

Medical Inventory Management

College Name: Pollachi College of Arts & Science
College Code: bruaf

TEAM ID: NM2025TMID21332

TEAM MEMBERS:

Team Leader Name: JANANI S

Email : 23nmcsjanani@gmail.com

Team Member : SHANMUGAPRIYA G

Email : 23nmcsshanmugapriya@gmail.com

Team Member : NITHISH S

Email : 23nmcsnithish@gmail.com

Team Member : DHARSHAN R

Email : 23nmcsdharshan@gmail.com

1. INTRODUCTION

1.1 Project Overview

This project is a comprehensive Salesforce application to streamline and manage various operational aspects of medical inventory. The system aims to efficiently maintain supplier details, manage purchase orders, track product details and transactions, and monitor the expiry dates of products. Maintain detailed records of suppliers, including contact information. Catalog product information, including descriptions, stock levels. Monitor and track product expiry dates to avoid using expired items. Comprehensive reports to track supplier performance, and purchase orders.



1.2 Purpose:

The main objective of the project is to provide a centralized system that helps healthcare organizations efficiently track, control, and optimize their medical supplies and equipment. It ensures real-time visibility of stock levels, prevents shortages or overstocking, and reduces waste by monitoring expiry dates and usage patterns. By automating processes such as reordering, alerts, and reporting, it saves time and lowers costs while also supporting regulatory compliance through accurate record-keeping. Ultimately, effective medical inventory management improves patient care by ensuring that essential medicines and equipment are always available when needed.

DEVELOPMENT PHASE

Creating Developer Account:

By using this URL –

<https://www.salesforce.com/form/developer-signup/?d=pb>

The screenshot shows the Salesforce Developer Edition sign-up page. The header reads "Build enterprise-quality apps fast and get hands-on with Agentforce and Data Cloud." Below this, there's a list of benefits: "Sign up for your Developer Edition." followed by a bulleted list: "Build apps fast with drag-and-drop tools", "Go further with Apex code", "Build AI agents with Agentforce", "Harmonize your data with Data Cloud", "Ground Agentforce with structured and unstructured data", and "Integrate with anything using APIs". To the right, there's a form titled "Sign up for your Developer Edition" with fields for First name (Janani), Last name (S), Job title (Developer), Work email (23nmcsjanani@gmail.com), Company (Pollachi college of art), and Country/Region (India). A checkbox for agreeing to the terms and conditions is present, along with a link to the Privacy Statement. At the bottom, there's a note about org provisioning and a timestamp (2001 11-09-2025).

https://www.salesforce.com/form/developer-signup/?d=pb

Build enterprise-quality apps fast and get hands-on with Agentforce and Data Cloud.

Sign up for your Developer Edition.

- ✓ Build apps fast with drag-and-drop tools
- ✓ Go further with Apex code
- ✓ Build AI agents with Agentforce
- ✓ Harmonize your data with Data Cloud
- ✓ Ground Agentforce with structured and unstructured data
- ✓ Integrate with anything using APIs

Agentforce

Customer Support

Steps
Select Type

First name Janani ✓

Last name S ✓

Job title Developer ✓

Work email 23nmcsjanani@gmail.com ✓

Company Pollachi college of art ✓

Country/Region India ✓

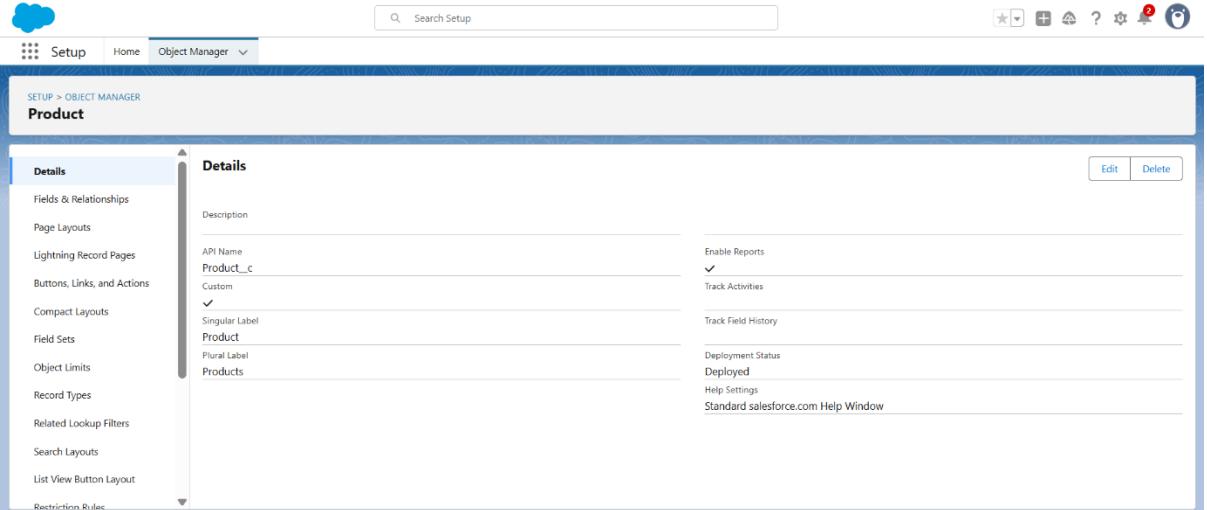
Your org may be provisioned on or migrated to Hyperforce, Salesforce's public cloud infrastructure.

I agree to the Main Services Agreement – Developer Services and Salesforce Program Agreement. I acknowledge, as described in the Developer Documentation: (1) the Developer Edition includes autonomous and other generative AI features; and (2) Salesforce may limit use of those features and the org, and may terminate any org that has been inactive for 45 days.

We value your privacy. To learn more, visit our Privacy Statement.

2001 11-09-2025

- Created objects: Product, Purchase Order, Order Item, Inventory Transaction and Supplier objects.



