

Medical Inventory Management

College Name: Sri Ramalinga sowdambigai college of science
and commerce

College Code: Bru3y

TEAM ID: NM2025TMID25514

TEAM MEMBERS: 4

Team Leader Name: Chandru S

Email: Dhineshchandru8656@gmail.com

Team Member1: Boopathi C

Email: Bboopathixo@gmail.com

TeamMember: Pranav M

Email: Pranavpranavm10@gmail.com

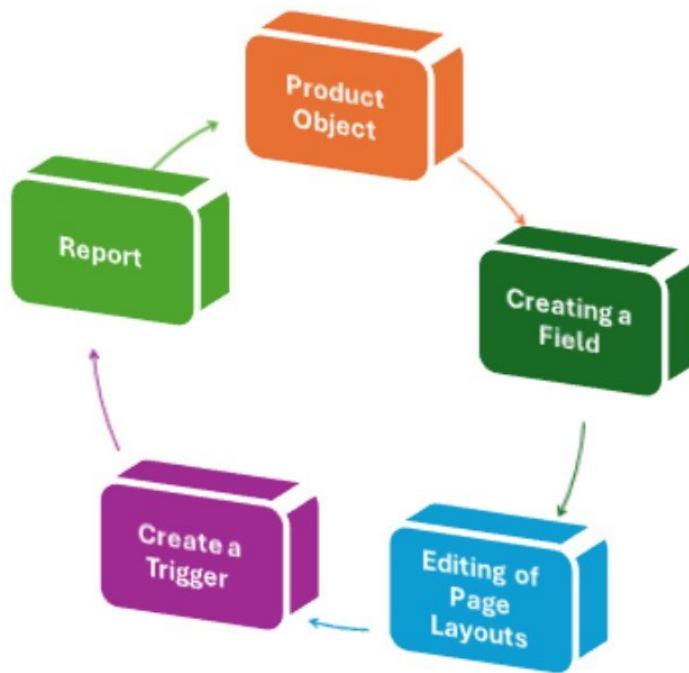
TeamMember: Hari Prasath T

Email: Mrrxveriyan950@gmail.com

1.INTRODUCTION

1.1 Project Overview

The Medical Management System is a Salesforce-based application developed under the Naan Mudhalvan scheme to streamline hospital and clinic operations. It manages patients, doctors, appointments, prescriptions, billing, and medical records in a centralized cloud-based platform. The system ensures quick access to patient history, automated appointment reminders, and secure communication between patients and healthcare providers.



1.2 Purpose

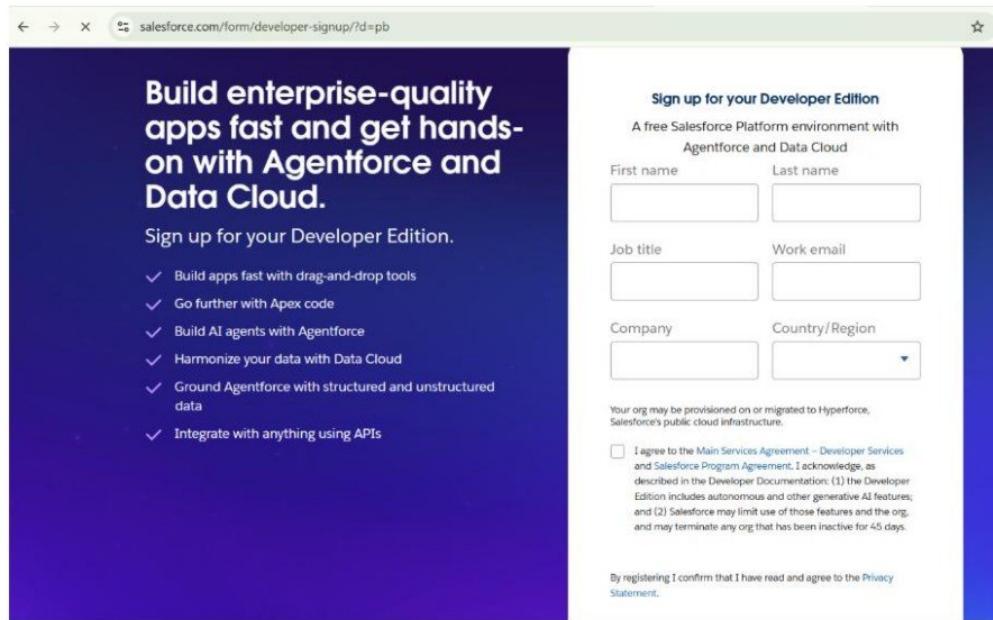
The main objective of the project is to create a centralized medical management solution that ensures efficiency, accuracy, and transparency in healthcare. The purpose is:

- To simplify patient registration and reduce duplicate records.
 - To enable online and automated appointment booking with doctors.
 - To send reminders and follow-up notifications to patients.
 - To maintain accurate electronic health records (EHRs).
 - To automate billing and approval workflows.
- To improve doctor–patient communication using Salesforce’s email and notification system.

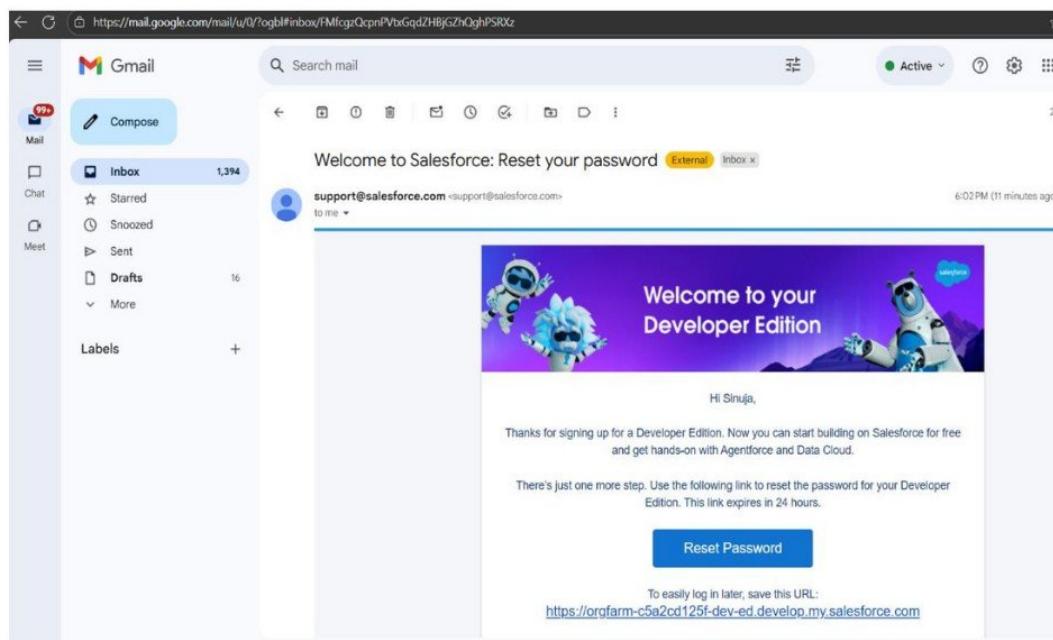
2.DEVELOPMENT PHASE

2.1 Creating Developer Account:

- Registered on Salesforce Developer Edition using <https://www.salesforce.com/form/developer-signup/?d=pb>
- Configured the org with profiles, permission sets, and security settings.



2.2 Account Activation:



2.3 Verify Account

https://orgfarm-c5a2cd125f-dev-ed.my.salesforce.com/_ui/system/security/ChangePassword?retURL=%2Fhome%2Fhome.jsp&fromFrontdoor=1&setupid=ChangePassword

Change Your Password

Enter a new password for
23bsit156sinujas857@agentforce.com. Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

* New Password

* Confirm New Password

* Security Question

In what city were you born?

* Answer

*=required

Change Password

Password was last changed on 9/17/2025, 5:32 AM.

