients__c</object>
            </objects>
        </targetConfig>
    </targetConfigs>
</lightningComponentBundle>

```

```

<template>
    <lightning-card title="Patient Quick Search">
        <div class="slds-m-around_medium">
            <lightning-input label="Search Patient"
                value={searchKey}
                onchange={handleKeyChange}
                placeholder="Enter patient name">
            </lightning-input>
            <lightning-button label="Search" onclick={handleSearch}></lightning-button>
        </div>
        <template if:true={patients}>
            <lightning-datatable
                key-field="id"
                data={patients}
                columns={columns}>
            </lightning-datatable>
        </template>
    </lightning-card>
</template>

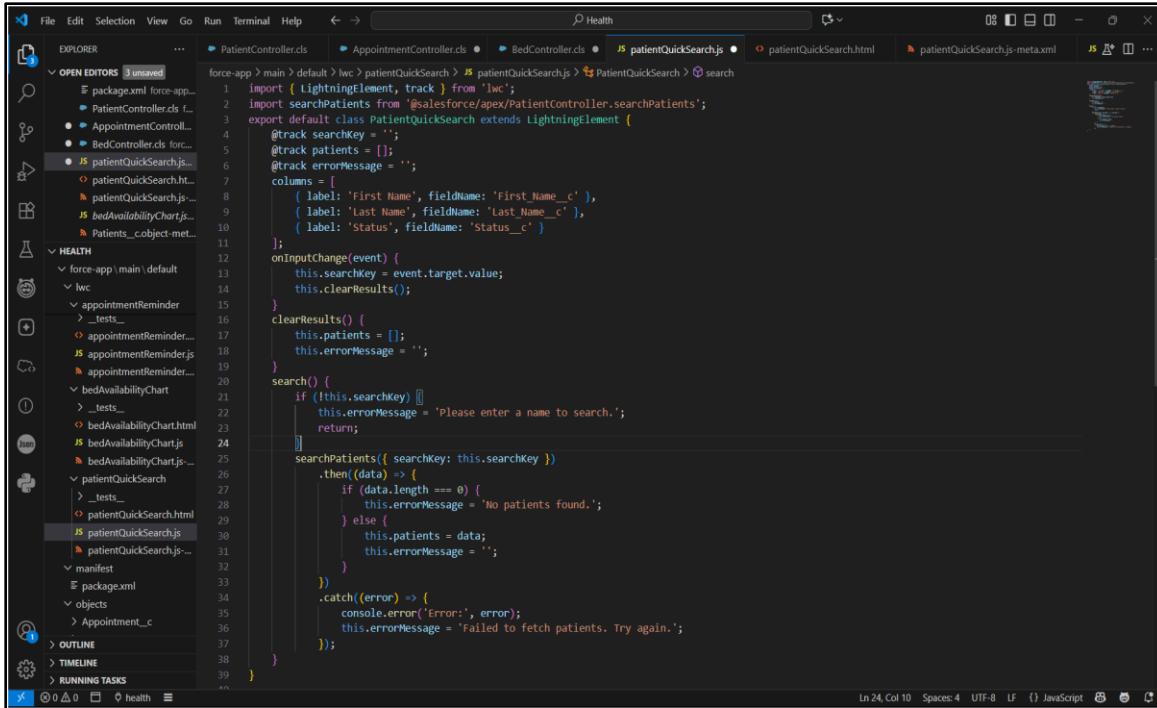
```

❖ Events in LWC and Wire Adapters:

These are already done no need do now.

❖ Imperative Apex Calls:

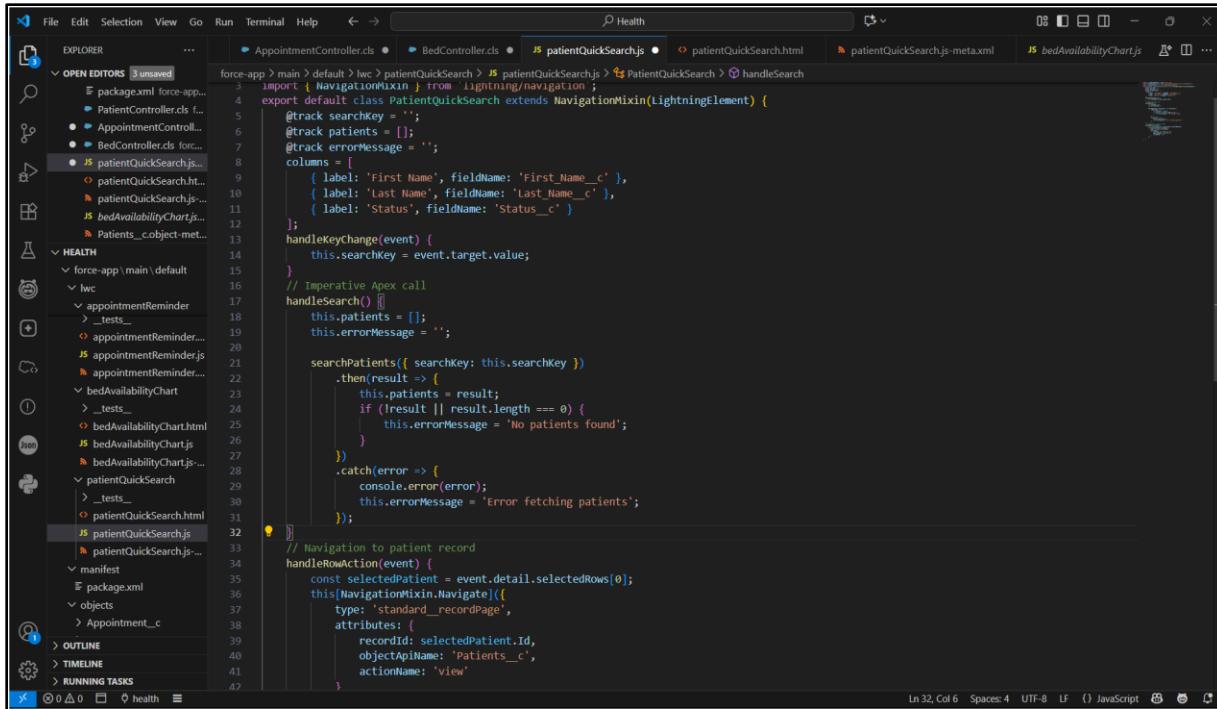
- Update the patientQuickSearch.js code in VS code
- Right click on the code and select the Deploy this source to Org



```
force-app > main > default > lwc > patientQuickSearch > JS patientQuickSearch.js > PatientQuickSearch > search
1 import { LightningElement, track } from 'lwc';
2 import searchPatients from '@salesforce/apex/PatientController.searchPatients';
3 export default class PatientQuickSearch extends LightningElement {
4     @track searchKey = '';
5     @track patients = [];
6     @track errorMessage = '';
7     columns = [
8         { label: 'First Name', fieldName: 'First_Name_c' },
9         { label: 'Last Name', fieldName: 'Last_Name_c' },
10        { label: 'Status', fieldName: 'Status_c' }
11    ];
12    onInputChange(event) {
13        this.searchKey = event.target.value;
14        this.clearResults();
15    }
16    clearResults() {
17        this.patients = [];
18        this.errorMessage = '';
19    }
20    search() {
21        if (!this.searchKey) {
22            this.errorMessage = 'Please enter a name to search.';
23            return;
24        }
25        searchPatients({ searchKey: this.searchKey })
26            .then(data) => {
27                if (data.length === 0) {
28                    this.errorMessage = 'No patients found.';
29                } else {
30                    this.patients = data;
31                    this.errorMessage = '';
32                }
33            }
34            .catch(error) => {
35                console.error('Error:', error);
36                this.errorMessage = 'Failed to fetch patients. Try again.';
37            });
38    }
39 }
```

❖ Navigation Service:

- Update the patientQuickSearch.js code in VS code
- After updating right click on the file and select SFDX: Deploy this source to Org
- Wait until the successfully message done