The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar and various icons are also present. The main area displays the 'Product' object details. On the left, a sidebar lists navigation options: Details, Fields & Relationships, 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, and Restriction Rules. The main 'Details' section shows the API Name as 'Product_c', which is custom and singular. It also shows the plural label as 'Products'. Other settings include 'Enable Reports' (checked), 'Track Activities' (checked), 'Track Field History' (unchecked), 'Deployment Status' (Deployed), and 'Help Settings' (Standard salesforce.com Help Window). There are 'Edit' and 'Delete' buttons at the top right of the details section.

- Purchase Order Object

The screenshot shows the Salesforce Setup interface under the Object Manager. The main title is "Purchase Order". On the left, a sidebar lists various configuration options: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, and List View Button Layout. The main panel is titled "Details" and contains the following information:

Description	
API Name	Purchase_Order__c
Custom	<input checked="" type="checkbox"/>
Singular Label	Purchase Order
Plural Label	Purchase Orders
Enable Reports	<input checked="" type="checkbox"/>
Track Activities	
Track Field History	
Deployment Status	Deployed
Help Settings	Standard salesforce.com Help Window

Buttons at the top right of the main panel are "Edit" and "Delete".



Order Item Object

The screenshot shows the Salesforce Setup interface under the Object Manager. The main title is "Order Item". The left sidebar is identical to the one in the previous screenshot. The main panel is titled "Details" and contains the following information:

Description	
API Name	Order_Item__c
Custom	<input checked="" type="checkbox"/>
Singular Label	Order Item
Plural Label	Order Items
Enable Reports	<input checked="" type="checkbox"/>
Track Activities	
Track Field History	
Deployment Status	Deployed
Help Settings	Standard salesforce.com Help Window

Buttons at the top right of the main panel are "Edit" and "Delete".



Inventory Transaction Object

Setup | Home | Object Manager

SETUP > OBJECT MANAGER
Inventory Transaction

Details

Fields & Relationships
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

Details

Description
API Name: **Inventory_Transaction__c**
Custom:
Singular Label: **Inventory Transaction**
Plural Label: **Inventory Transactions**

Enable Reports:
Track Activities
Track Field History
Deployment Status: **Deployed**
Help Settings: Standard salesforce.com Help Window

Edit | Delete



Supplier Object

Setup | Home | Object Manager

SETUP > OBJECT MANAGER
Supplier

Details

Fields & Relationships
Page Layouts
Lightning Record Page
Buttons, Links, and Actions
Compact Layouts
Field Sets
Object Limits
Record Types
Related Lockup Filters
Search Layouts
List View Button Layout

Details

Description
API Name: **Suppliers__c**
Custom:
Singular Label: **Supplier**
Plural Label: **Suppliers**

Enable Reports:
Track Activities
Track Field History
Deployment Status: **Deployed**
Help Settings: Standard salesforce.com Help Window

Edit | Delete

- Created tabs: Product, Purchase Order, Order Item, Inventory Transaction and Supplier objects.

The screenshot shows the Salesforce Setup interface with the 'Custom Tabs' page open. The left sidebar shows 'User Interface' and 'Tabs'. The main content area has a heading 'Custom Tabs' with a sub-section 'Custom Object Tabs'. A table lists the following tabs:

Action	Label	Tab Style	Description
Edit Del	Inventory Transactions	Thermometer	
Edit Del	Order Items	Stethoscope	
Edit Del	Products	Stethoscope	
Edit Del	Purchase Orders	Stethoscope	
Edit Del	Suppliers	Thermometer	

Below this are sections for 'Web Tabs', 'Visualforce Tabs', 'Lightning Component Tabs', and 'Lightning Page Tabs', each stating 'No [tab type] have been defined'.

- Creating a Lightning App for Medical Inventory Management.

➤ Created a Text Field, Text Area Field, Number Field, Currency Field, Date Field in Product Object.

- Created a Text Field, Lookup relationship, Date Field, Roll-up summary Field, Currency field in Purchase Order Object.

Field Label	API Name	Type
Actual Delivery Date	Actual_Delivery_Date__c	Date
Created By	CreatedBy	Lookup(User)
Expected Delivery Date	Expected_Delivery_Date__c	Date
Last Modified By	LastModifiedBy	Lookup(User)
Order Count	Order_Count__c	Roll-Up Summary (COUNT Order Item)
Order Date	Order_Date__c	Date
Owner	OwnerId	Lookup(User,Group)
Purchase Order ID	Name	Text(80)
Supplier ID	Supplier_ID__c	Lookup(Supplier)
Total Order Cost	Total_Order_Cost__c	Currency(16, 2)

- Created a Text Field, Lookup Relationship, Master-detail, Number Field, Formula Field in Order Item Object.

Field Label	API Name	Type
Amount	Amount_del_c	Number(18, 0)
Created By	CreatedBy	Lookup(User)
Last Modified By	LastModifiedBy	Lookup(User)
Order Item ID	Name	Text(80)
Product ID	Product_ID__c	Lookup(Product)
Purchase Order ID	Purchase_Order_ID__c	Master-Detail(Purchase Order)
Quantity Ordered	Quantity__c	Number(18, 0)
Quantity Received	Quantity_Received__c	Number(18, 0)
Unit Price	Unit_Price_del_c	Number(15, 2)

- Created a Text Field, Lookup Relationship, Date Field, Picklist Field, Formula Field in Inventory Transaction Object.

Fields & Relationships					
	FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Page Layouts	Created By	CreatedById	Lookup(User)		
Lightning Record Pages	Last Modified By	LastModifiedById	Lookup(User)		
Buttons, Links, and Actions	Owner	OwnerId	Lookup(User,Group)		▼
Compact Layouts	Purchase Order ID	Purchase_Order_ID__c	Lookup(Purchase Order)		▼
Field Sets	Total Order Cost	Total_Order_Cost__c	Formula (Currency)		▼
Object Limits	Transaction Date	Transaction_Date__c	Date		▼
Record Types	Transaction ID	Name	Text(80)		▼
Related Lookup Filters	Transaction Type	Transaction_Type__c	Picklist		▼
Search Layouts					
List View Button Layout					
Restriction Rules					

➤ Created a Text field, Phone Field, Email Field, TextArea Field in Supplier Object.