2.4 Salesforce Setup Page

https://orgfarm-c5a2cd125f-dev-ed.lightning.force.com/lightning/setup/SetupOneHome/home

The screenshot shows the Salesforce Setup Home page. The left sidebar contains a navigation menu with sections like Setup Home, Administration, and Platform Tools. The main content area displays three setup cards: Data Cloud, Get Started with Einstein Bots, and Mobile Publisher. Each card has a 'Watch Video' or 'Get Started' button. Below the cards is a section titled 'Most Recently Used' which is currently empty. The top of the page includes the Salesforce logo, a search bar, and various browser icons.

2.5 Objects Created: Product

New Custom Object

Custom Object Definition Edit

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.

Label: Example: Account
Plural Label: Example: Accounts
Starts with vowel sound:

The Object Name is used when referencing the object via the API.

Object Name: Example: Account

Description:

Context: Sensitive Help Setting: Open the standard Salesforce.com Help & Training window Open a window using a Visualforce page

Content Name:

Enter Record Name Label and Format

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name: Example: Account Name

Data Type: Warning: If you plan to insert a high volume of records in this object, via the API for example, use the Text data type.

Optional Features

- Allow Reports
- Allow Activities
- Track Field History
- Allow in Chatter Groups
- Enable Licensing

Object Classification

When these settings are enabled, this object is classified as an Enterprise Application object. When these settings are disabled, this object is classified as a Light Application object. Learn more.

- Allow Sharing
- Allow Bulk API Access
- Allow Streaming API Access

Deployment Status

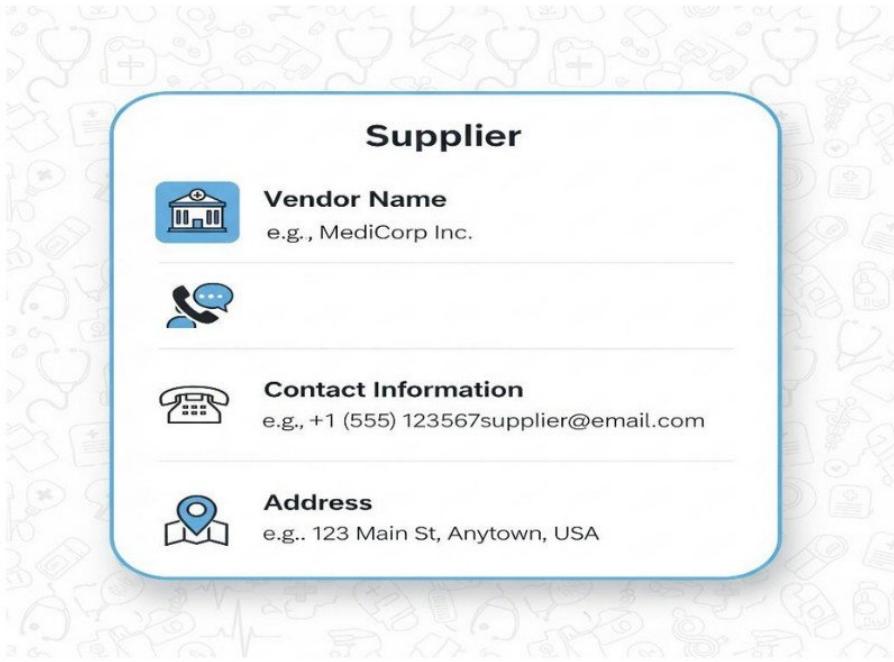
In Development Deployed [What is this?](#)

Search Status

When this setting is enabled, your users can find records of this object type when they search. Learn more.

Allow Search

- Supplier: Contains information about vendors who supply the medicines.



- Purchase Order: Tracks orders placed with suppliers, linking them to specific medicines and quantities.

New Custom Object

Custom Object Definition Edit

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.

Labeled	<input type="text" value="Purchase Order"/>	Example: Account
Plural Label	<input type="text" value="Purchase Orders"/>	Example: Accounts
Starts with vowel sound	<input type="checkbox"/>	

The Object Name is used when referencing the object via the API.

Object Name	<input type="text" value="Purchase_Order"/>	Example: Account
-------------	---	------------------

Description

Context-Sensitive Help Setting

Open the standard Salesforce.com Help & Training window
 Open a window using a Visualforce page

Content Name

2.3 Configurations Done

A series of configurations were implemented to automate processes and enforce business rules:

Creating Remaining Tabs

Procedure:

-Create tabs for the following objects:

-Purchase Order

-Order Item

-Inventory Transaction

-Supplier

Follow the same steps as described **Activity 1 (Creating a Tab for the Product Object)** to complete the process for each object.



The Lightning App

Activity 1: Creating a Lightning App for Medical Inventory Management

Procedure:

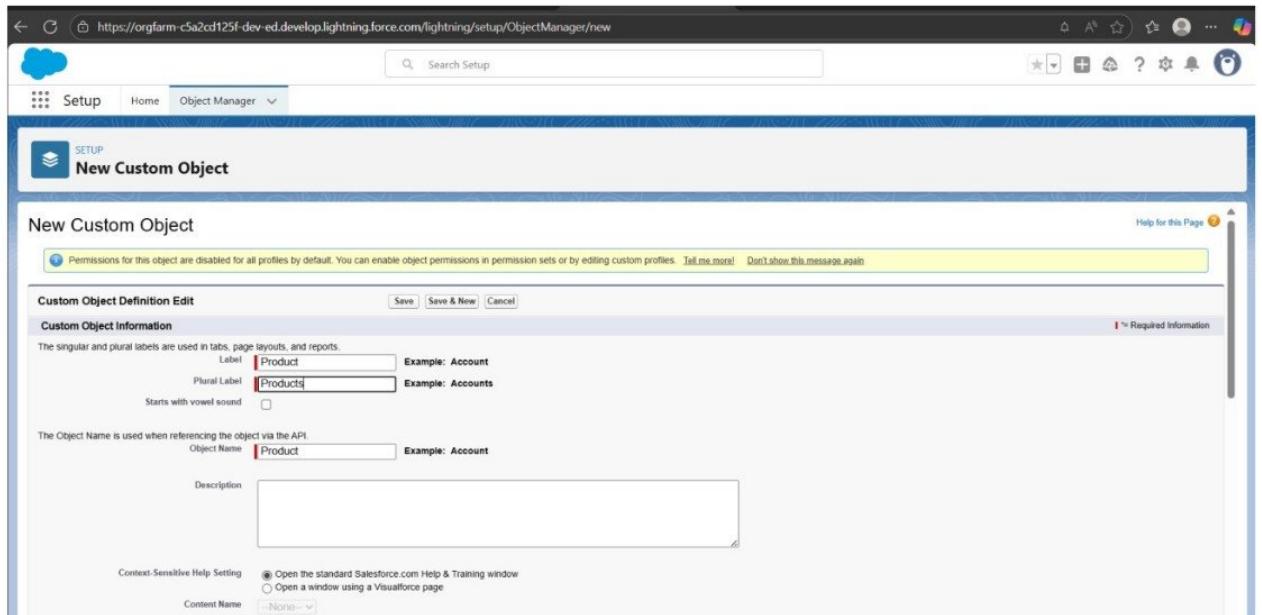
1. From **Setup**, enter **App Manager** in the Quick Find bar and select **App Manager**.
2. Click **New Lightning App**.
3. Enter **Medical Inventory Management** as the **App Name**.
 - Optionally, upload an image related to medical inventory.
 - Click **Next**.
4. Under **App Options**, leave the default selections and click **Next**.
5. Under **Utility Items**, retain the default configuration and click **Next**.