```
force-app > main > default > lwc > patientQuickSearch > JS patientQuickSearch.js > PatientQuickSearch > handleSearch
3 import { NavigationMixin } from 'lightning/navigation';
4 export default class PatientQuickSearch extends NavigationMixin(LightningElement) {
5     @track searchKey = '';
6     @track patients = [];
7     @track errorMessage = '';
8     columns = [
9         { label: 'First Name', fieldName: 'First_Name_c' },
10        { label: 'Last Name', fieldName: 'Last_Name_c' },
11        { label: 'Status', fieldName: 'Status_c' }
12    ];
13    handleKeyChange(event) {
14        this.searchKey = event.target.value;
15    }
16    // Imperative Apex call
17    handleSearch() {
18        this.patients = [];
19        this.errorMessage = '';
20
21        searchPatients({ searchKey: this.searchKey })
22            .then(result => {
23                this.patients = result;
24                if (!result || result.length === 0) {
25                    this.errorMessage = 'No patients found';
26                }
27            })
28            .catch(error => {
29                console.error(error);
30                this.errorMessage = 'Error fetching patients';
31            });
32    }
33    // Navigation to patient record
34    handleRowAction(event) {
35        const selectedPatient = event.detail.selectedRows[0];
36        this.NavigationMixin.Navigate({
37            type: 'standard__recordPage',
38            attributes: {
39                recordId: selectedPatient.Id,
40                objectApiName: 'Patients__c',
41                actionName: 'View'
42            }
43        });
44    }
45}
```

Ln 32, Col 6 Spaces: 4 UTF-8 LF {} JavaScript

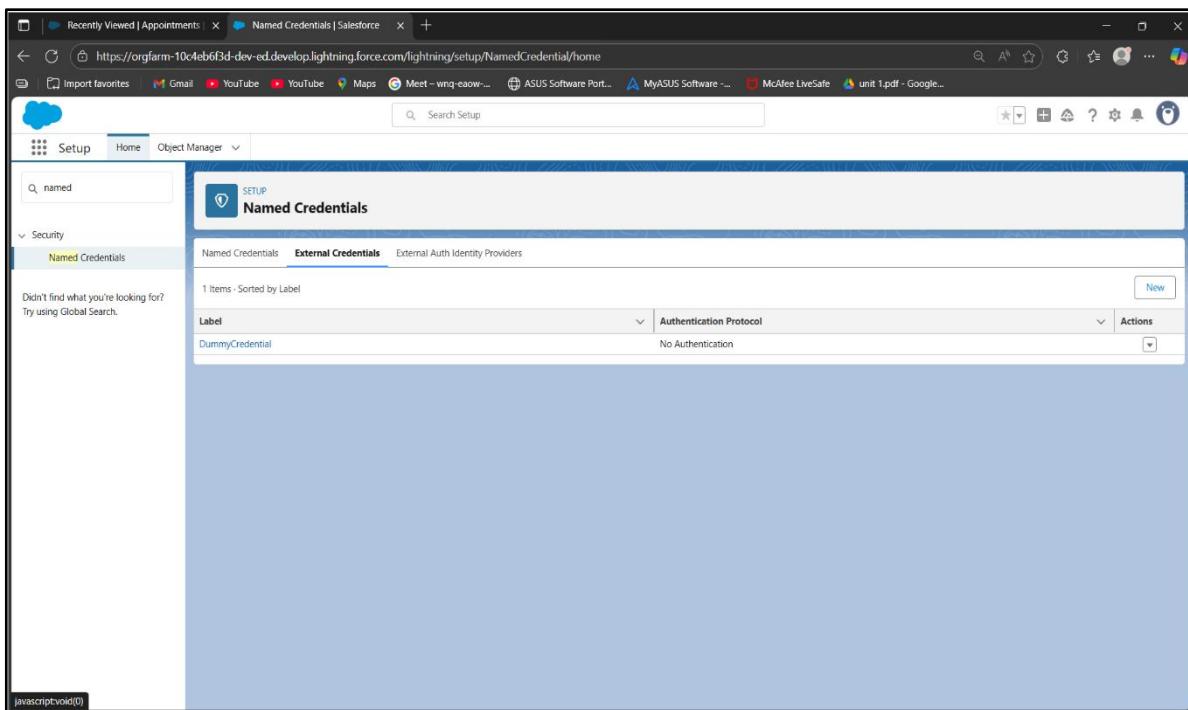
Salesforce Project Phase-7

Health Patient Management and Telemedicine CRM

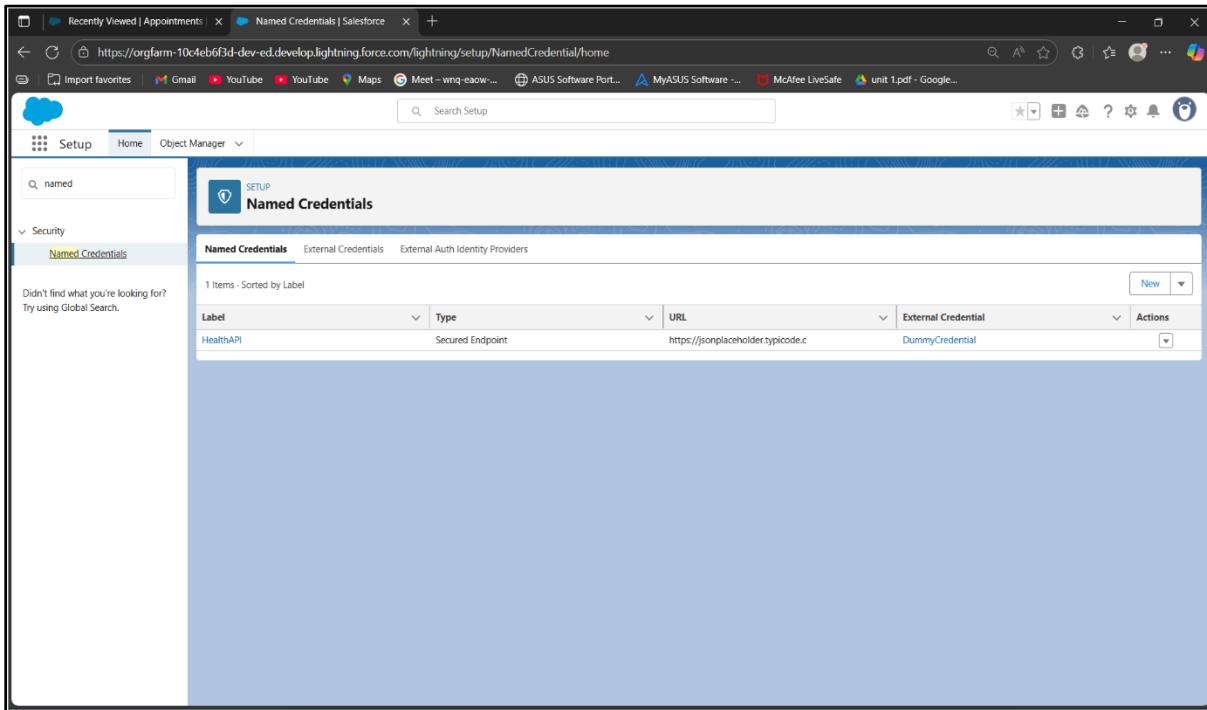
Phase 7: Integration & External Access

❖ Named Credentials:

- Go to the Setup.
- In the Quick Find box, type the Named Credentials.
- Before creating the Named Credential you need to create the External Credential.
- Click on the External Credential and click on New
- Fill the details,
Label and Name: DummyCredential
Authentication Protocol: No Authentication
- Click Save.



- Now, click on the Named Credential and click on New
- Fill the details:
 Label and Name: HealthAPI
 URL: <https://jsonplaceholder.typicode.com>
 External Credential: DummyCredential
 Generate Authorization Header leave unchecked
- Click on Save



- Now create the Apex class with named PatientAPIService
- Type the code and save.
- Right click on the file and select the SFDX: Deploy this source to Org
- Wait until the successful message is displayed.