Fields & Relationships					
	FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Page Layouts	Address	Address__c	Text Area(255)		▼
Lightning Record Pages	Contact Person	Contact_Person__c	Text(15)		▼
Buttons, Links, and Actions	Created By	CreatedById	Lookup(User)		
Compact Layouts	Email	Email__c	Email		▼
Field Sets	Last Modified By	LastModifiedById	Lookup(User)		▼
Object Limits	Owner	OwnerId	Lookup(User,Group)		▼
Record Types	Phone Number	Phone_Number__c	Phone		▼
Related Lookup Filters	Supplier ID	Name	Text(80)		▼
Search Layouts	Supplier Name	Supplier_Name__c	Text(15)		▼
List View Button Layout					
Restriction Rules					

➤ Edit a Page Layout: Product Object, Purchase Order Object, Order Item Object, Inventory Transaction Object, Supplier Object

The image consists of two vertically stacked screenshots of the Salesforce Object Manager interface, specifically for the Product object.

Screenshot 1: Product Page Layout Setup

This screenshot shows the "Page Layouts" tab selected in the left sidebar. The main area displays the "Product Detail" page layout. At the top, there's a "Fields" section containing fields like "Section", "Expiry Date", "Product Description", "Blank Space", "Last Modified By", "Product ID", "Created By", "Minimum Stock Level", "Product Name", "Current Stock Level", "Owner", and "Unit Price". Below this is the "Product Detail" section, which includes "Information" and "System Information" headers. The "Information" header contains fields for Product ID, Product Name, Product Description, Unit Price, Current Stock Level, Minimum Stock Level, and Owner. The "System Information" header contains fields for Created By and Last Modified By.

Screenshot 2: Product Page Layout List

This screenshot shows the "Page Layouts" tab selected in the left sidebar. The main area displays a list of page layouts for the Product object. The table has columns for "PAGE LAYOUT NAME", "CREATED BY", and "MODIFIED BY". There is one item listed: "Product Layout" created by Sajinath S. on 9/5/2025, 2:10 AM, last modified by Sajinath S. on 9/9/2025, 7:19 PM.

➤ Purchase Order Object

Setup > Object Manager > Purchase Order

Page Layouts

Purchase Order Detail

Field Name	Type	Description
Expected Delivery Date	Date	Owner
Last Modified By	User	Purchase Order ID
Actual Delivery Date	Date	Order Count
Created By	User	Supplier ID
Order Date	Date	Total Order Cost

Information (Header visible on edit only)

- Purchase Order ID: Sample Text
- Supplier ID: Sample Text
- Order Date: 9/9/2025
- Expected Delivery Date: 9/9/2025

System Information (Header visible on edit only)

- Actual Delivery Date: 9/9/2025
- Order Count: 85,095
- Total Order Cost: \$123.45
- Owner: Sample Text

Search Setup

Page Layouts

PAGE LAYOUT NAME ▲ **CREATED BY** **MODIFIED BY**

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Purchase Order Layout	Sajinath S, 9/5/2025, 2:18 AM	Sajinath S, 9/9/2025, 7:21 PM

➤ Order Item Object

Order Item

Page Layouts

Order Item Detail

Product Details

System Information

Page Layouts

Order Item Layout

➤ Inventory Transaction Object

Inventory Transaction

Page Layouts

Inventory Transaction Detail

Page Layouts

Inventory Transaction Layout

SETUP > OBJECT MANAGER
Inventory Transaction

Details
Fields & Relationships
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
Scoping Rules
Object Access
Triggers
Flow Triggers
Validation Rules
Conditional Field Formatting

Page Layouts
1 Items, Sorted by Page Layout Name

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Inventory Transaction Layout	Sajinath S, 9/5/2025, 2:20 AM	Sajinath S, 9/9/2025, 7:26 PM

➤ Supplier Object

SETUP > OBJECT MANAGER
Supplier

Details
Fields & Relationships
Page Layouts
Lightning Record Pages
Buttons, Links, and Actions
Compact Layouts
Field Sets
Object Limits
Record Types
Related Lookup Filters

Page Layouts

Save ▾ Quick Save Preview As... Cancel Undo Redo Layout Properties

Fields

Field Name	Created By	Phone Number
Section	Created By	Phone Number
Blank Space	Email	Supplier ID
Address	Last Modified By	Supplier Name
Contact Person	Owner	

Information (Header visible on edit only)

Supplier ID	Sample Text	Phone Number	1-415-555-1212
Supplier Name	Sample Text	Email	sarah.sample@company.com
Contact Person	Sample Text	Address	Sample Text
		Owner	Sample Text

System Information (Header visible on edit only)

Created By	Sample Text	Last Modified By	Sample Text

Page Layouts

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Supplier object Layout	Sajinath S, 9/5/2025, 2:22 AM	Sajinath S, 9/9/2025, 7:29 PM

➤ Creating a Compact Layout: Product Object, Purchase Order Object

Product Compact Layout

Compact Layout Edit

Select Compact Layout Fields

Product Compact Layouts

Compact Layout Assignment

Primary Compact Layout: Product Compact Layout

The screenshot shows the Salesforce Setup interface for the Product object. The left sidebar lists various configuration options under 'Compact Layouts'. The main area displays a table titled 'Compact Layouts' with two items:

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
Product Compact Layout	Product_Compact_Layout	✓	Sajinath S	9/7/2025, 6:28 AM
System Default	SYSTEM			

➤ Purchase Order Object

The screenshot shows the Salesforce Setup interface for the Purchase Order object. The left sidebar lists various configuration options under 'Compact Layouts'. The main area displays a table titled 'Compact Layouts' with two items:

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
Purchase Order Compact Layout	Purchase_Order_Compact_Layout	✓	Sajinath S	9/7/2025, 6:33 AM
System Default	SYSTEM			

Purchase Order

Compact Layout Assignment

Primary Compact Layout

Select the compact layout to use when this object's records appear as list items in the mobile app.

Primary Compact Layout: Purchase Order Compact Layout

Compact Layout Edit

Enter Compact Layout Information

Label: Purchase Order Compact L
Name: Purchase_Order_Compact

Select Compact Layout Fields

Available Fields: Actual Delivery Date, Created By, Expected Delivery Date, Last Modified By, Order Count, Owner

Selected Fields: Purchase Order ID, Order Date, Total Order Cost, Supplier ID

Add, Remove, Top, Up, Down, Bottom

Use SHIFT + click to select adjacent fields. Use CTRL + click to select an assortment of fields.