Fields

Creating a Text Field in the Product Object

Steps:

- 1.
- 2.
3. Click the **gear icon** and select **Setup** (opens in a new tab).
4. In Setup, go to the **Object Manager** tab.
5. Select the **Product** custom object.
6. From the left navigation, click **Fields & Relationships**.
7. Click **New**.
Choose **Text** as the field type and click **Next**.
Enter the following details:
 - o **Field Label:** Product Name
 - o **Length:** 255
8. Select the **Required Field** checkbox.
9. Click **Next → Next → Save & New** to create the field.



Creating a Number Field in the Product Object

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Product and select the Product custom object.
- From the left panel, click Fields & Relationships.
- Click New.

-ChooseNumber as the data type and click Next.

Enter the details:

Field Label: Current Stock Level

Length: 18

Decimal Places: 0

Click Next → Next → Save to finish creating the field.

Creating a Currency Field in the Product Object Steps:

- Go to Setup → click on Object Manager.
 - In the Quick Find box, type Product and select the Product custom object.
 - From the left-hand menu, select Fields & Relationships.
 - Click New.
 - Choose Currency as the data type and click Next.
- Enter the details:** Field

Label: Unit Price Length: 16

Decimal Places: 2

Mark the field as Required.

Click Next → Next → Save.

The screenshot shows the Salesforce Setup interface for creating a new custom field. The left sidebar is titled 'FIELDS & RELATIONSHIPS' and includes options like 'Page Layouts', 'Lightning Record Pages', 'Buttons, Links, and Actions', 'Compact Layouts', 'Field Sets', 'Object Limits', 'Record Types', 'Related Lookup Filters', 'Search Layouts', 'List View Button Layout', 'Restriction Rules', and 'Scoping Rules'. The main area is titled 'Product New Custom Field' and 'Step 2. Enter the details'. It shows the following configuration:

- Field Label:** Unit Price
- Length:** 16
- Decimal Places:** 2
- Field Name:** Unit_Price
- Description:** (empty)
- Help Text:** (empty)
- Required:** Always require a value in this field in order to save a record
- Auto add to custom report type:** Add this field to existing custom report types that contain this entity
- Default Value:** Show Formula Editor

Creating a Lookup Relationship in the Purchase Order Object

A Lookup Relationship in Salesforce links two objects together, where one object (child) references another (parent). This helps maintain relational data integrity and allows easy navigation between related records.

In this activity, we'll establish a relationship from Purchase Order (child) to Supplier (parent).

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Purchase Order and select the Purchase Order custom object.
- From the left-hand menu, click Fields & Relationships.
- Click New.
- Select Lookup Relationship as the data type and click Next.
- For the related object, select Supplier.
- Click Next.

Enter the details:

Field Label: Supplier ID

Mark the field as Required.

Continue by clicking Next → Next → Next → Save.

The screenshot shows the Salesforce Setup interface under the Object Manager section for the Purchase Order object. A new relationship is being created, specifically for 'Supplier objects'. The 'Field Label' is set to 'Supplier objects', and the 'Field Name' is 'Supplier_objects'. Under 'Required', the 'Always require a value in this field in order to save a record' checkbox is checked. In the 'What to do if the lookup record is deleted?' section, the 'Don't allow deletion of the lookup record that's part of a lookup relationship' option is selected. The 'Help Text' and 'Description' fields are empty. The 'Child Relationship Name' is 'Purchase_Orders'. At the bottom, there are options to 'Auto add to custom report type' and 'Add this field to existing custom report types that contain this entity'. The status bar at the bottom right indicates 'Step 3 of 6'.

Creating a Date Field in the Purchase Order Object

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Purchase Order and select the Purchase Order custom object.
- From the left-hand menu, click Fields & Relationships.
- Click New.
- Select Date as the data type and click Next.
- Enter the following details:
- Field Label: Order Date
- Click Next → Next → Save to complete the creation of the date field.

The screenshot shows the Salesforce Setup interface under the Object Manager section for the Purchase Order object. A new custom field is being created, labeled 'Order Date'. The 'Field Label' is 'Order Date', and the 'Field Name' is 'Order_Date'. Under 'Required', the 'Always require a value in this field in order to save a record' checkbox is unchecked, while the 'Add this field to existing custom report types that contain this entity' checkbox is checked. The 'Default Value' section includes a 'Show Formula Editor' button. A note at the bottom explains the use of formula syntax: 'Use formula syntax: #Fields, list and plural names API names in dollar quotes ("Name", "list"), include numbers or field codes (#1, #list) and always use decimal (.00), and express date calculations in the standard format (today) + 7, To reference a field from a Custom Metadata type record use: \$CustomMetadataType__mdRecordName.FieldName_c'.

Creating a Roll-Up Summary Field in Purchase Order object To create fields in an object:

1. Goto setup>>click on Object Manager>>type object name(PurchaseOrder)

- in quick find box>> click on the Purchase Order object.
2. Now click on “Fields & Relationships”
 3. Click on New.
 4. Select Data type as “Roll-Up Summary” and click Next.
 5. Enter Field Label as “ Order Count”.
 6. Choose the Summarized Object as “Order Items”.
 7. For Select Roll-Up Type select “Count”.
 8. Click on Next, Next and Save

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedBy	Lookup(User)		
Last Modified By	LastModifiedBy	Lookup(User)		
Order Date	Order_Date__c	Date		
Order_Count	Order_Count__c	Master-Detail(Order Item)	✓	▼
Purchase Order ID	Name	Text(90)	✓	▼
Supplier ID	Supplier_ID__c	Lookup(Supplier)	✓	▼

Creating a Unit Price Formula Field in the Order Item Object

Steps:

- Go to **Setup** → click on **Object Manager**.
- In the Quick Find box, type **OrderItem** and select the **Order Item** custom object.
- From the left-hand menu, click **Fields&Relationships**.
- Click **New**.
- Select **Formula** as the data type and click **Next**. Enter
the following details:

Field Label: Unit Price

Formula ReturnType: Currency

In the formula editor, enter the advanced formula:

`Product_ID__r.Unit_Price__c`

This pulls the **Unit Price** directly from the related **Product** object. Click **Next** → **Next** → **Save** to complete the field creation

Creating an Amount Formula Field in the Order Item Object

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Order Item and select the Order Item custom object.
- From the left-hand menu, click Fields & Relationships.
- Click New.
- Select Formula as the data type and click Next.

Enter the following details:

Field Label: Amount

Formula Return Type: Currency

In the formula editor, enter the advanced formula:

`Quantity_Received__c * Unit_Price__c`

This calculates the total price for each Order Item automatically.

Click Next → Next → Save to complete the field creation.

Creating a Picklist Field in the Inventory Transaction Object

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Inventory Transaction and select the object.

-From the left-hand menu, click Fields & Relationships.

-Click New.

-Select Picklist as the data type and click Next.

Enter the following details:

Field Label: Transaction Type

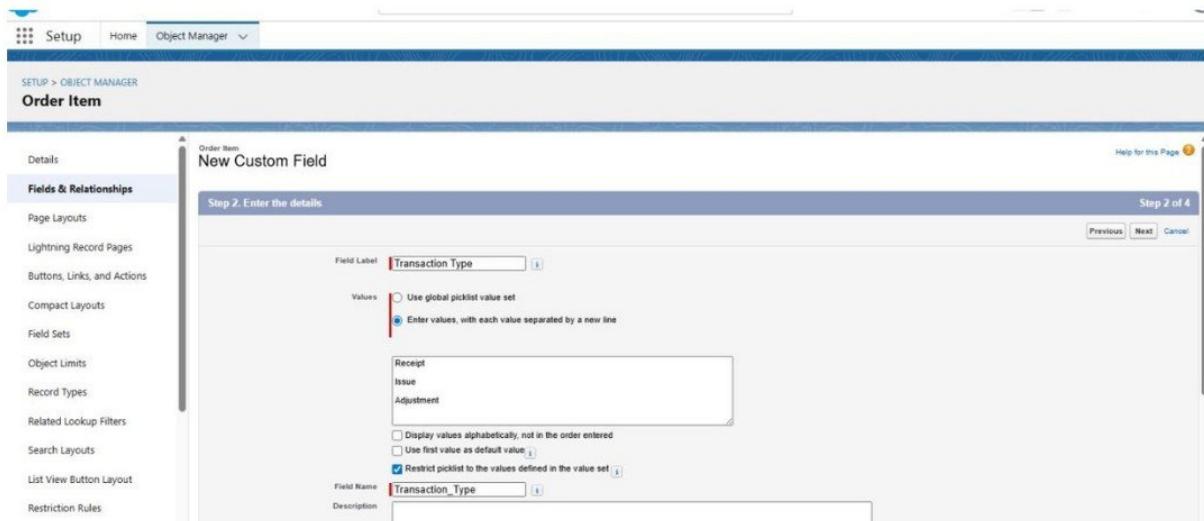
Values: Enter manually, each on a new line:

Receipt

Issue

Adjustment

Click Next → Next → Save to complete the picklist creation.