```

force-app > main > default > classes > PatientAPIService.cls > PatientAPIService > getPatientsFromAPI() - List<Patients__c>
1  public with sharing class PatientAPIService {
2      @AuraEnabled(cacheable=false)
3      public static List<Patients__c> getPatientsFromAPI() {
4          HttpRequest req = new HttpRequest();
5          req.setEndpoint('callout:HealthAPI/users');
6          req.setMethod('GET');
7
8          Http http = new Http();
9          HttpResponse res = http.send(req);
10
11         List<Patients__c> patientList = new List<Patients__c>();
12
13         return patientList;
14     }
15 }
16
17

```

The screenshot shows the VS Code interface with the PatientAPIService.cls file open in the editor. The code implements a class PatientAPIService with a static method getPatientsFromAPI() that sends a GET request to the Health API to retrieve patient data. The code uses the callout:HealthAPI endpoint and the GET method. The response is stored in a List<Patients__c> object.

❖ External Services:

- Open VS Code.
- Create a HealthAPI.json file in patient object.

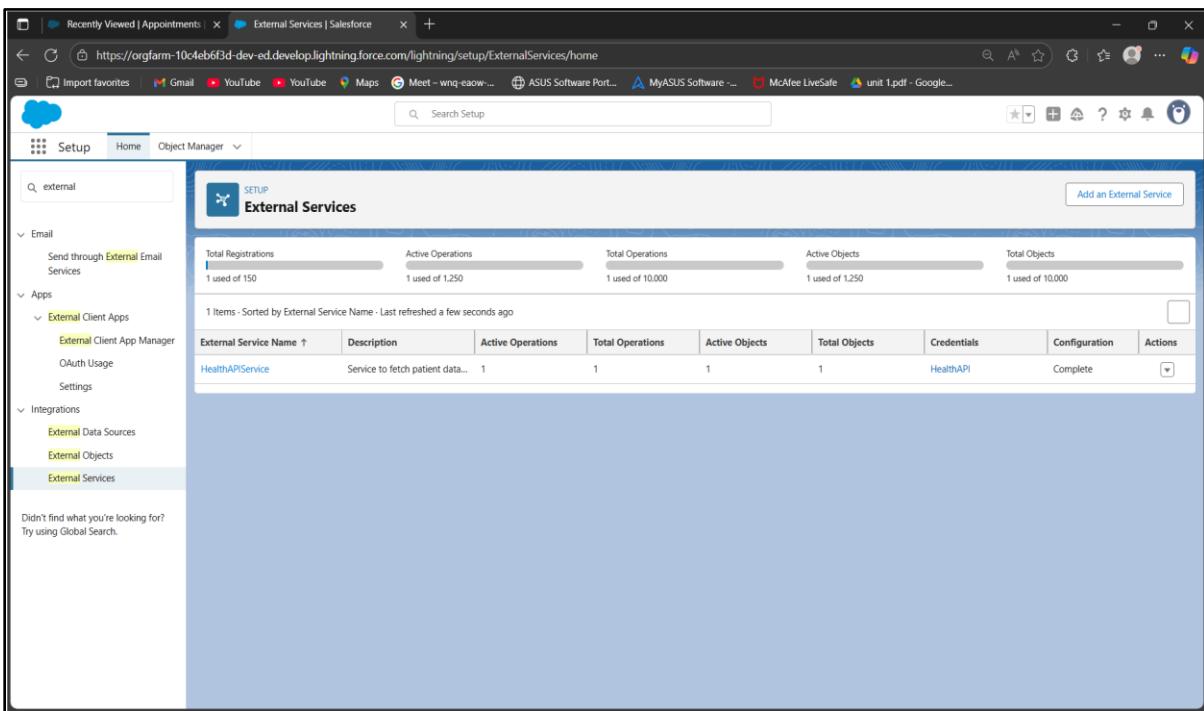
```

{
    "swagger": "2.0",
    "info": {
        "title": "HealthAPI",
        "version": "1.0"
    },
    "paths": {
        "/patients": {
            "get": {
                "summary": "get all patients",
                "responses": {
                    "200": {
                        "description": "list of patients",
                        "schema": {
                            "type": "array",
                            "items": { "$ref": "#/definitions/Patient" }
                        }
                    }
                }
            }
        }
    },
    "definitions": {
        "Patient": {
            "type": "object",
            "properties": {
                "First_Name": { "type": "string" },
                "Last_Name": { "type": "string" },
                "Status": { "type": "string" }
            }
        }
    }
}

```

The screenshot shows the VS Code interface with the HealthAPI.json file open in the editor. The JSON file defines the swagger version, info, paths, and definitions for the patients API. It includes a definition for the Patient object with properties First_Name, Last_Name, and Status.

- Go to Setup.
- In the Quick Find box, type the External Services and click it.
- Now, click on the Add an External Service.
- Select the From API Specification and click on Next.
- Fill the details:
 - External Service Name: HealthAPIService
 - Description: Service to fetch patient data from exter Health API
 - Service Schema: Upload the created JSON file
 - Relative URL: Leave blank
 - Named Credential: HealthAPI
- Click on save
- Select the GetPatients and click Next.
- Click on Finish.



❖ Web Services (REST/SOAP):

Already done this we need to do this.

❖ Callouts:

- Open the VS Code.
- Open the patientAPIService file and update the code as required
- Right click on the file and select the SFDX: Deploy this Source to Org

```

1  public with sharing class PatientAPIService {
2
3      @AuraEnabled(cacheable=false)
4      public static List<Patients__c> getPatientsFromAPI() {
5          List<Patients__c> patientList = new List<Patients__c>();
6
7          try {
8              HttpRequest req = new HttpRequest();
9              req.setEndpoint('callout:HealthAPI/users');
10             req.setMethod('GET');
11             req.setHeader('Content-Type', 'application/json');
12             Http http = new Http();
13             HttpResponse res = http.send(req);
14             if(res.getStatusCode() == 200) {
15                 List<Object> results = (List<Object>) JSON.deserializeUntyped(res.getBody());
16                 for(Object obj : results) {
17                     Map<String, Object> patientMap = (Map<String, Object>) obj;
18                     Patients__c p = new Patients__c();
19                     p.First_Name__c = (String) patientMap.get('first_name');
20                     p.Last_Name__c = (String) patientMap.get('last_name');
21                     p.Status__c = (String) patientMap.get('status');
22                     patientList.add(p);
23                 }
24             } else {
25                 System.debug('API call failed with status code: ' + res.getStatusCode());
26             }
27         } catch(Exception e) {
28             System.debug('Error in API call: ' + e.getMessage());
29         }
30
31     }
32
33 }

```

Ln 21, Col 40 Spaces: 4 UTF-8 LF () Apex

❖ Platform Events:

- Go to the Setup.
- In the Quick Find box, type the Platform Event and click it.
- Now click on the New Platform Event
- Fill the details:
Label: PatientUpdateEvent
Plural Label: PatientUpdateEvent
Publish Behaviour: Publish After Commit
- Click Save
- Scroll down to see the Custom Fields and Relationships
- Click New
- Select the Text, Text and Text Area(Long) respectively
- Fill the details:

Patients

Field Label: PatientId_c

Length: 18

Appointments

Field Label: AppointmentId_c

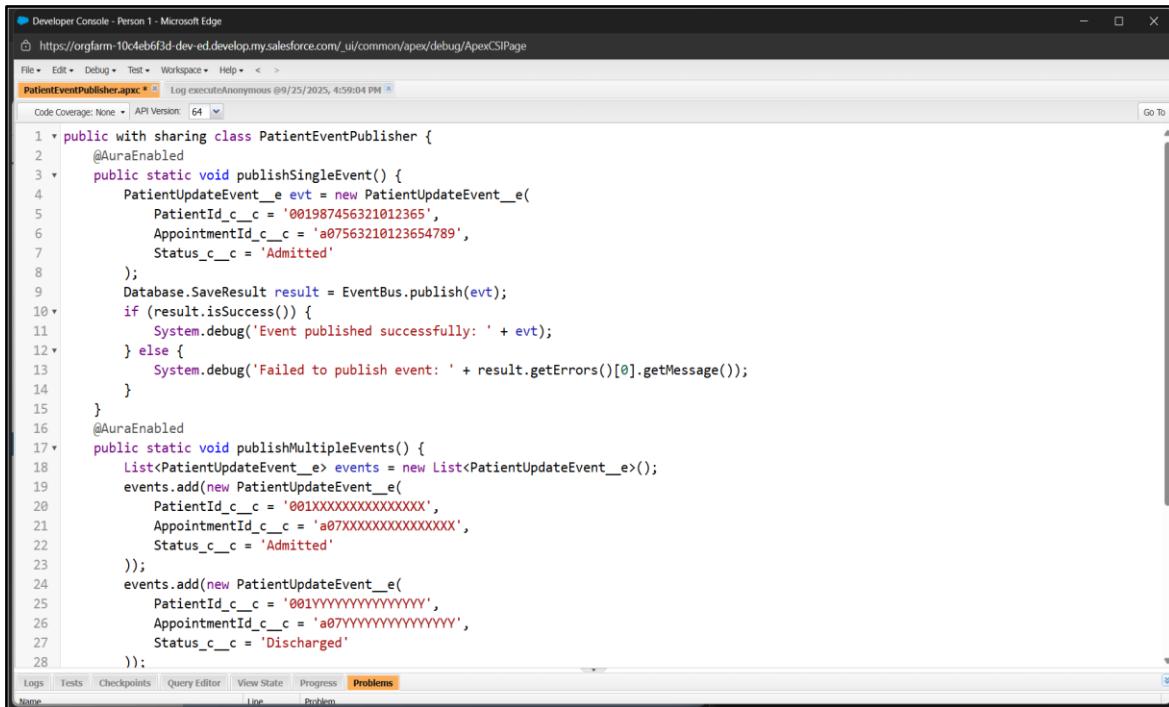
Length: 50

Status

Field Label: Status_c

Length: 32768

- Click Save
- Now go to Developer Console
- Click on File and select New Apex class
- Enter File Name PatientEventPublisher
- Type the code and save it

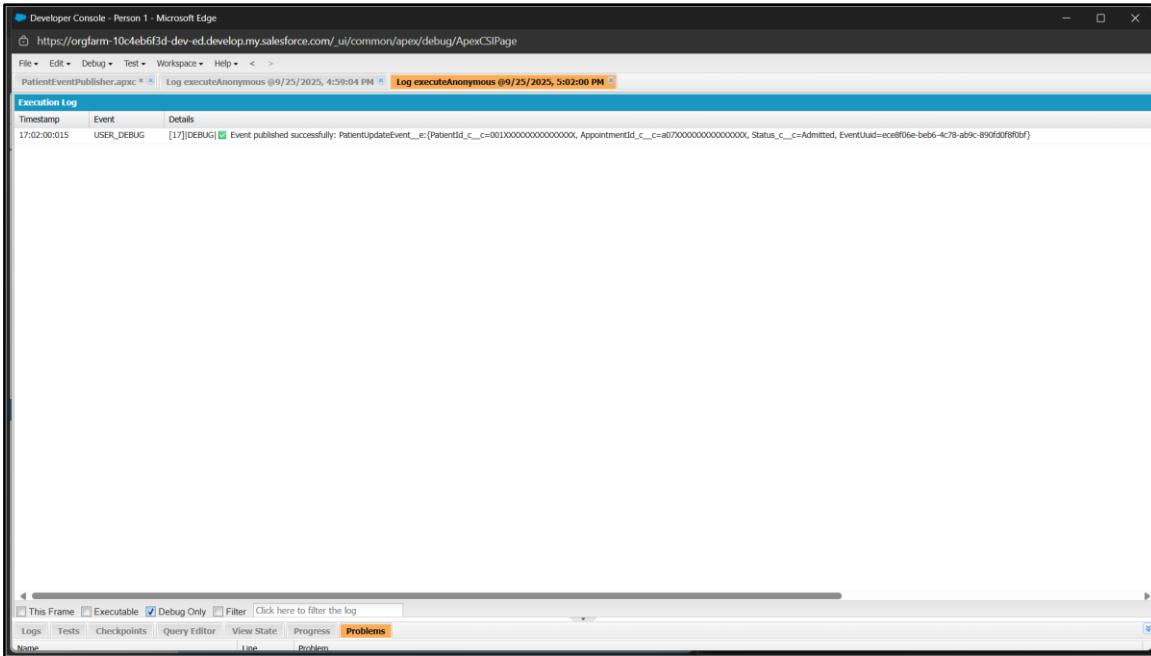


The screenshot shows the Salesforce Developer Console in Microsoft Edge. The URL is https://orgfarm-10ceb63d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The page title is "PatientEventPublisher.apxc". The code editor contains the following Apex code:

```
1 public with sharing class PatientEventPublisher {
2     @AuraEnabled
3     public static void publishSingleEvent() {
4         PatientUpdateEvent__e evt = new PatientUpdateEvent__e(
5             PatientId_c__c = '001987456321012365',
6             AppointmentId_c__c = 'a07563210123654789',
7             Status_c__c = 'Admitted'
8         );
9         Database.SaveResult result = EventBus.publish(evt);
10        if (result.isSuccess()) {
11            System.debug('Event published successfully: ' + evt);
12        } else {
13            System.debug('Failed to publish event: ' + result.getErrors()[0].getMessage());
14        }
15    }
16    @AuraEnabled
17    public static void publishMultipleEvents() {
18        List<PatientUpdateEvent__e> events = new List<PatientUpdateEvent__e>();
19        events.add(new PatientUpdateEvent__e(
20            PatientId_c__c = '001XXXXXXXXXXXXXX',
21            AppointmentId_c__c = 'a07XXXXXXXXXXXXXX',
22            Status_c__c = 'Admitted'
23        ));
24        events.add(new PatientUpdateEvent__e(
25            PatientId_c__c = '001YYYYYYYYYYYYYY',
26            AppointmentId_c__c = 'a07YYYYYYYYYYYYYY',
27            Status_c__c = 'Discharged'
28        ));
29    }
}
```

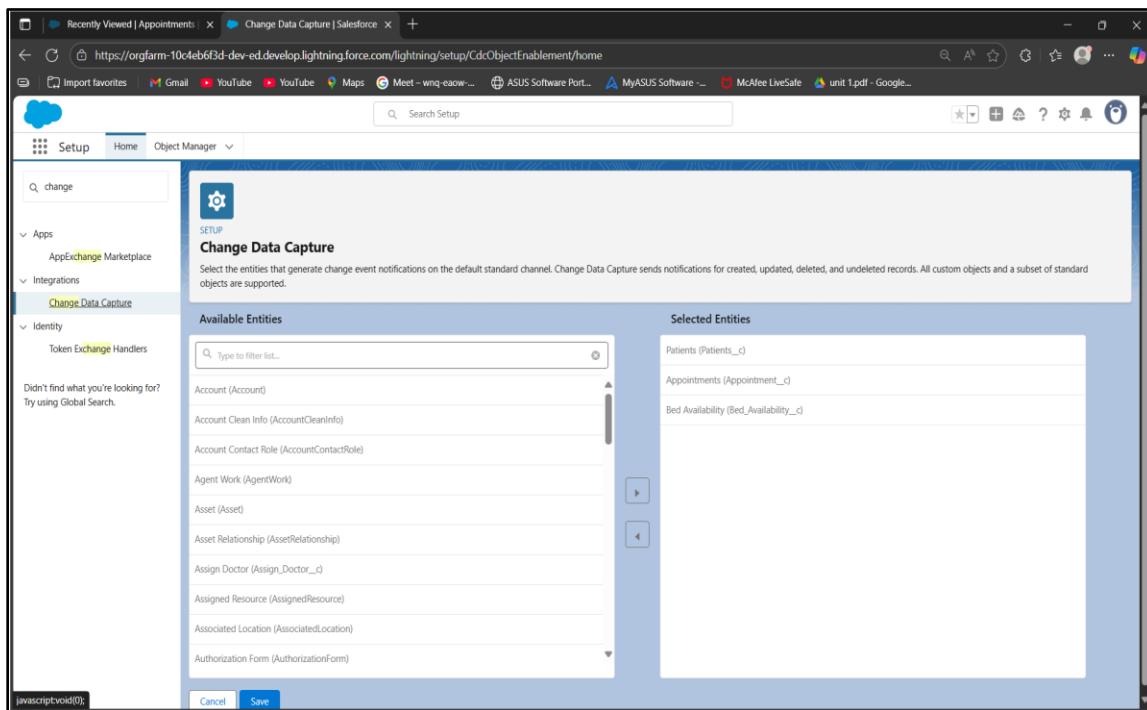
The code defines a class PatientEventPublisher with two methods: publishSingleEvent() and publishMultipleEvents(). Both methods use the EventBus.publish() method to publish PatientUpdateEvent__e objects. The first method publishes one event with PatientId_c__c = '001987456321012365', AppointmentId_c__c = 'a07563210123654789', and Status_c__c = 'Admitted'. The second method publishes two events with PatientId_c__c = '001XXXXXXXXXXXXXX' and '001YYYYYYYYYYYYYY' respectively, and AppointmentId_c__c = 'a07XXXXXXXXXXXXXX' and 'a07YYYYYYYYYYYYYY' respectively, both with Status_c__c = 'Admitted'.

- Now click the Debug and select the Execute the Anonymous window
- PatientEventPublisher.publishSingleEvent();
- Now Execute

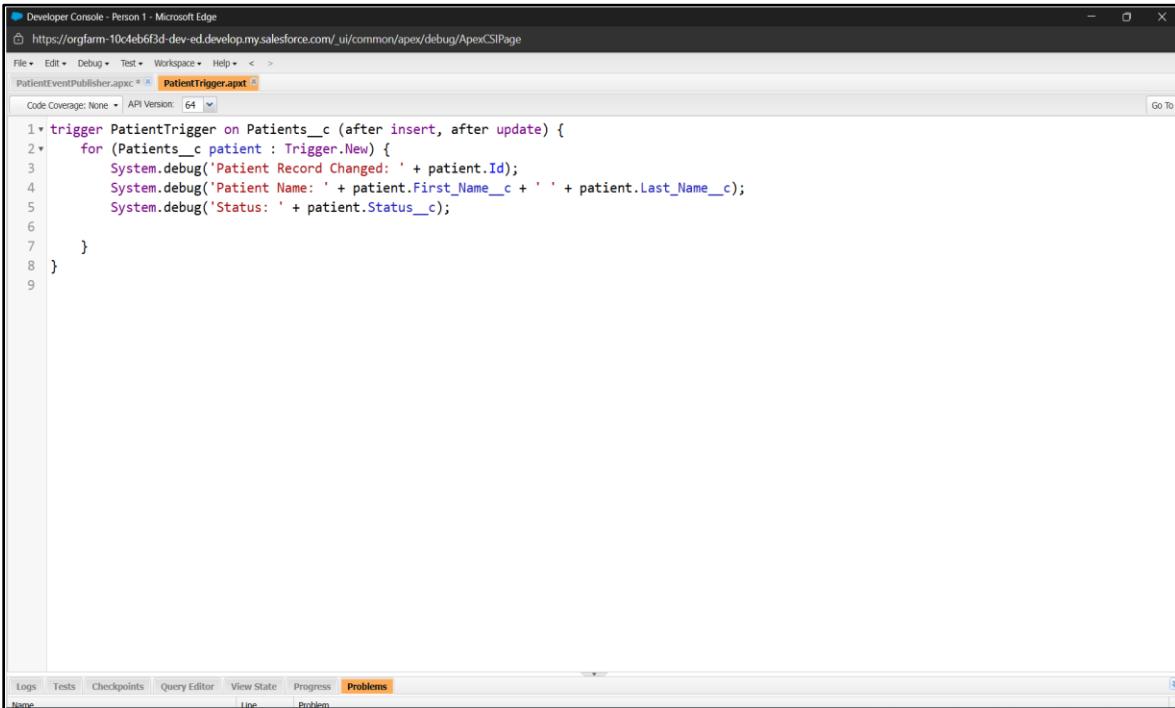


❖ Change Data Capture:

- Go to the Setup.
- In the Quick Find box, type the Change Data Capture and click it.
- From the list, select Patients_c, Appointment_c, Bed_Availability_c
- Click save.



- Now go to Develop Console.
- Click the File and select New Apex trigger.
- Name the trigger as PatientCDCTrigger and select sObject Patient_c
- Type the code and save it.



The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Person 1 - Microsoft Edge" and the URL is "https://orgfarm-10c4eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab bar has "PatientEventPublisher.apxc" and "PatientTrigger.apxt" selected. The code editor contains the following Apex trigger code:

```

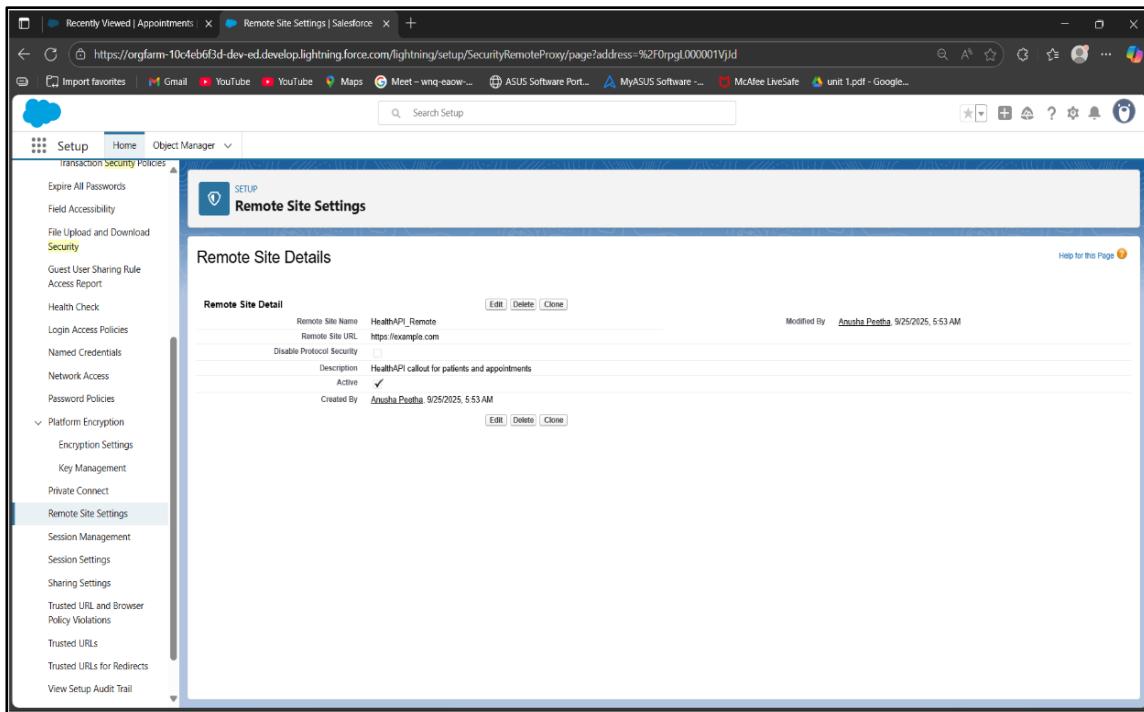
1 * trigger PatientTrigger on Patients__c (after insert, after update) {
2 *     for (Patients__c patient : Trigger.New) {
3 *         System.debug('Patient Record Changed: ' + patient.Id);
4 *         System.debug('Patient Name: ' + patient.First_Name__c + ' ' + patient.Last_Name__c);
5 *         System.debug('Status: ' + patient.Status__c);
6 *
7     }
8 }

```

The bottom navigation bar includes tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems, with the Problems tab currently selected.

❖ Remote site settings:

- Go to Setup.
- In the Quick Find box, type the Remote site settings and click it.
- Click on the New Remote Site.
- Fill the details:
 Remote site Name: HealthAPI_Remote
 Remote Site URL: <https://example.com/api>
 Description: HealthAPI callout for patients and appointments
 Enable Active
- Click Save.



- Open VS Code.
- Open the PatientAPIService and update the code.
- After updating right click on the File Name and select the SFDX: Deploy this source to Org

```

1  public with sharing class PatientAPIService {
2      @AuraEnabled(cacheable=false)
3      public static List<Patient__c> getPatientsFromAPI() {
4          List<Patient__c> patientList = new List<Patient__c>();
5          try {
6              HttpRequest req = new HttpRequest();
7              req.setEndpoint('https://example.com/api/patients');
8              req.setMethod('GET');
9              HttpResponse res = http.send(req);
10             System.debug('API Response: ' + res.getBody());
11             if(res.getStatusCode() == 200) {
12                 List<Object> records = (List<Object>) JSON.deserializeUntyped(res.getBody());
13                 for(Object rec : records) {
14                     Map<String,Object> r = (Map<String,Object>) rec;
15                     Patients__c p = new Patients__c();
16                     p.First_Name__c = (String) r.get('firstName');
17                     p.Last_Name__c = (String) r.get('lastName');
18                     p.Status__c = (String) r.get('status');
19                     patientList.add(p);
20                 }
21             }
22         } catch(Exception e) {
23             System.debug('Error calling API: ' + e.getMessage());
24         }
25     }
26     return patientList;
27 }
28 }
```

OUTPUT DEBUG CONSOLE TERMINAL PORTS HISTORY Filter Salesforce CLI

Changed PatientAPIService ApexClass force-app\main\default\classes\PatientAPIService.cls
Changed PatientAPIService ApexClass force-app\main\default\classes\PatientAPIService.cls-meta.xml

18:26:02.530 Ended SFDX: Deploy This Source to Org

Ln 25, Col 10 Spaces: 4 UTF-8 LF {} Apex

- Now click on the Debug and select the Execute Anonymous window.
- Type the code,