➤ Creating an Expected Delivery Date Validation rule to a Employee Object

Validation Rule Edit

Rule Name: Expected_Delivery_Date_Validation

Active:

Description:

Error Condition Formula

Example: Discount_Percent_c < 0.30 | More Examples...
Display an error if Discount is more than 30%
If this formula expression is true, display the text defined in the Error Message area

Functions

ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN

Insert Field, Insert Operator, Insert Selected Function, ABS(number), Returns the absolute value of a number, a number without its sign

Error Message

Example: Discount percent cannot exceed 30%
This message will appear when Error Condition formula is true

Error Message: The Expected Delivery Date should not exceed 7 days.

This error message can either appear at the top of the page or below a specific field on the page

Error Location: Top of Page Field

Quick Tips

- Operators & Functions

The screenshot shows the Salesforce Setup interface for the Purchase Order object. The left sidebar lists various configuration options like Details, Fields & Relationships, and Validation Rules. The Validation Rules section is selected, displaying a table with one item:

RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY
Expected_Delivery_Date_Validation	Top of Page	The Expected Delivery Date should not exceed 7 days.	✓	Sajinath S., 9/7/2025, 6:38 AM

➤ Created a Profile: Inventory Manager Profile, Purchase Manager Profile

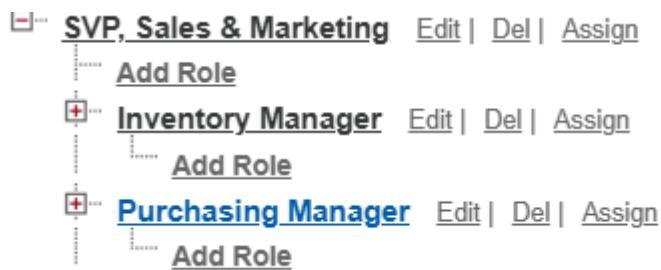
The screenshot shows the Salesforce Profiles page. The table displays the Purchase Manager profile, which includes the Identity User license and is assigned to the Custom user type.

Action	Profile Name	User License	Custom
<input type="checkbox"/> Edit Clone	Identity_User	Identity	<input type="checkbox"/>
<input type="checkbox"/> Edit Del ...	Inventory_Manager	Salesforce	✓

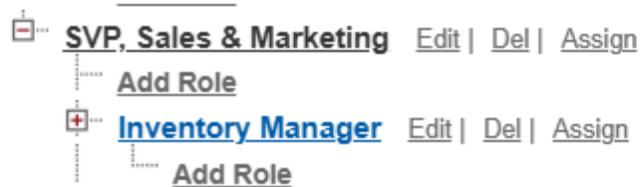
➤ Purchase Manager Profile

Profiles		
All Profiles Edit Delete Create New View		
Action	Profile Name	User License
Edit Clone	Partner App Subscription User	Partner App Subscription
Edit Clone	Partner Community Login User	Partner Community Login
Edit Clone	Partner Community User	Partner Community
Edit Del...	Purchase Manager	Salesforce

- Created a Role: Purchasing Manager, Inventory Manager.



- Inventory Manager



➤ Creating a Permission Set.

API Name

Edit Properties

Label Purchase Manager Create /

API Name Purchase_Manager [i](#)

Description

Session Activation [i](#)

Required

Save **Cancel**

SETUP

Permission Sets

On this page you can create, view, and manage permission sets.

All Permission Sets [Edit](#) | [Delete](#) | [Create New View](#)

New [New](#) [Clone](#)

Action	Permission Set Name	Description	License
<input type="checkbox"/>	Partner Connect Partner Admin Setup	Set up Partner Connect from a partner org. Partner Connect is a Partner R...	Salesforce
<input type="checkbox"/>	Payments Administrator	Has all the user permissions to gate access to APIs that are available to Sa...	Salesforce Payments Internal
<input type="checkbox"/>	Prompt_Template_Manager	Manage prompt templates using Prompt Builder and run them using genera...	Einstein Prompt Templates
<input type="checkbox"/>	Prompt_Template_User	Run prompt templates using generative AI features.	Einstein Prompt Templates
<input type="checkbox"/>	PromptTemplatePermSet		Cloud Integration User
<input type="checkbox"/>	Publish Suggested for You Nudges: Integration User	Access the Core Adoption Service and tenant orgs, which are used to publi...	Cloud Integration User
<input type="checkbox"/>	Purchase Manager Create Access		

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | All

Help for this Page [?](#)

➤ Creating Flow to update the Actual Delivery Date.

SETUP Flows

Flow Definitions All Flows 

50+ items • Sorted by Flow Label • Filtered by All flow definitions • Updated a few seconds ago

Flow Label ↑	Process Type	A...	Te...	Package State	Pa...	La...	La...
Add or Modify Service Appointment Attendees	Salesforce Scheduler Flow	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Managed-Installed			
Approvals Workflow: Evaluate Approval Requests	Screen Flow	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-Installed			
Approvals Workflow: Process Approval Submission	Screen Flow	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Managed-Installed			
Authentication Provider User Registration	Identity User Registration Fl...	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-Installed			
Basic Approval Request	Flow Orchestration for CMS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-Installed			
Book Appointment from Invitation	Salesforce Scheduler Flow	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-Installed			
Cancel Item Flow	Screen Flow	<input type="checkbox"/>	<input type="checkbox"/>	Managed-Installed			

Record-Triggered Flow Start

Object: Purchase Order Trigger: A record is created or updated Optimize for: Fast Field Updates

Open Flow Trigger Explorer for Purch...

Add Element

get

Get Records

Configure Start

Run Debug View Tests Save As New Version Save Activate

Optimize Flow

Optimize the Flow for:

Fast Field Updates

Update fields on the record that triggers the flow to run. This high-performance flow runs **before the record is saved** to the database.

Actions and Related Records

Update any record and perform actions, like send an email. This more flexible flow runs **after the record is saved** to the database.