Creating a Total Order Cost Formula Field in the Inventory Transaction Object

Steps:

-Go to Setup → click on Object Manager.

-In the Quick Find box, type Inventory Transaction and select the object.

-From the left-hand menu, click Fields & Relationships.

-Click New.

-Select Formula as the data type and click Next.

-Enter the following details:

Field Label: Total Order Cost

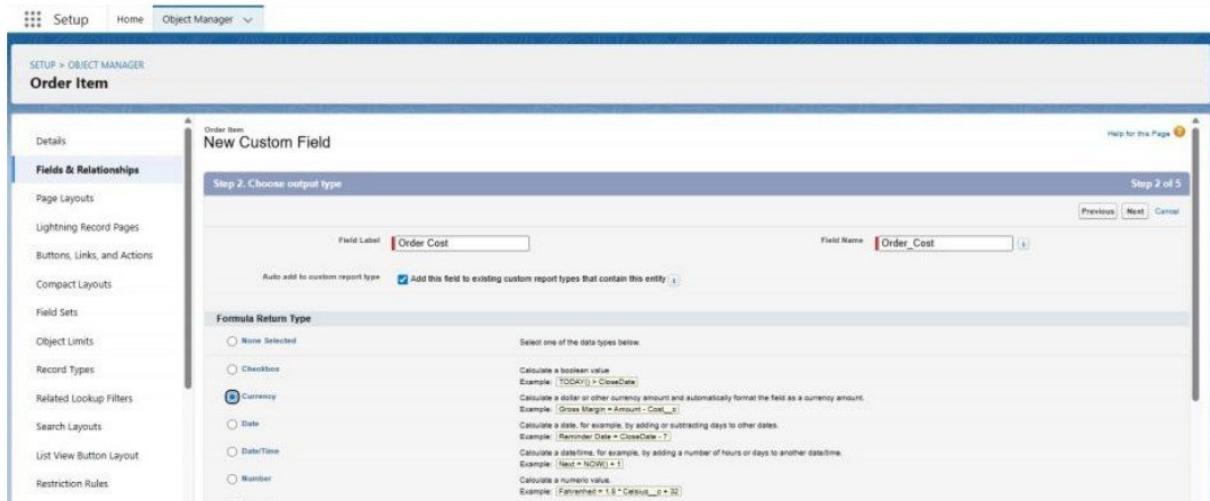
Formula Return Type: Currency

-In the formula editor, enter the advanced formula:

Purchase_Order_ID_r.Total_Order_Cost c

-This formula pulls the total cost from the related Purchase Order, ensuring accurate cost tracking for inventory transactions.

-Click Next → Next → Save to complete the field creation.



Creating a Phone Field in the Supplier Object

Steps:

-Go to Setup → click on Object Manager.

-In the Quick Find box, type Supplier and select the Supplier custom object.

-From the left-hand menu, click Fields & Relationships.

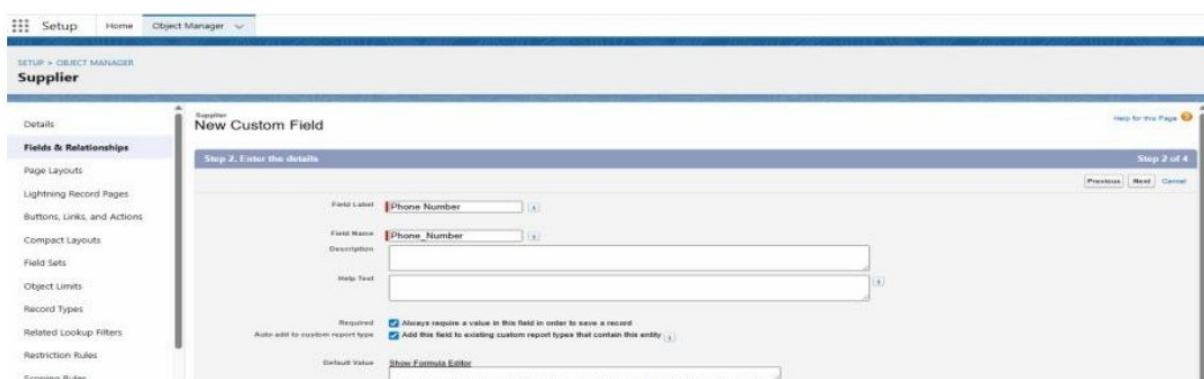
-Click New.

-Select Phone as the data type and click Next.

Enter the following details:

Field Label: Phone Number -Mark the field as Required. -

Click Next → Next → Save to complete the field creation.



Creating an Email Field in the Supplier Object

Steps:

Go to **Setup** → click on **Object Manager**.

In the Quick Find box, type **Supplier** and select the **Supplier** custom object. From the left-hand menu, click **Fields & Relationships**.

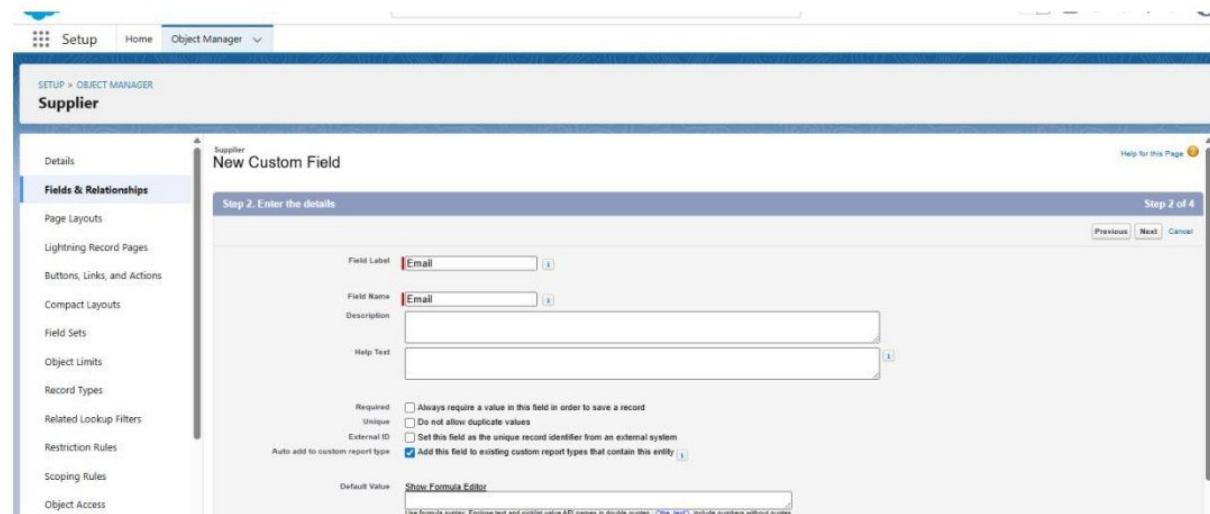
Click **New**.

Select **Email** as the data type and click **Next**.