```
List<Patients__c> patients = PatientAPIService.getPatientsFromAPI();
System.debug(patients);
```

The screenshot shows the Developer Console in Microsoft Edge. The URL is https://orgfarm-104eb6f3d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The title bar says "Developer Console - Person 1 - Microsoft Edge". The tab bar has "PatientTrigger.apex" and "Log executeAnonymous @9/25/2025, 6:26:18 PM". The main area is titled "Execution Log" and shows a table of logs. The logs are timestamped from 18:26:18:359 to 18:26:18:000. Each log entry is a "USER_DEBUG" event with a timestamp, event type, and a large "Details" column containing the generated HTML code for a simple page with a title, meta tags, and a style block.

Timestamp	Event	Details
18:26:18:359	USER_DEBUG	[17][DEBUG] API Response: <!doctype html>
18:26:18:000	USER_DEBUG	<html>
18:26:18:000	USER_DEBUG	<head>
18:26:18:000	USER_DEBUG	<title>Example Domain</title>
18:26:18:000	USER_DEBUG	<meta charset="utf-8" />
18:26:18:000	USER_DEBUG	<meta http-equiv="Content-type" content="text/html; charset=utf-8" />
18:26:18:000	USER_DEBUG	<meta name="viewport" content="width=device-width, initial-scale=1" />
18:26:18:000	USER_DEBUG	<style type="text/css">
18:26:18:000	USER_DEBUG	body { background-color: #f0f0f2; }
18:26:18:000	USER_DEBUG	margin: 0;
18:26:18:000	USER_DEBUG	padding: 0;
18:26:18:000	USER_DEBUG	font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", Helvetica Neue, Arial, sans-serif;
18:26:18:000	USER_DEBUG	}
18:26:18:000	USER_DEBUG	div { width: 600px; }
18:26:18:000	USER_DEBUG	margin: 5em auto;
18:26:18:000	USER_DEBUG	padding: 2em;
18:26:18:000	USER_DEBUG	background-color: #fdfdff;
18:26:18:000	USER_DEBUG	border-radius: 0.5em;
18:26:18:000	USER_DEBUG	box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
18:26:18:000	USER_DEBUG	}
18:26:18:000	USER_DEBUG	a.link, a.visitrel {
18:26:18:000	USER_DEBUG	color: #38488E;
18:26:18:000	USER_DEBUG	text-decoration: none;
18:26:18:000	USER_DEBUG	}
18:26:18:000	USER_DEBUG	@media (max-width: 700px) {
18:26:18:000	USER_DEBUG	div {
18:26:18:000	USER_DEBUG	margin: 0 auto;

At the bottom, there are checkboxes for "This Frame", "Executable", "Debug Only", and "Filter", followed by a link "Click here to filter the log".

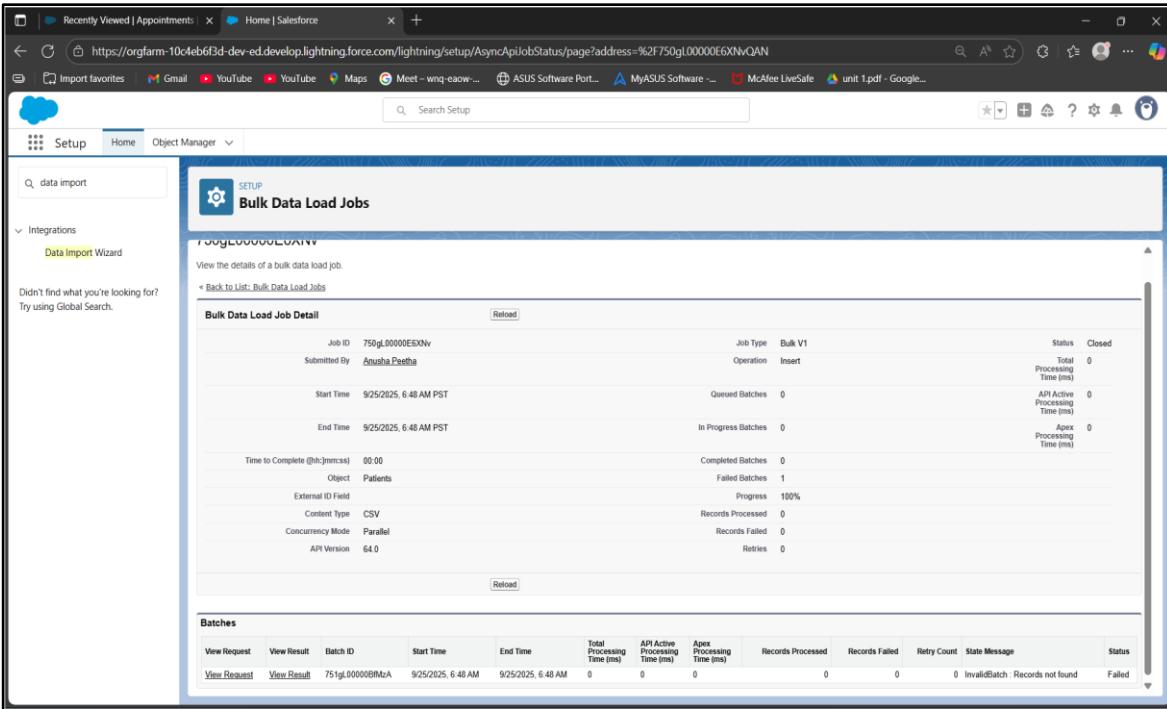
Salesforce Project Phase-8

Health Patient Management and Telemedicine CRM

Phase 8: Data Management & Deployment

❖ Data Import Wizard:

- Go to the Google sheets or Microsoft Excel.
- Create the patient csv file with the existing details
- Save the file.
- Now go to the Setup.
- In the Quick Find box, type the Data Import Wizard and click it.
- Click on the Launch Wizard.
- Select the Object called Patients.
- Click choose csv file
- Click Next.
- The Salesforce will auto-map csv columns if it shows unmapped click on the Map and select the required fields
- Click on save and then finish.



❖ Data Loader:

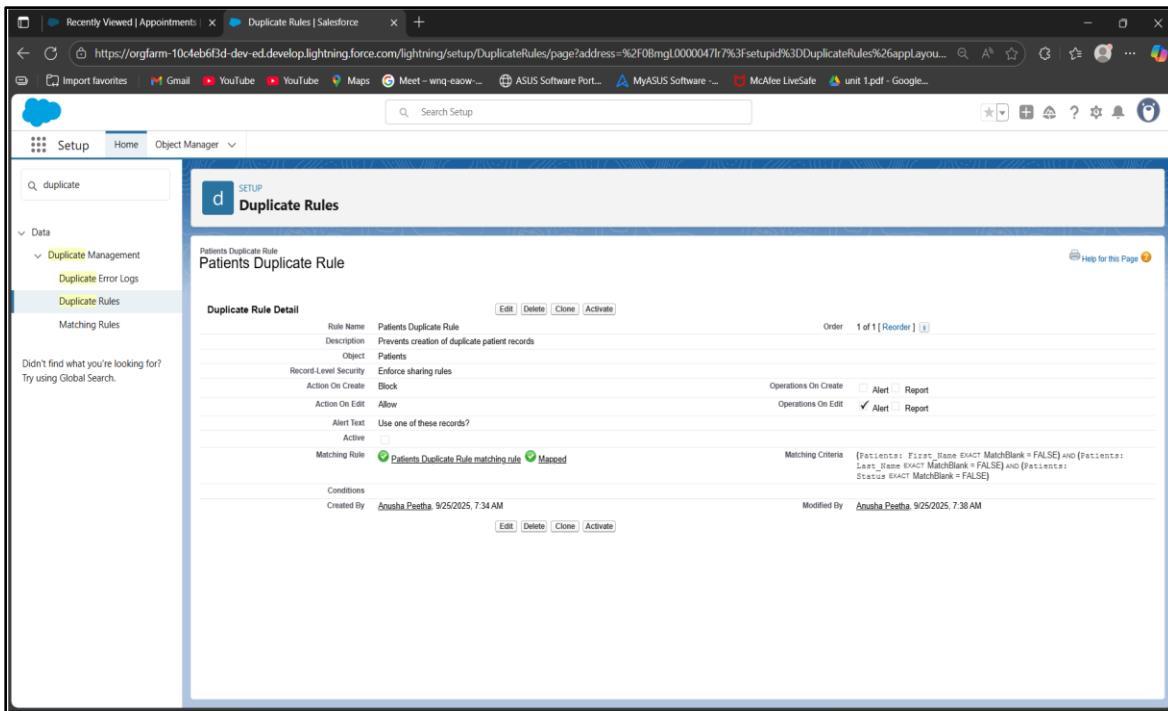
- Go to Setup.
- In the Quick Find box, type Data Loader and click it.
- Click on Downloads and click on Confirm
- Launch the install.bat
- Click on Allow
- In command prompt if asked yes/no type yes
- Choose operation insert and choose object patient
- Browse the existing CSV file
- Create on Create or Edit Map and then click ok
- Click Next and Finish

❖ Duplicate Rules:

- Go to the Setup.
- In the Quick Find box, type the Duplicate rules and click it.
- Click on New Rule
- Select the Patients Object