Is this flow making an external callout or connecting to an external system?

An asynchronous path is required for flows that involve external systems.

Add Asynchronous Path

Record-Triggered Flow Start

Object: Purchase Order Trigger: A record is created or updated Optimize for: Fast Field Updates

Open Flow Trigger Explorer for Purch...

Get Purchase Record

Get Records

How Many Records to Store

Only the first record

How to Store Record Data

Choose fields and let Salesforce do the rest

Select Purchase Order Fields to Store in Variable

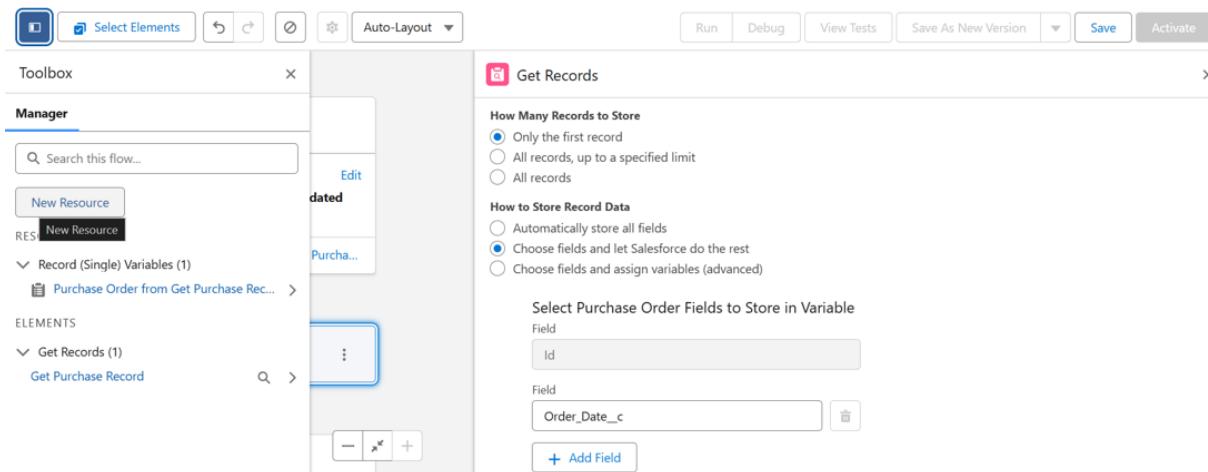
Field: Id

Field: Order_Date__c

Add Field

End

Run Debug View Tests Save As New Version Save Activate



New Resource

* Resource Type
Variable

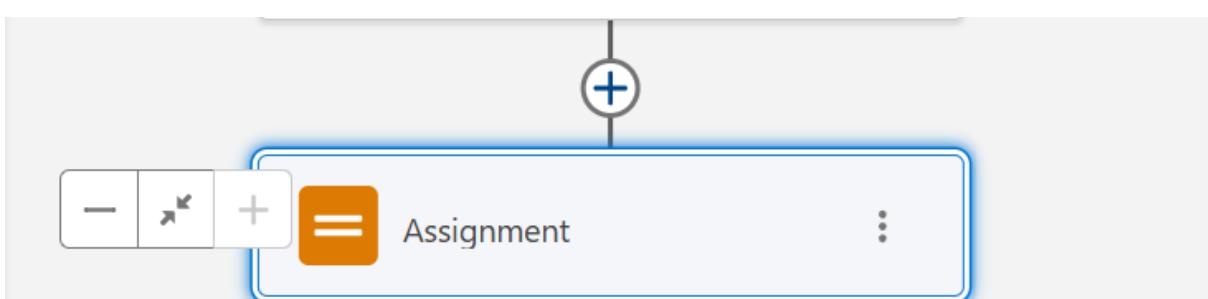
* API Name ⓘ
ActualDeliveryDate

Description

* Data Type
Date Allow multiple values (collection) ⓘ

Cancel Done

This is a 'New Resource' configuration dialog. It requires filling out fields for Resource Type (Variable), API Name (ActualDeliveryDate), and Data Type (Date). An optional checkbox allows for multiple values (collection). At the bottom are 'Cancel' and 'Done' buttons.



Variable	Operator	Value
<input type="text" value="ActualDeliveryDate"/> X	Add	3
🔍 trash		
+ Add Assignment		

Add Element

upda

Update Triggering Record (i)

Update Related Records

Update Records (i)

Update Records: Update Salesforce records using values from the flow.

Update Records X

* Label (i) * API Name (i)

Updating Purchasing Order Updating_Purchasing_Order

Description

*** How to Find Records to Update and Set Their Values**

- Use the purchase order record that triggered the flow
- Update records related to the purchase order record that triggered the flow
- Use the IDs and all field values from a record or record collection
- Specify conditions to identify records, and set fields individually

(i) Because this flow runs *before* a record is saved, you can only update the record that triggered the flow to run. To update other records, configure the trigger to run the flow *after* the record is saved.