Enter the following details:

Field Label: Email

Click **Next** → **Next** → **Save** to complete the field creation.



Page Layout Customization

Editing a Page Layout in the Product Object

Steps:

- Go to Setup → click on Object Manager.

- In the Quick Find box, type Product and select the Product custom object.

- From the left-hand menu, click Page Layouts.

- Select the layout named Product Layout.

- Drag and arrange the fields on the page layout as required to optimize data entry and display.

- Save it

Editing a Page Layout in the Purchase Order Object

Steps:

- Go to Setup → click on Object Manager. -In the Quick Find box, type Purchase Order and select the Purchase Order custom object.
- From the left-hand menu, click Page Layouts.
- Select the layout named Purchase Order Layout.
- Drag and arrange the fields on the layout as required to optimize data entry and display.

For the Order Datefield:

- Click on the field → click Settings → select Required → save.
- For the Total Order Cost field:
- Click on the field → click Settings → select Read-Only → save.
- Click Save to finalize the layout changes.

Editing a Page Layout in the Order Item Object

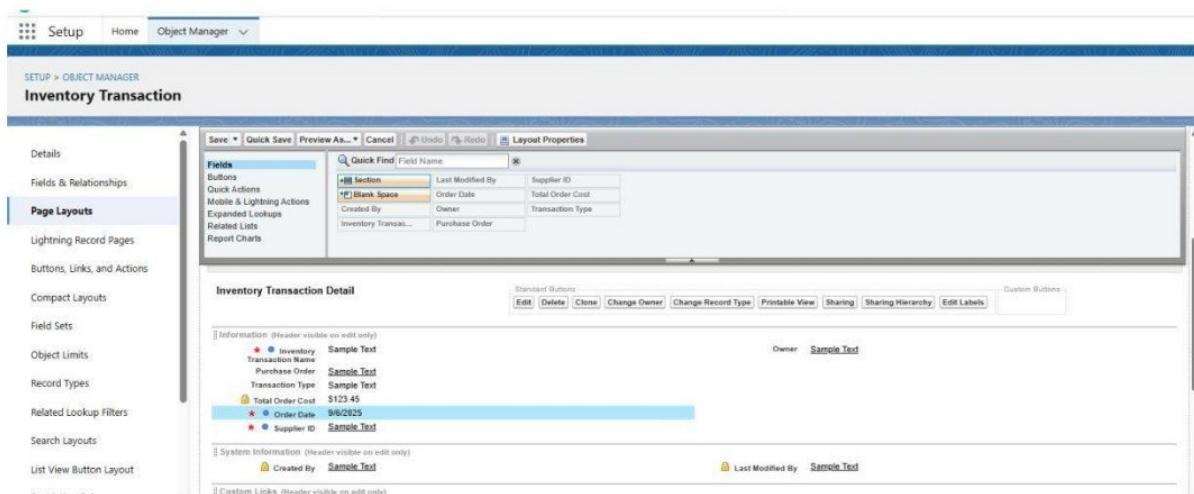
Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Order Item and select the Order Item custom object.
- From the left-hand menu, click Page Layouts.
- Select the layout named Order Item Layout.
- Drag and arrange the fields on the layout as required to optimize data entry and display.
- Click Save to finalize the layout changes.

Editing a Page Layout in the Inventory Transaction Object

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Inventory Transaction and select the Inventory Transaction custom object.
- From the left-hand menu, click Page Layouts.
- Select the layout named Inventory Transaction Layout.
- Drag and arrange the fields on the layout as required to optimize data entry and display.
- Click Save to finalize the layout changes.



Editing a Page Layout in the Supplier Object

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Supplier and select the Supplier custom object.
- From the left-hand menu, click Page Layouts.
- Select the layout named Supplier Layout.
- Drag and arrange the fields on the layout as required to optimize data entry and display.
- Click Save to finalize the layout changes.

Compact Layouts

Creating a Compact Layout for the Product Object

Steps:

- Go to Setup → click on Object Manager.
- In the Quick Find box, type Product and select the Product custom object.
- From the sidebar, click Compact Layouts.
- Click New.
- Enter the following details:

Label: Product Compact Layout

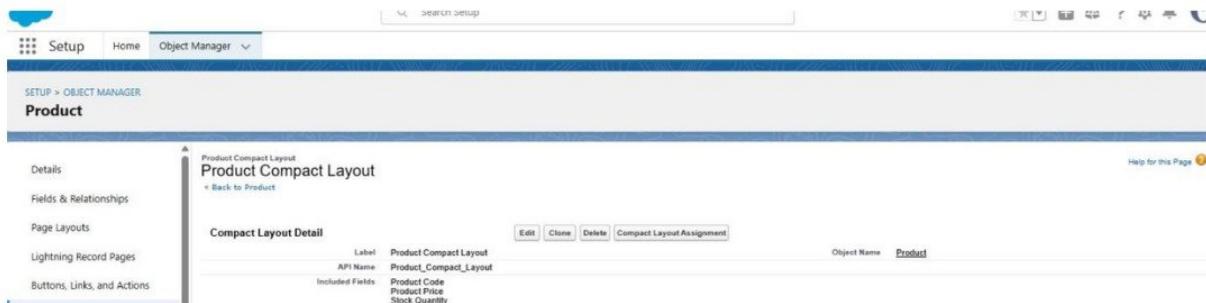
- Select the fields to display in the compact layout:

Product Name

Unit Price

Current Stock Level

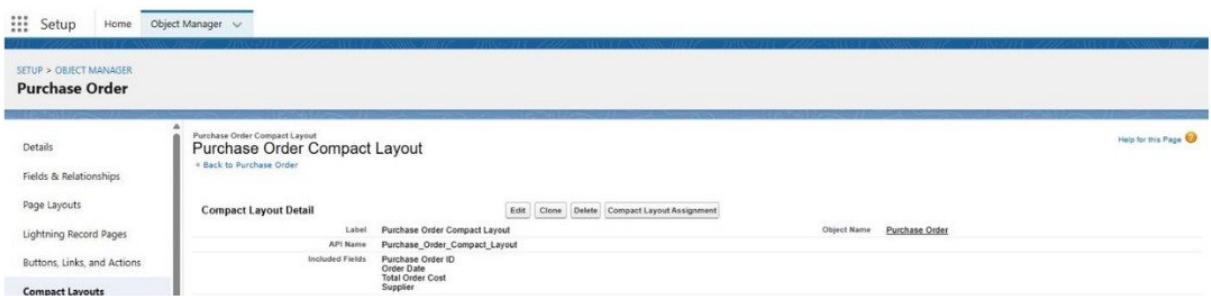
- Click Save.
- Click Compact Layout Assignment.
- Click Edit Assignment.
- Choose Product Compact Layout from the dropdown and click Save



Creating a Compact Layout for the Purchase Order Object

Steps:

1. 2.
3. Go to **Setup** → click on **Object Manager**.
4. In the Quick Find box, type **Purchase Order** and select the **Purchase Order** custom object.
5. From the sidebar, click **Compact Layouts**.
6. Click **New**.
- Enter the following details:
 - o **Label:** Purchase Order Compact Layout
- Select the fields to display in the compact layout:
 - o Purchase Order ID
 - o Order Date
 - o Total Order Cost
 - o Supplier ID
7. Click **Save**.
8. Click **Compact Layout Assignment** → **Edit Assignment**.
9. Choose **Purchase Order Compact Layout** from the dropdown.
10. Click **Save**.



Validation Rules

Activity 1: Creating an Expected Delivery Date Validation Rule for the Purchase Order Object

Steps:

1. 2.
- Go to **Setup** → click on **Object Manager**.
- In the QuickFind box, type **Purchase Order** and select the **Purchase Order** custom object.