- Fill the details:
Rule Name: Patients Duplicate Rule
Description: Prevents creation of duplicate patient records
Enable the Enforce sharing rules
Select the Matching Rule, if don't have the matching rule click on the New Matching Rule.

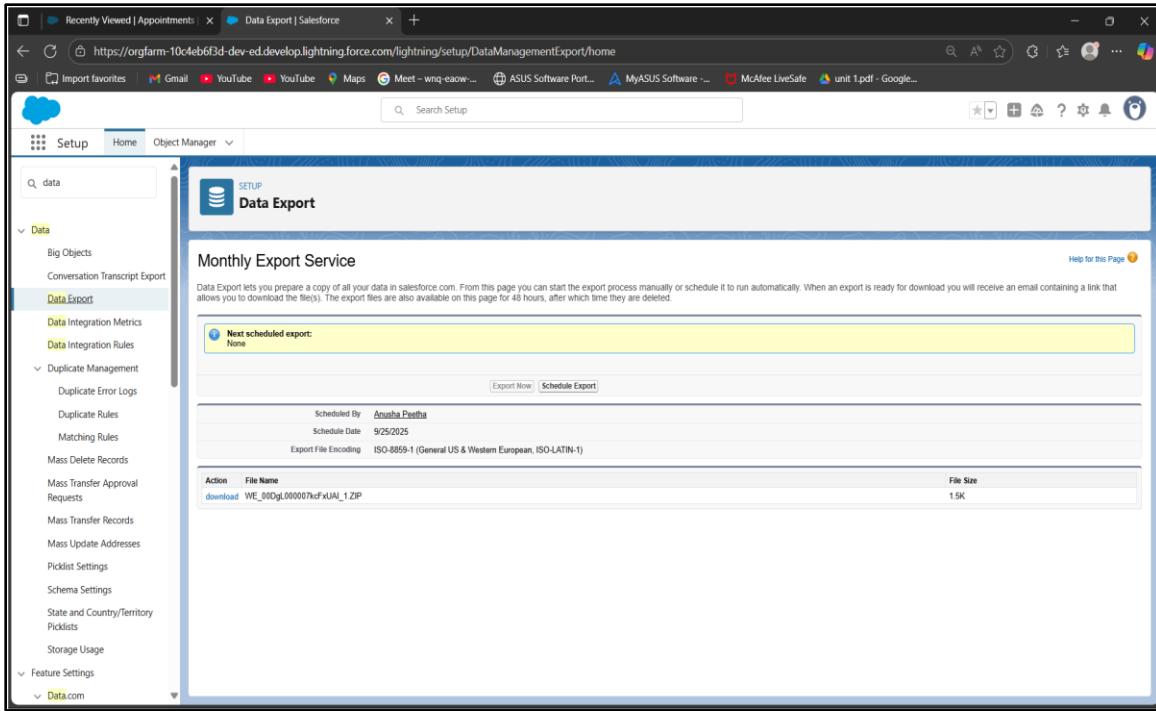
Define Matching Rule: Field: First_Name, Method: Exact
 Field: Last_Name, Method: Exact
 Field: Status, Method: Exact

- Click on save and activate.



❖ Data Export & Backup:

- Go to the Setup.
- In the Quick Find box, type the Data and click the Data Export.
- Click on the Export Now.
- Select the Patients and Appointments Object.
- Click on the Start Export.
- Now it will Scheduled to give it will notify through the Email.
- Click Download next to completed export.
- Extract the .zip file.
- Save the exported CSV files in a secure location.
- Now you have a backup of all patients and appointment records.



❖ VS Code & SFDX:

- Install the VS Code: <https://code.visualstudio.com/>
- Intall the Salesforce CLI: <https://developer.salesforce.com/tools/sfdxcli>
- In VS Code install the Salesforce Extension Pack in the Extensions
- Setup the project:
 - Open VS Code
 - Press the ctrl + shift + p and type SFDX: Create Project
 - Choose Standard Project
 - Enter the project name: Health
- Connect the Salesforce Org:
 - Press ctrl + shift + p and type SFDX: Authorize an Org
 - Choose Production
 - Log in to the Org in the browser
- Retrieve Metadata:
 - Press ctrl + shift +p and type SFDX: Retrieve source from Org
- Deploy Metadata to Org:
 - Press ctrl + shift +p and type SFDX: Deploy this source to Org

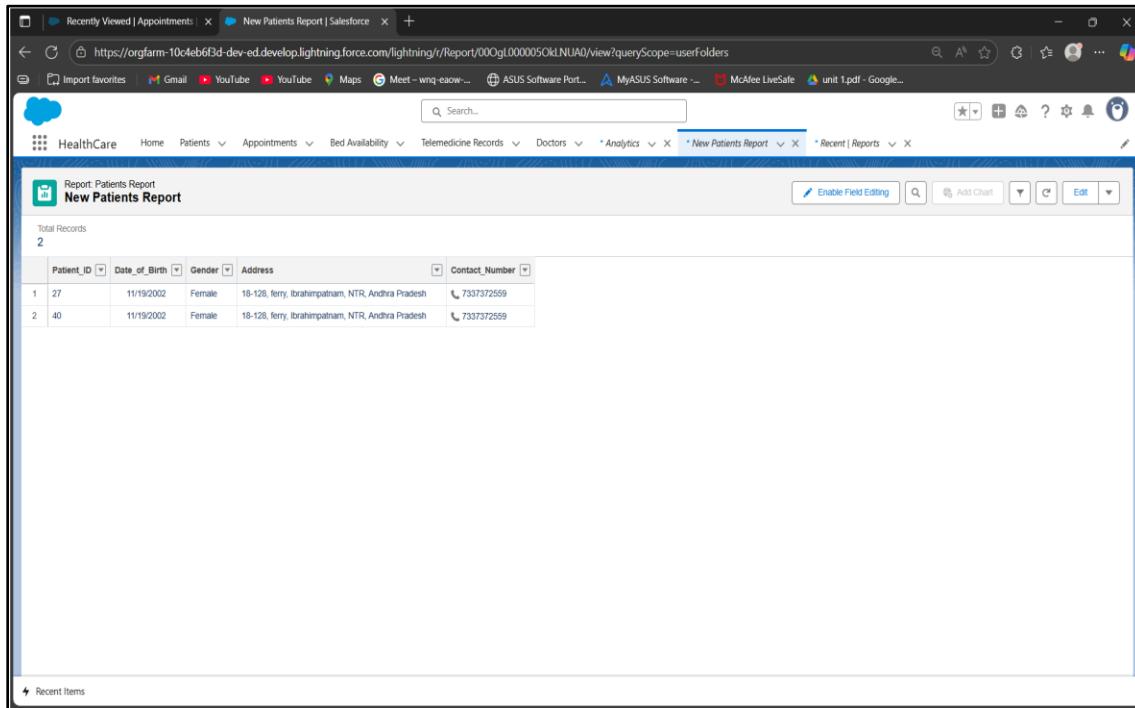
Salesforce Project Phase-9

Health Patient Management and Telemedicine CRM

Phase 9: Reporting, Dashboards & Security Review

❖ Reports (Tabular, Summary, Matrix, Joined):

- Go to Setup.
- In the App Launcher type the Reports and click it.
- Click on create.
- Select the Patient Object.
- In the Outline, select the required rows and columns
- In the filter, give the required conditions
- Click on Save & Run

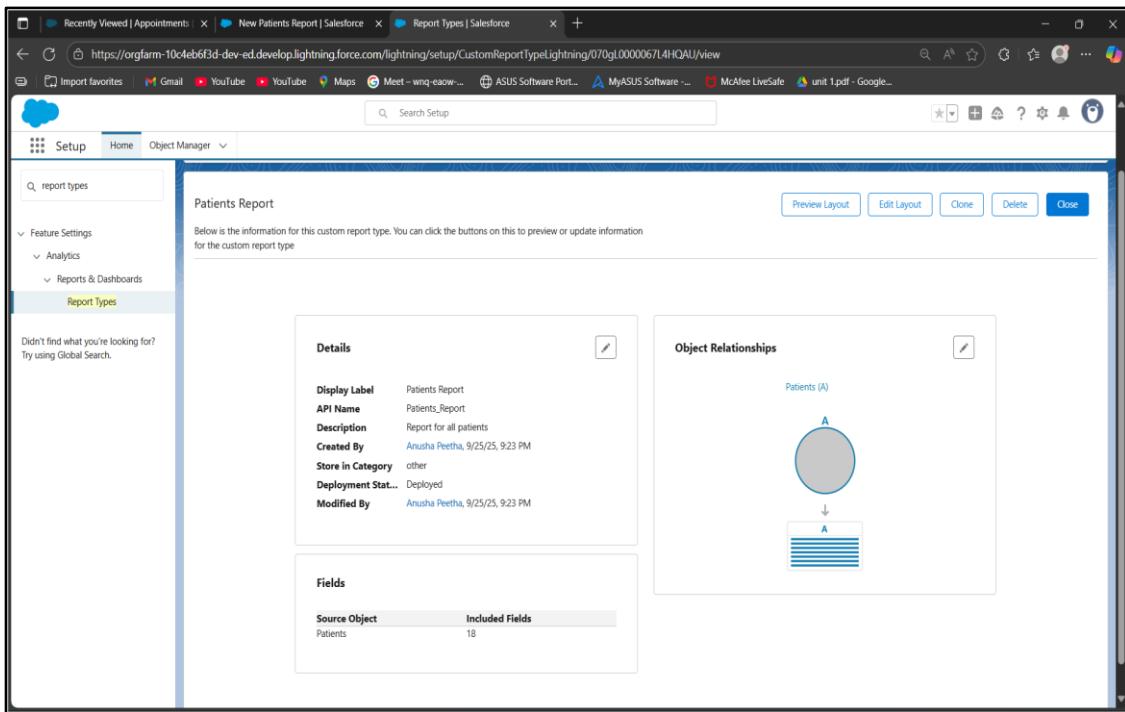


The screenshot shows a Salesforce report titled "Report: Patients Report" and "New Patients Report". The report displays two patient records in a tabular format. The columns are: Patient_ID, Date_of_Birth, Gender, Address, and Contact_Number. The data is as follows:

	Patient_ID	Date_of_Birth	Gender	Address	Contact_Number
1	27	11/19/2002	Female	18-128, Ferry, Ibrahimpatnam, NTR, Andhra Pradesh	7337372559
2	40	11/19/2002	Female	18-128, Ferry, Ibrahimpatnam, NTR, Andhra Pradesh	7337372559

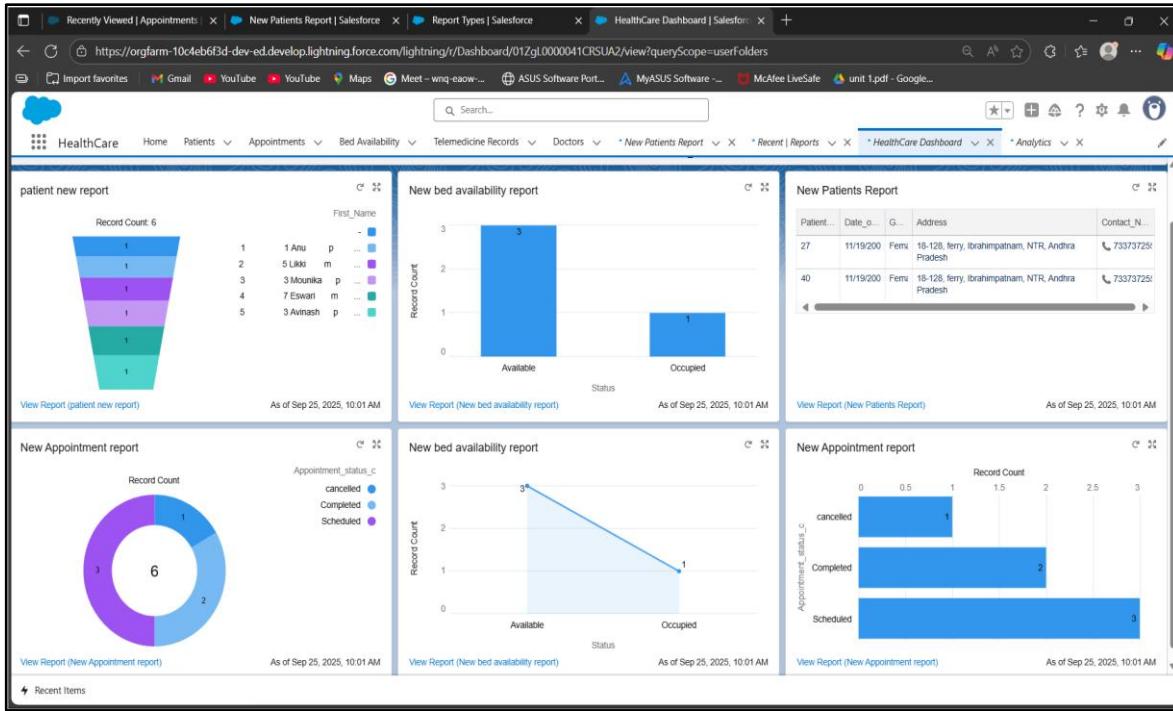
❖ Report Types:

- Go to Setup.
- In the Quick Find box, type the Report Types and click it.
- Click on New Custom Report Type.
- Primary Object: Patients
- Report Type Label: Patients Report
- Click to relate another object Appointments
- Choose relationship type, each patient may or may not have related appointments
- Click Save



❖ Dashboards:

- Go to Setup
- Click on App Launcher, type the Dashboards and click it
- Click on New Dashboard
- Name it as HealthCare Dashboard
- Choose the Private folder and click Next
- Click on +Widget and select the required reports
- Now arrange them in required format
- Click on Save and then Done.



❖ Dynamic Dashboards:

- Go to Setup
- In the Quick Find box, type the Dashboards and click it
- Click on New Dashboard
- Name it as HealthCare dynamic dashboard