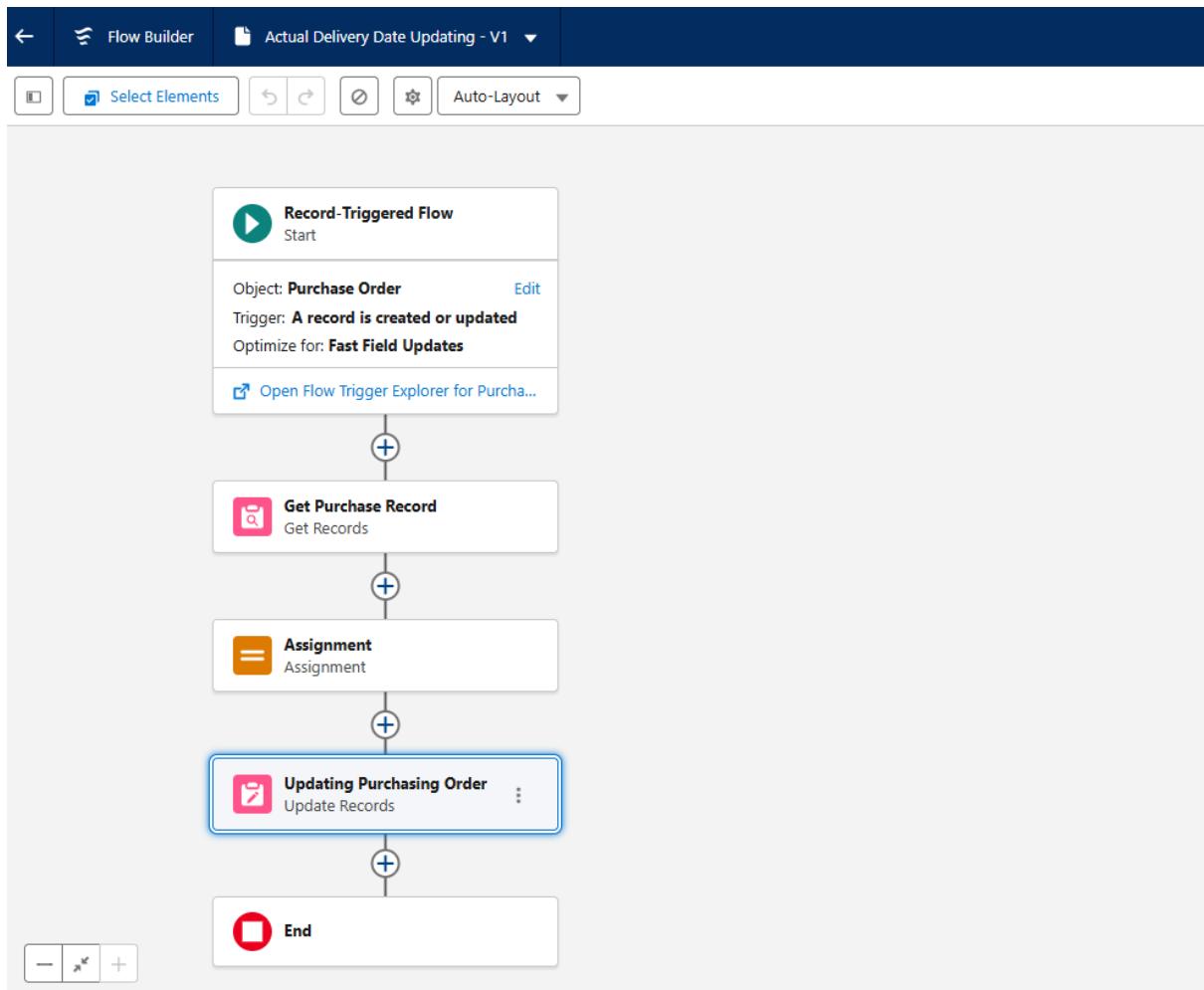
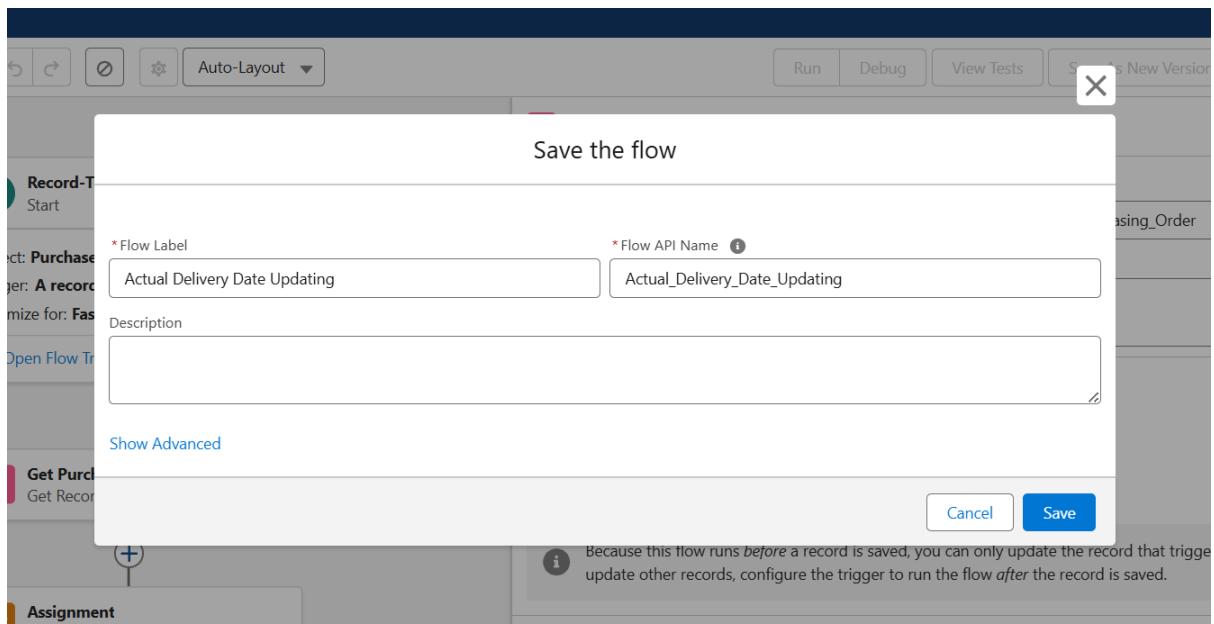
Set Filter Conditions

