StockBridge: Real-Time Inventory and Supplier Integration

User Story

The Smart Inventory Management System enhances inventory oversight and sales order tracking by integrating key functionalities. When an inventory item's stock level falls below a predefined threshold, the system automatically triggers reorder notifications to suppliers, ensuring timely restocking of products. As sales orders are created, updated, or canceled, the system adjusts stock quantities accordingly to maintain accurate inventory records. Additionally, customers receive email notifications regarding the status of their sales orders, keeping them informed about changes such as order completions or cancellations. This seamless integration of features not only enables efficient inventory management and reduces the risk of stockouts but also strengthens communication with suppliers and keeps customers updated, ultimately improving overall operational efficiency and satisfaction.

Objective:-

In the Smart Inventory Management project, I developed three essential objects: Inventory, Sales Order, and Supplier. Each object features crucial fields that capture vital information; the Inventory object includes product name, stock quantity, and reorder thresholds, ensuring effective monitoring of product availability. The Sales Order object is designed to record customer orders, detailing quantity ordered, associated products, and the current order status. The Supplier object manages communication with suppliers for timely stock replenishments. By establishing connections between these objects, I facilitate efficient interactions among inventory tracking, order processing, and supplier notifications. To enhance operational efficiency, I employed Salesforce Apex Triggers to automate reorder notifications to suppliers and used flows to send real-time order status updates to customers, ensuring seamless communication and streamlined inventory management processes.

Project Flow

Milestone 1 : Creation of developer account
Milestone 2 : Object Creation
Milestone 3 : Tabs
Milestone 4 : The Lightning App
Milestone 5 : Fields & Relationships
Milestone 6 : Apex Triggers
Milestone 7 : Email Alert
Milestone 8 : Flow
Milestone 9 : Conclusion

Implementation

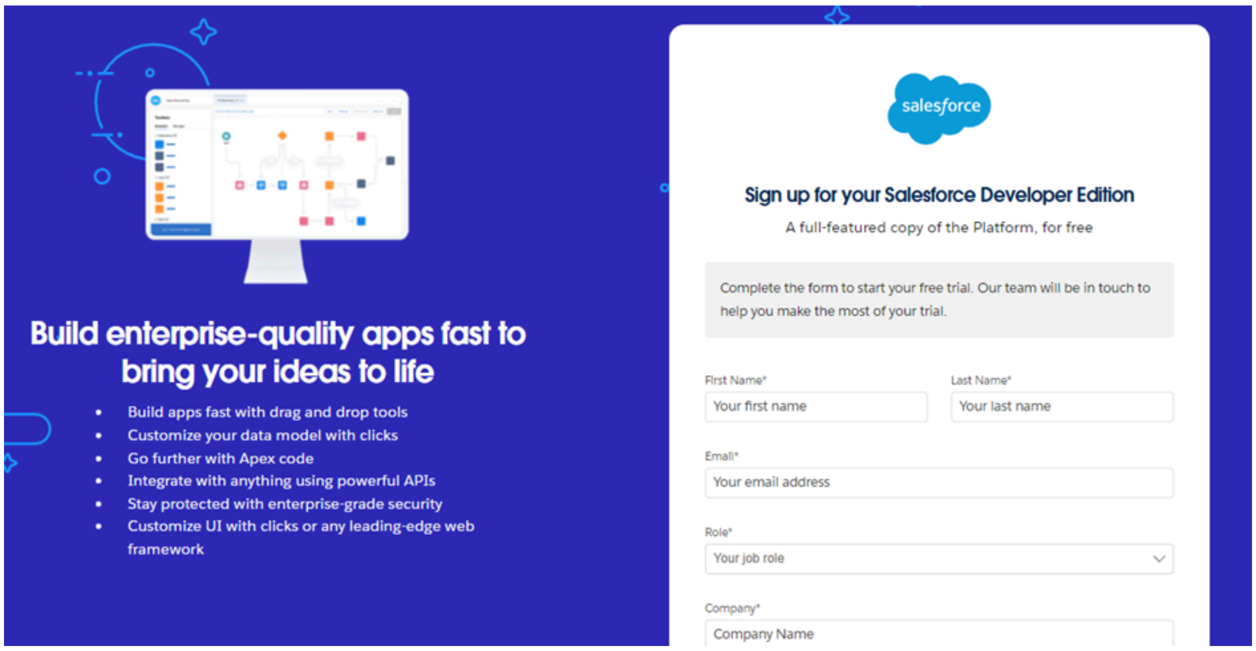
Milestone 1 - Salesforce developer account creation :

Activity 1: Creating Developer Account:

Creating a developer org in salesforce.