3. From the left-hand menu, click **Validation Rules** → **New**.
4. Enter the following details:
 - o **Rule Name:** Expected Delivery Date Validation
 - o **Active:** Checked
5. In the formula editor, enter the error condition formula:
6. $(\text{Expected_Delivery_Date_c} - \text{Order_Date_c}) > 7$

This ensures that the expected delivery date cannot exceed 7 days from the order date.

7. Click **Save** to activate the validation rule.



Profiles

Creating an Inventory Manager Profile Steps:

Go to Setup → type Profiles in the Quick Find box → click Profiles.

Locate Standard User → click Clone.

Enter the Profile Name: Inventory Manager → click Save.

On the newly created profile page, click Edit.

Configure the following settings:

Custom App Settings: Set Medical Inventory Management as default.

Password Policies:

User passwords expire in: Never Expires

Minimum password length: 8

Click Save.

Creating a Purchase Manager Profile Steps:

Go to Setup → type Profiles in the Quick Find box → click Profiles.

Locate Standard User → click Clone.

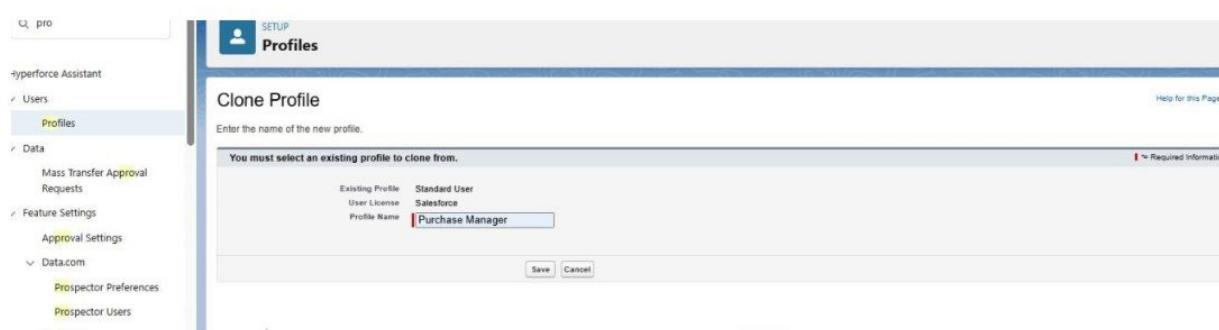
Enter the Profile Name: Purchase Manager → click Save.

On the newly created profile page, click Edit.

Configure the following settings:

Custom App Settings: Set Medical Inventory Management as default.

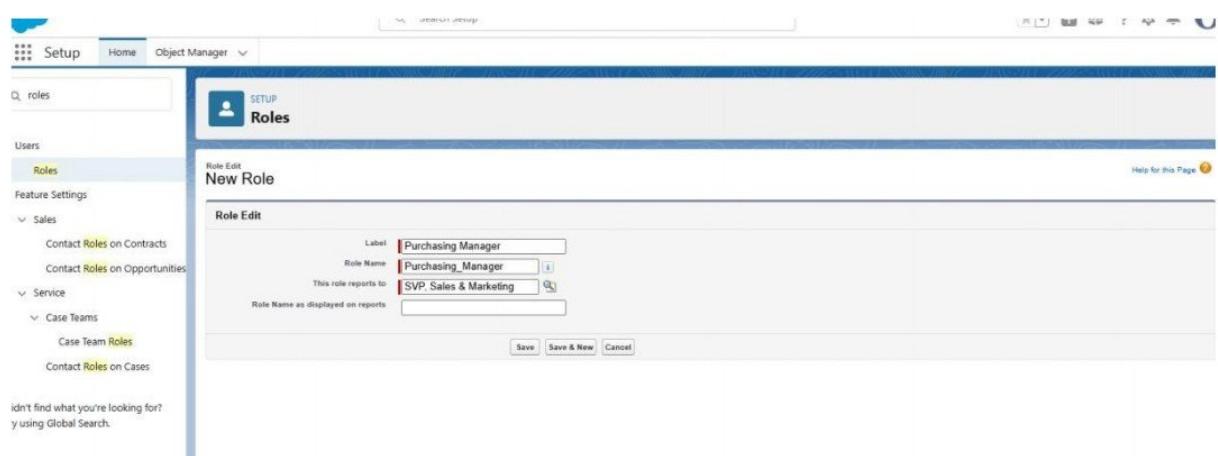
Click Save.



Roles

Creating a Purchasing Manager Role Steps:

1. Go to Setup → type Roles in the Quick Find box → click Set Up Roles.
2. Click Expand All to view the role hierarchy.
3. Under the SVP, Sales & Marketing role, click Add Role.
4. Enter the following details:
 - o Label: Purchasing Manager
 - o The Role Name will auto-populate.
5. Click Save to create the role.



Creating an Inventory Manager Role Steps:

Go to Setup → type Roles in the Quick Find box → click Set Up Roles.

Click Expand All to view the role hierarchy.

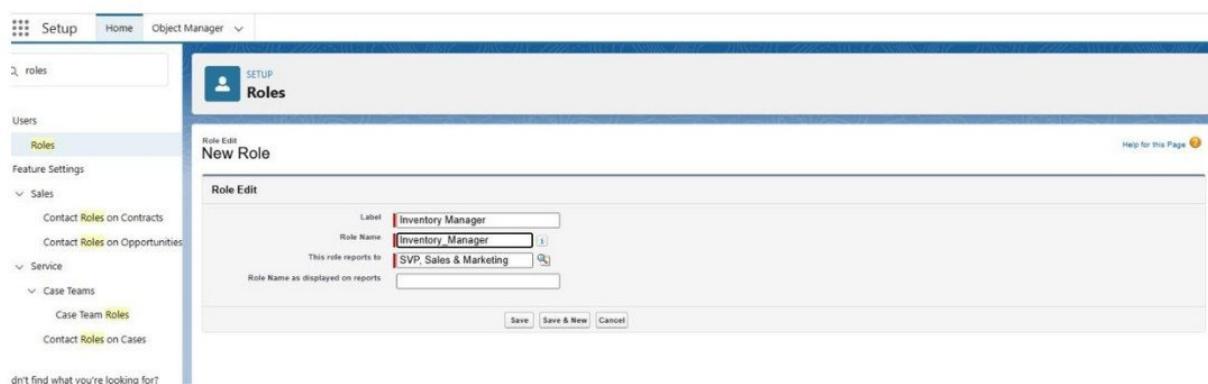
Under the SVP, Sales & Marketing role, click Add Role.

Enter the following details:

Label: Inventory Manager

The Role Name will auto-populate.

Click Save to create the role.



Permission Sets Activity 1:

Creating a Permission Set Steps:

Go to Setup → type Permission in the Quick Find box → select Permission Sets.

Click New.

Enter the following details:

Label: Purchase Manager Create Access

Click Save to create the permission set.

Action	Permission Set Name	Description	License
<input type="checkbox"/>	Close (Legacy) Data Cloud Da...	This Data Cloud permis...	Customer Data Platform
<input type="checkbox"/>	Close (Legacy) Data Cloud M...	Allows access to Data C...	Customer Data Cloud fo...
<input type="checkbox"/>	Close (Legacy) Data Cloud M...	This Data Cloud permis...	Customer Data Platform
<input type="checkbox"/>	Close (Legacy) Data Cloud M...	This Data Cloud permis...	Customer Data Platform
<input type="checkbox"/>	Close (Legacy) Data Cloud fo...	This Data Cloud permis...	Customer Data Cloud fo...
<input type="checkbox"/>	Close (Legacy) Data Cloud fo...	This Data Cloud permis...	Customer Data Cloud fo...
<input type="checkbox"/>	Close (Legacy) Data Cloud fo...	This Data Cloud permis...	Customer Data Cloud fo...
<input type="checkbox"/>	Access Agentforce Data -	Gives users access to t...	Agentforce (Default)
<input type="checkbox"/>	Close Agent Platform Builder	Allow access to agent pl...	Agent platform builder
<input type="checkbox"/>	Close Agentforce Default Adm...	Allows users to build an...	Agentforce (Default)
<input type="checkbox"/>	Close Agentforce Service Age...	Build and manage auto...	Agentforce Service Age...
<input type="checkbox"/>	Close Agentforce Service Age...	Access knowledge articl...	Agentforce Service Age...
<input type="checkbox"/>	Close Agentforce Service Age...	Set up and use Agentfir...	Agentforce Service Age...
<input type="checkbox"/>	Close Agentforce Service Age...	Analyze topics and perf...	Agentforce Service Age...