Condition Requirements to Update Record

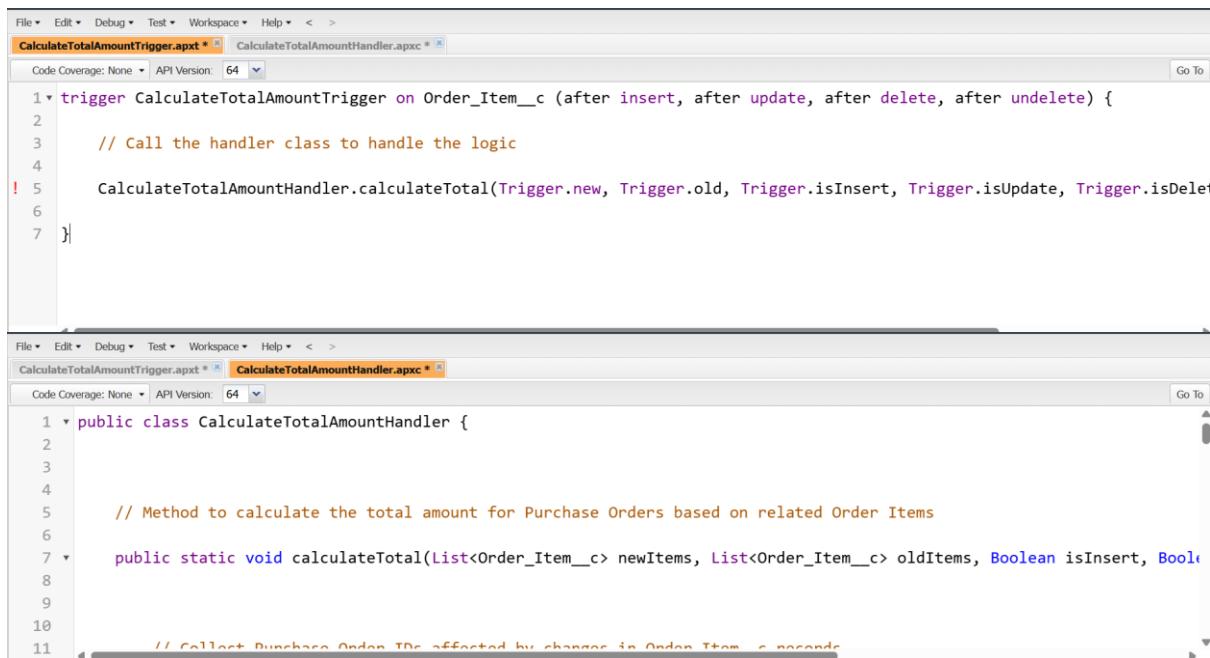
None—Always Update Record

Set Field Values for the Purchase Order Record

Field	Value
<input type="text" value="Actual Delivery Date"/> X	<input type="text" value="ActualDeliveryDate"/> X
← 🔍 trash	



➤ Create a Trigger to Calculate total amount on Order Item.

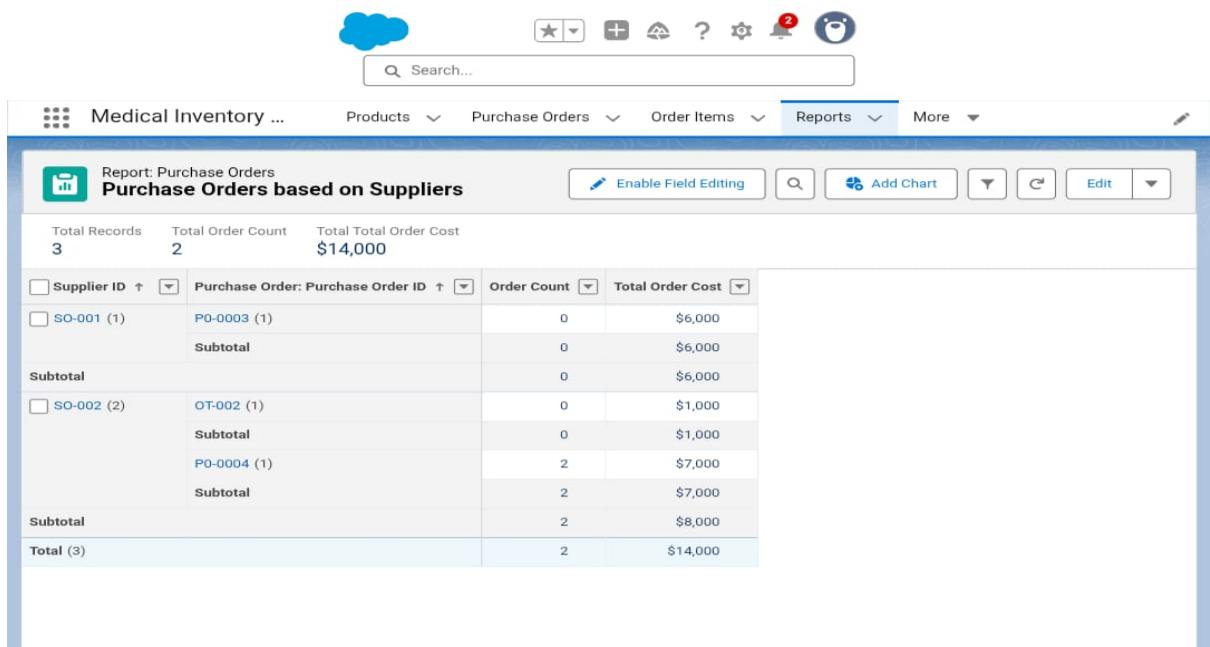


The image shows two side-by-side code editors in a Salesforce development environment. The top editor contains a trigger named 'CalculateTotalAmountTrigger' on the 'Order_Item__c' object. It includes logic to call a handler class 'CalculateTotalAmountHandler' to calculate the total amount based on insert, update, delete, and undelete events. The bottom editor contains the 'CalculateTotalAmountHandler' class, which has a static method 'calculateTotal' that takes lists of new and old items, and boolean parameters for insert, update, and delete operations. The code also includes a note about collecting Purchase Order IDs affected by changes in Order Item records.

```
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);
}

public class CalculateTotalAmountHandler {
    public static void calculateTotal(List<Order_Item__c> newItems, List<Order_Item__c> oldItems, Boolean isInsert, Boolean isUpdate, Boolean isDelete) {
        // Collect Purchase Order IDs affected by changes in Order Item__c records
    }
}
```

➤ Created Reports: Purchase Orders Based on Suppliers summary), Purchase Details



The image shows a screenshot of a medical inventory system interface. At the top, there's a navigation bar with links for Medical Inventory, Products, Purchase Orders, Order Items, Reports, More, and a search bar. Below the navigation is a report titled 'Report: Purchase Orders Purchase Orders based on Suppliers'. The report displays a table with the following data:

Total Records	Total Order Count	Total Total Order Cost	
3	2	\$14,000	
Supplier ID	Purchase Order: Purchase Order ID	Order Count	Total Order Cost
SO-001 (1)	PO-0003 (1)	0	\$6,000
	Subtotal	0	\$6,000
		0	\$6,000
Subtotal			
SO-002 (2)	OT-002 (1)	0	\$1,000
	Subtotal	0	\$1,000
	PO-0004 (1)	2	\$7,000
	Subtotal	2	\$7,000
Subtotal		2	\$8,000
Total (3)		2	\$14,000

➤ Purchase Details

Order Count	Actual Delivery Date	Supplier ID	Purchase Order: Purchase Order ID	Order Item: Order Item ID	Product: Product Name
1 (1)	9/21/2025 (1)	So-004 (1)	So-004	So-004	citizine
		Subtotal			
Subtotal					
2 (2)	9/9/2025 (2)	SO-002 (2)	P0-0004	OT-002	aspirin
		Subtotal	P0-0004	OT-001	DOLO650
Subtotal					
Total (3)					
Row Counts <input checked="" type="checkbox"/> Detail Rows <input checked="" type="checkbox"/> Subtotals <input checked="" type="checkbox"/> Grand Total <input checked="" type="checkbox"/>					

➤ Created Dashboard

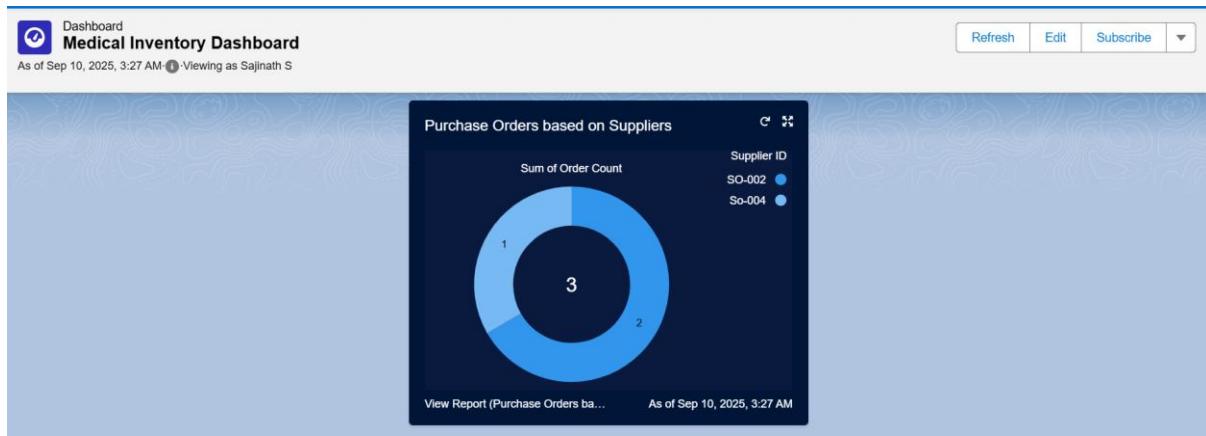
New Dashboard

* Name
Medical Inventory DashBoard

Description

Folder
Private Dashboards

➤ View Dashboard



➤ Advantages and Disadvantages of Medical Inventory Management

■ Advantages

- > **Centralized Inventory Tracking** – Manage all medicines, supplies, and equipment in one place.
- > **Real-Time Stock Visibility** – Check available stock instantly across departments or branches.
- > **Automation** – Auto-reordering when stock is low, expiry alerts, and stock updates.
- > **Integration** – Connects with ERP, EHR, pharmacy, billing, and POS systems.
- > **Compliance Support** – Helps follow FDA, HIPAA, CDSCO and other medical regulations.
- > **Advanced Reporting** – Generate audit-ready reports on usage, expiry, and stock history.

- > **Better Patient Care** – Ensures required medicines and equipment are available on time.
- > **Scalability** – Works for small clinics, hospitals, and large healthcare chains.
- > **Customization** – Workflows, dashboards, and automation can be fully tailored.
- > **Improved Decision-Making** – Data-driven insights help in better procurement planning.

■ **Disadvantages**

- > **High Setup Cost** – Salesforce licensing and customization are expensive.
- > **Complex Customization** – Advanced medical inventory needs skilled Salesforce developers.
- > **Training Requirement** – Staff, nurses, and pharmacists need proper training to use it.
- > **Internet Dependency** – Being cloud-based, it requires a stable connection.
- > **Data Security Concerns** – Sensitive healthcare data needs strict security controls.
- > **Ongoing Maintenance Costs** – Continuous updates, integration fixes, and enhancements add expenses.
- > **Integration Challenges** – Linking with existing hospital systems can be complex.

- > **User Adoption Issues** – If staff don't update stock properly, data becomes inaccurate.
- > **Compliance Risk** – Misconfigured access or incorrect reporting may cause legal issues.
- > **Overkill for Small Clinics** – For small setups, Salesforce may be too heavy and costly.

➤ **Conclusion**

Medical inventory management in Salesforce provides healthcare organizations with a centralized, efficient, and automated platform to track, manage, and optimize their medical supplies. By integrating real-time inventory tracking, automated stock updates, and detailed reporting, Salesforce helps reduce manual errors, avoid stockouts or overstocking, and ensure timely availability of essential medical items.

However, successful implementation requires proper configuration, user training, and data accuracy. When used effectively, Salesforce empowers hospitals, clinics, and healthcare providers to streamline operations, improve patient care, and reduce operational costs, making it a powerful solution for modern medical inventory management.

➤ **APPENDIX**

Source Code: Provided in Apex Triggers and Apex Classes

● **CalculateTotalAmountTrigger**

```
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);
}
```

● **CalculateTotalAmountHandler**

```
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 records
        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__);
            }
        }
    }
}
```

```

}

// 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()) {
    // Perform an aggregate query to sum the Amount__c for each Purchase Order
    List<AggregateResult> aggrList = [
        SELECT Purchase_Order_Id__c, SUM(Amount__c)
        totalAmount
        FROM Order_Item__c
        WHERE Purchase_Order_Id__c IN :parentIds
        GROUP BY Purchase_Order_Id__c
    ];

    // Map the result to Purchase Order IDs
    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;
        }
    }
}
```