1.Go to https://developer.salesforce.com/signup

2.On the sign up form, enter the following details :



Build enterprise-quality apps fast to bring your ideas to life

- Build apps fast with drag and drop tools
- Customize your data model with clicks
- Go further with Apex code
- Integrate with anything using powerful APIs
- Stay protected with enterprise-grade security
- Customize UI with clicks or any leading-edge web framework

Sign up for your Salesforce Developer Edition

A full-featured copy of the Platform, for free

Complete the form to start your free trial. Our team will be in touch to help you make the most of your trial.

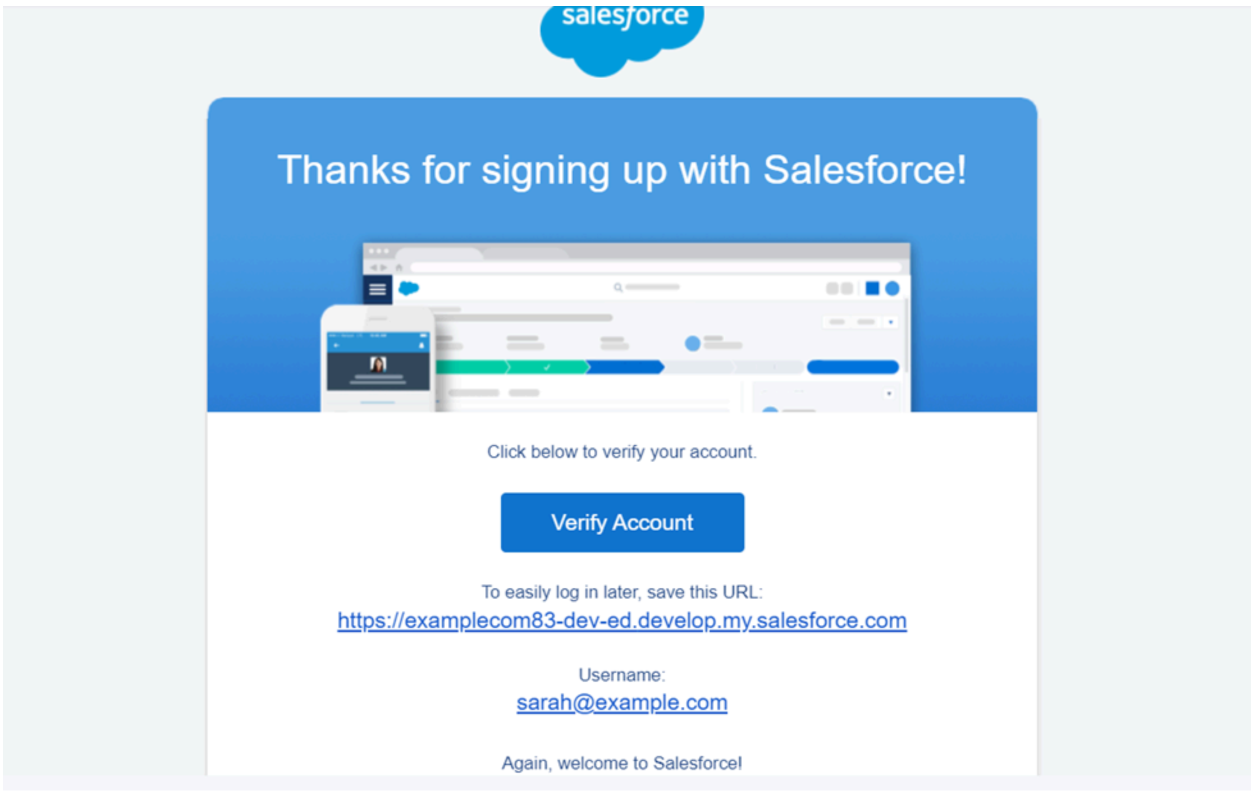
First Name* Your first name
Last Name* Your last name
Email* Your email address
Role* Your job role
Company* Company Name

First name & Last name
1) Email
2) Role : Developer
3) Company : College or Company Name
4) County : India
5) Postal Code : pin code
6) Username : should be a combination of your name and company

This need not be an actual email id, you can give anything in that format : username@organization.com
Click on sign me up after filling these.