Flows

Creating a Flow to Update the Actual Delivery Date Steps:

Go to Setup → type Flow in the Quick Find box → click Flows → New Flow → select Start From Scratch.

Choose Record-Triggered Flow → click Create.

Under Object, select Purchase Order.

Configure the trigger: A record is created or updated.

Set Entry Conditions: None.

Select Fast Field Updates → click Done.

Get Records Element

1. Click the “+” icon → select Get Records.
2. Enter Label: Get Purchase Record.
3. Select Object: Purchase Order.
4. For Condition Requirements, choose All Conditions Are Met (AND).
5. Set the condition:

Field: Id

Operator: Equals

Value: {!\$Record.Id}

How Many Records to Store: Only the First Record.

How to Store Record Data: Choose fields and let Salesforce do the rest → select Order_Date_c → click Done.

Create a Variable

14. InFlowBuilder, click Manager → New Resource.
15. Resource Type: Variable
16. API Name: ActualDeliveryDate
17. Data Type: Date → click Done.

Assignment Element

18. Drag and drop Assignment from the Toolbox.
19. Enter Label: Assignment.
20. Set Variable Values:

Variable: {!ActualDeliveryDate}

Operator: Equals

Value: {!\$Record.Order_Date_c}

Variable: {!ActualDeliveryDate}

Operator: Add

Value: 3

Click Done.

Update Records Element

22. Drag and drop UpdateRecords → connect it to the Assignment element.
23. Enter Label: Updating Purchase Order.
24. How to Find Records to Update: Use the Purchase Order record that triggered the flow.
25. Filter Conditions: None—Always Update Record.
26. Set Field Values:

Field: Actual_Delivery_Date_c

Value: {!ActualDeliveryDate}

Click Done.

Save and Activate Flow

28. Save the flow as ActualDeliveryDate Updating.
29. Activate the flow.

Triggers

Creating a Trigger to Calculate Total Amount on Order Item

Step 1: Login to Salesforce

Log in to your Salesforce account with administrative privileges.

Step 2: Navigate to Developer Console

Click the **gear icon** (Setup) at the top-right corner → open the **Setup menu**.

Click **Developer Console** → opens in a new browser tab/window.

Step 3: Create the Apex Trigger

In Developer Console, go to **File → New → Apex Trigger**.

Name the trigger: `CalculateTotalAmountTrigger`.

Paste the following code:

```
trigger CalculateTotalAmountTrigger on Order_Item_c (after insert, after update,
after delete, after undelete) {
```

```
// Call the handler class to handle the logic
```

```
    CalculateTotalAmountHandler.calculateTotal(Trigger.new, Trigger.old,
Trigger.isInsert, Trigger.isUpdate, Trigger.isDelete, Trigger.isUndelete);
}
```

Step 4: Create the Apex Handler Class

In Developer Console, go to **File → New → Apex Class**.

Name it `CalculateTotalAmountHandler`.

Paste the following code:

```

public class CalculateTotalAmountHandler {

    // Method to calculate the total amount for Purchase Orders based on related
    Order Items

    public static void calculateTotal(List<Order_Item_c> newItems,
List<Order_Item_c> oldItems, Boolean isInsert, Boolean isUpdate, Boolean
isDelete, Boolean isUndelete) {

        // Collect Purchase Order IDs affected by changes in Order_Item_c
        Set<Id> parentIds = new Set<Id>();

        // For insert, update, and undelete scenarios
        if (isInsert || isUpdate || isUndelete) {
            for (Order_Item_c ordItem : newItems) {
                parentIds.add(ordItem.Purchase_Order_Id_c);
            }
        }

        // For update and delete scenarios
        if (isUpdate || isDelete) {
            for (Order_Item_c ordItem : oldItems) {
                parentIds.add(ordItem.Purchase_Order_Id_c);
            }
        }

        // Calculate the total amounts for affected Purchase Orders
        Map<Id, Decimal> purchaseToUpdateMap = new Map<Id, Decimal>();
        if (!parentIds.isEmpty()) {
            List<AggregateResult> aggrList = [
                SELECT Purchase_Order_Id_c, SUM(Amount __c) totalAmount
                FROM Order_Item_c
                WHERE Purchase_Order_Id_c IN :parentIds
            ];
            for (AggregateResult ar : aggrList) {
                purchaseToUpdateMap.put(ar.get('Purchase_Order_Id_c'), ar.get('totalAmount'));
            }
        }
    }
}

```

```

        GROUP BY Purchase_Order_Id__c
    ];

for (AggregateResult aggr : aggrList) {
    Id purchaseOrderId = (Id) aggr.get('Purchase_Order_Id_c');
    Decimal totalAmount = (Decimal) aggr.get('totalAmount');
    purchaseToUpdateMap.put(purchaseOrderId, totalAmount);
}

// Prepare Purchase Order records for update
List<Purchase_Order_c> purchaseToUpdate = new
List<Purchase_Order_c>();
for (Id purchaseOrderId : purchaseToUpdateMap.keySet()) {
    Purchase_Order__c purchaseOrder = new Purchase_Order__c(
        Id = purchaseOrderId,
        Total_Order_cost_c = purchaseToUpdateMap.get(purchaseOrderId)
    );
    purchaseToUpdate.add(purchaseOrder);
}

// Update Purchase Orders if there are any changes
if (!purchaseToUpdate.isEmpty()) {
    update purchaseToUpdate;
}
}

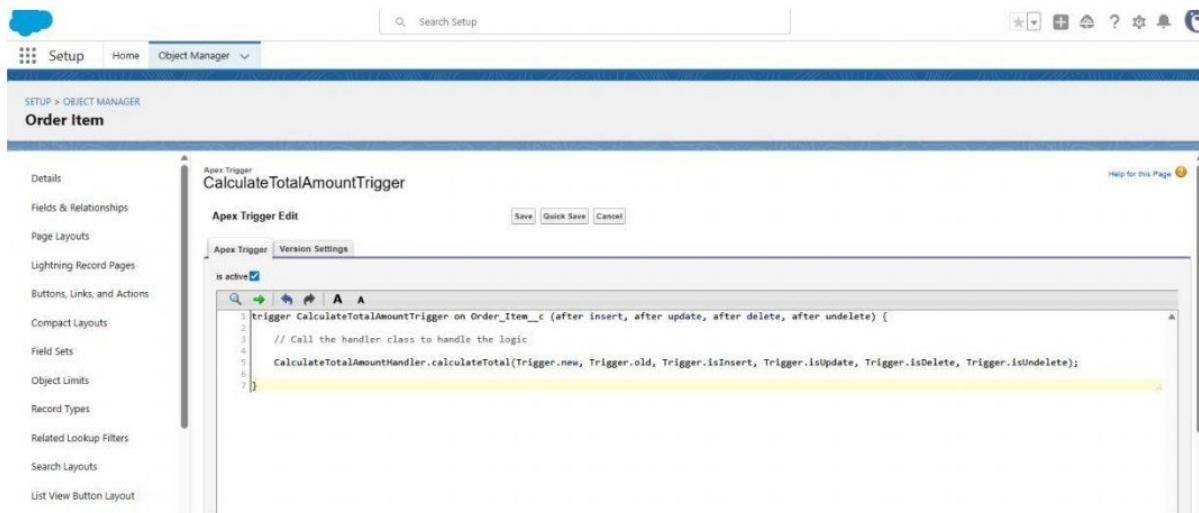
}

```

Step 5: Save and Test

Click **Save** for both the Trigger and the Handler Class.