- Choose the Public folder and Click Next
- Click on +Widget and select required reports and click on Add
- In the top right click on Gear icon and select the View dashboard as The dashboard Viewer and Save.
- Click Save and Done.

❖ Session Settings:

- Go to Setup
- In the Quick Find box, type the security and then select the Session Settings
- Check:
Timeout Value: 2 hours
Enable force logout on Session Timeout
- Save changes

❖ Login IP Ranges:

- Go to Setup
- In the Quick Find box, type the Profiles and them Select the Doctor profile

- Scroll down to see Login IP Ranges and click on New
 - Fill the details:
 - StartIp: 49.37.130.127
 - EndIp: 49.37.130.127
 - Click on Save
- ❖ **Audit Trail:**
- Go to Setup
 - In the Quick Find box, type the Security and then select the view Setup Audit Trail
 - You can see last 180 days of changes

Salesforce Project Phase-10

Health Patient Management and Telemedicine CRM

Phase 10: Final Presentation & Demo Day

❖ Pitch Presentation:

- **Problem:** Inefficient tracking of patients, appointments, and bed availability using disconnected spreadsheets or manual records, leading to errors and delayed decision-making.
- **Solution:** A centralized HealthCare Management System on Salesforce to manage patients, appointments, bed availability, and analytics in one platform.
- **Benefits:**
 - Streamlined patient and appointment management
 - Accurate bed tracking and allocation
 - Quick search and navigation with Lightning Web Components
 - Data-driven insights via dynamic dashboards and reports
 - API integration for external patient data

❖ Demo Walkthrough:

- Start on the Home Page Dashboard to get an overview of patient status, appointments, and bed availability.
- Open a Patient record and demonstrate the Patient Quick Search LWC.
- Navigate to Appointment records: create a new appointment, update its status, and show the workflow in action.
- Navigate to Bed Availability records: update bed status and demonstrate tracking of occupied vs. available beds.
- Show Reports & Charts:
 - Pie chart for bed occupancy
 - Bar chart for daily appointments
 - Summary of patient statuses

- Demonstrate API Callouts using PatientAPIService to fetch external patient data.

❖ **Handoff Documentation:**

User Guide:

- How to create a new patient record
- How to schedule or update appointments
- How to track bed availability
- How to use quick search and navigation features

Admin Guide:

- Overview of custom objects: Patients__c, Appointment__c, Bed_Availability__c
- Lightning Web Components used
- Apex classes for API integration and imperative calls
- Reports and dashboards setup
- Sharing settings, field-level security, and login IP ranges

❖ **Portfolio Project Showcase:**

Professional summary for LinkedIn/Portfolio:

- Title: Health Patient Management and Telemedicine CRM
- Description:
“Designed, developed, and deployed a custom HealthCare Management System on Salesforce to efficiently manage patients, appointments, and bed availability. The system provides real-time analytics through dashboards, enables quick searches with LWC components, integrates external patient data via API callouts, and ensures accurate reporting for hospital operations.”
- Skills: Salesforce Administration, Apex, Lightning Web Components (LWC), SOQL, Flows & Process Automation, Data Modeling, API Integration, Reports & Dashboards
- Link: <https://github.com/anu379/Healthcare-Patient-Management-and-Telemedicine-CRM>