Test by creating, updating, or deleting **Order Items**. The **Total Order Cost** on the related Purchase Order should update automatically.



Reports

Create Purchase Orders based on Suppliers(Summary) Report

1. Click App Launcher
2. Select Medical Inventory Management App
3. Click on Reports tab
4. Click on New Report.
5. Click the report type as Purchase Orders Click Start report.
6. Click on Filters and select as follows and click on Apply
7. Customize your report, in group rows select – Supplier ID, Purchase Order: Purchase Order ID, for columns Order Count, Total Order Cost (In this way we are making a Summary Report).
8. Click save and run
9. Give report name – Purchase Orders based on Suppliers.
10. Click Save

NOTE: In this report you can see your all record of the object you selected for reporting

What you selects in "Select a report type option")

(View Report

1. Click on App Launcher on the left side of the screen.
2. Search Medical Inventory Management App & click on it.
3. Click on Reports Tab.
4. Click on Purchase Orders based on Suppliers and see records.

Create a Complete Purchase Details

Report

1. Click App Launcher
2. Select Medical Inventory Management App
3. Click on Reports tab
4. Click on New Report.
5. Click the report type as Purchase Orders with Order Items and Product ID

>> Click Start report.

6. Click on Filters and select as follows and click on Apply

The screenshot shows the 'Create Report' dialog box. On the left, there's a sidebar with categories like 'Recently Used', 'All', 'Accounts & Contacts', etc. The main area has a search bar with 'Q: purch'. Below it, a table lists report types: Activities with Purchase Orders, Purchase Orders, Purchase Orders with Supplier, Purchase Orders with Order Items, Purchase Orders with Order Items and Product, Purchase Order History, and Inventory Transactions with Purchase Order. The 'Purchase Orders with Order Items' report is selected. The 'Details' pane on the right shows its icon, name ('Purchase Orders with Order Items'), category ('Standard'), and a 'Start Report' button.

Dashboards

Create Dashboard

Open the Dashboards tab within the Medical Inventory Management application.

Click New Dashboard.

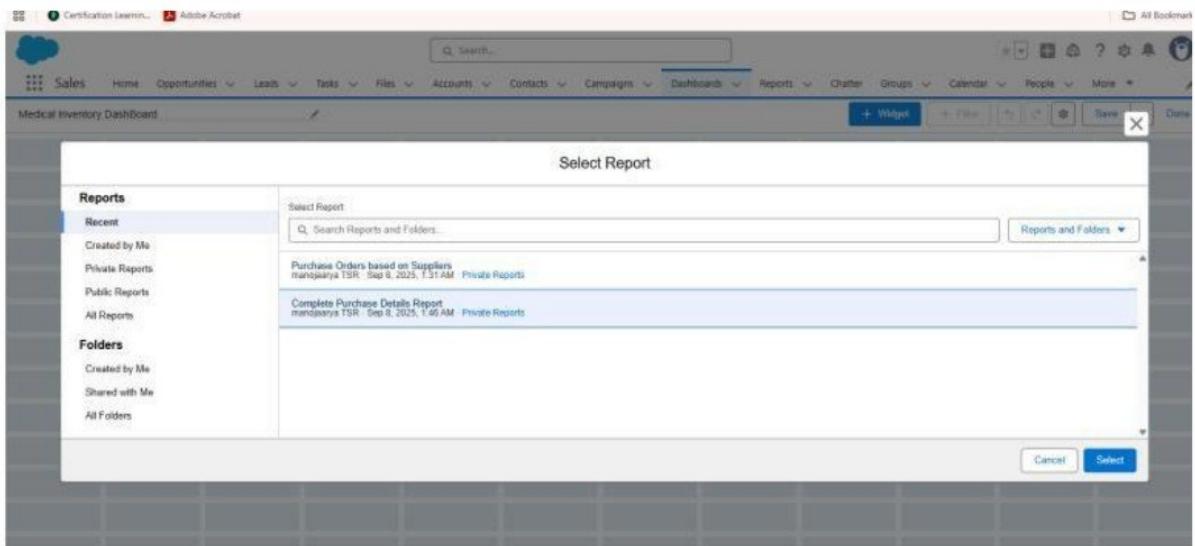
Enter the Name: Medical Inventory Dashboard → Click Create.

Click +Widget to add a component.

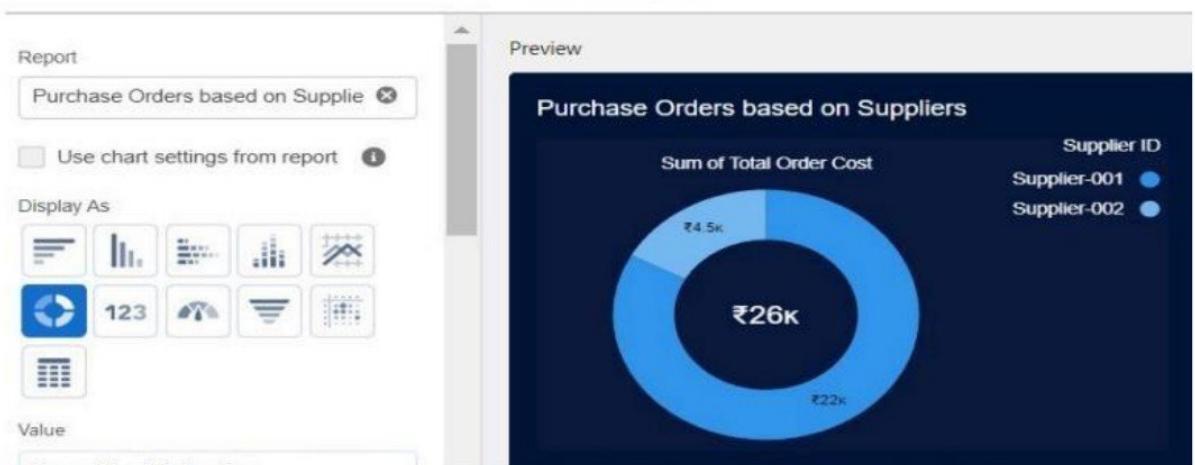
Select the Purchase Orders based on Suppliers report.

Choose a suitable data visualization type (chart, table, etc.) based on your requirement.

Click Add → then Save.



Add Widget



View Dashboard

Click on App Launcher (left-hand side of the screen).

Search for Medical Inventory Management → Click to open the app.

Go to the Dashboard tab.

Click on Medical Inventory Dashboard to view the graphical representation of records.

The top screenshot displays the Salesforce Setup Home page. It features a search bar at the top right, followed by navigation tabs for Setup, Home, and Object Manager. Below this is a sidebar with links to various Salesforce tools like App Launcher, Apps, and Administration. A main content area shows a "Cloud" section with a video thumbnail and a "Get Started with Einstein Bots" section with a "Get Started" button. To the right, there's a "Mobile Publisher" section with a "Learn More" button. A "Most Recently Used" list is also present.

The bottom screenshot shows the Medical Inventory Dashboard. The dashboard has a header with a search bar and navigation links for Products, Purchase Orders, Order Items, Inventory Transactions, Suppliers, Reports, and Dashboards. The main content area features a chart titled "Purchase Orders based on Suppliers" showing the sum of total order cost. The chart indicates that Supplier-001 accounts for 74% of the cost, totaling ₹26K, while Supplier-002 accounts for 26%.

Conclusion

The Medical Inventory Management System effectively automates and streamlines inventory management in a healthcare environment. By leveraging Salesforce CRM features, the system improves efficiency, ensures data accuracy, and enhances transparency in managing medical supplies. This project highlights the practical application of Salesforce in addressing real-world challenges, as part of the Naan Mudhalvan initiative.

Thank you