Activity 2: Account Activation :

1.Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.
2.Click on Verify Account.
3. Give a password and answer a security question and click on change password.



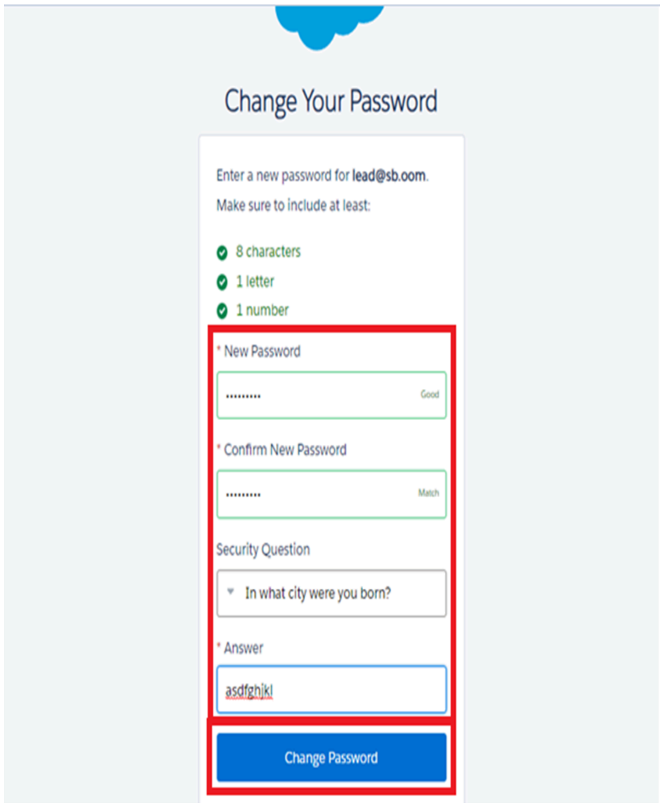
Thanks for signing up with Salesforce!

Click below to verify your account.

To easily log in later, save this URL:
https://examplecom83-dev-ed.develop.my.salesforce.com

Username:
sarah@example.com

Again, welcome to Salesforce!



Change Your Password

Enter a new password for lead@sb.oom.

Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

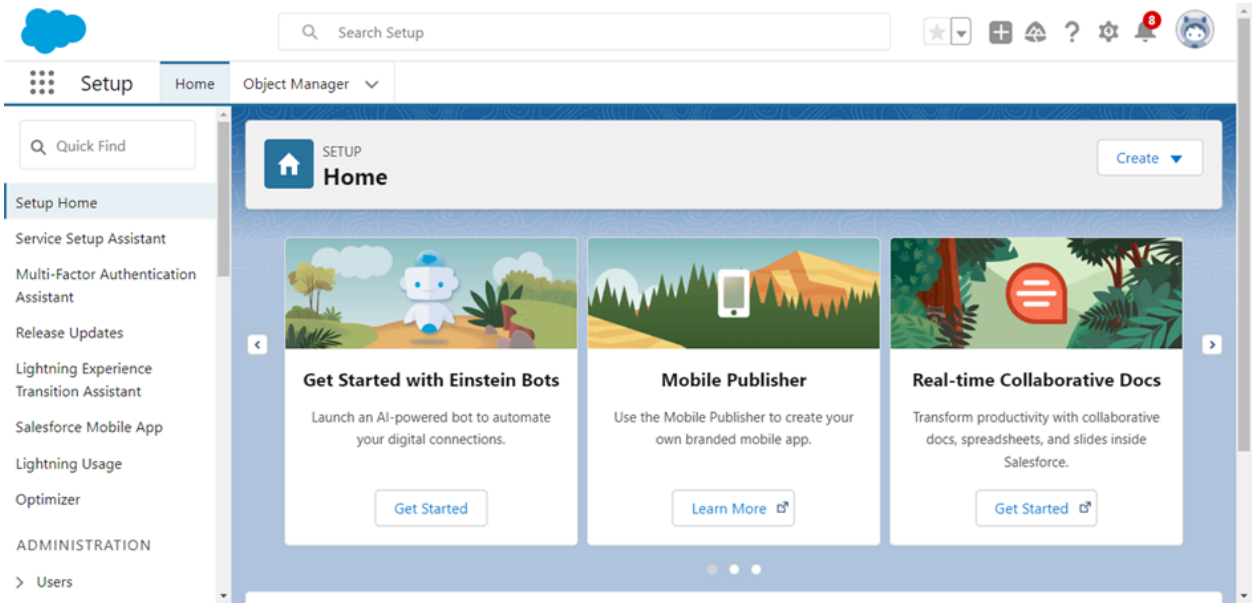
New Password: ••••••••• Good

Confirm New Password: ••••••••• Match

Security Question: In what city were you born?

Answer: asdfghjkl

4.Then you will redirect to your salesforce setup page.

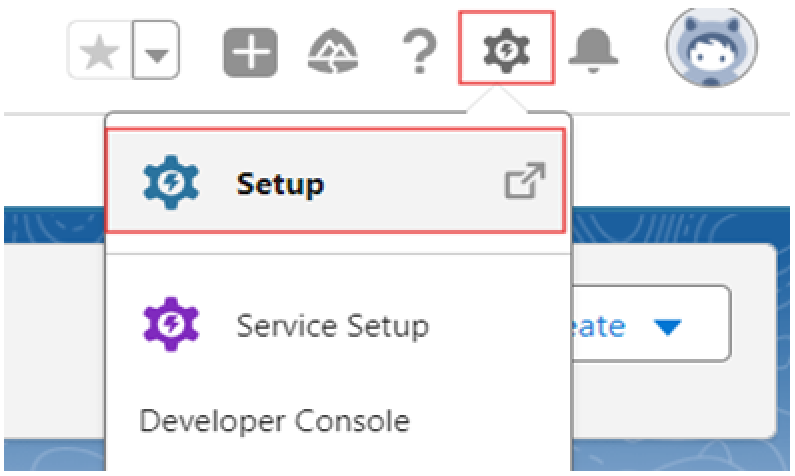


Search Setup
Setup
Home
Object Manager
Quick Find
Setup Home
Service Setup Assistant
Multi-Factor Authentication Assistant
Release Updates
Lightning Experience Transition Assistant
Salesforce Mobile App
Lightning Usage
Optimizer
ADMINISTRATION
Users
SETUP
Home
Create
Get Started with Einstein Bots
Launch an AI-powered bot to automate your digital connections.
Get Started
Mobile Publisher
Use the Mobile Publisher to create your own branded mobile app.
Learn More
Real-time Collaborative Docs
Transform productivity with collaborative docs, spreadsheets, and slides inside Salesforce.
Get Started

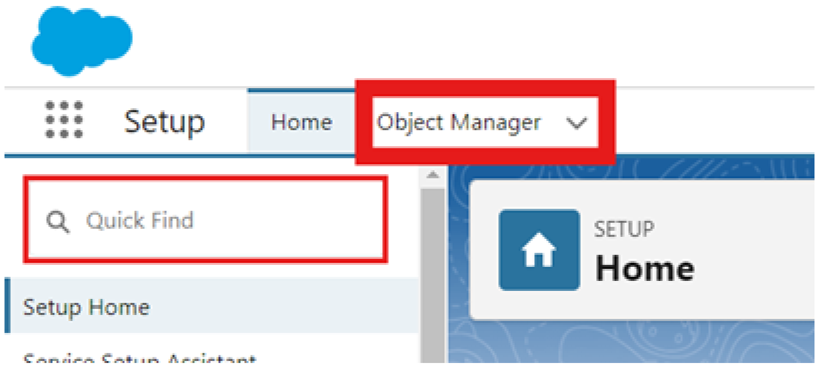
Milestone 2 - Object Creation

Activity 1 :Customer contact details

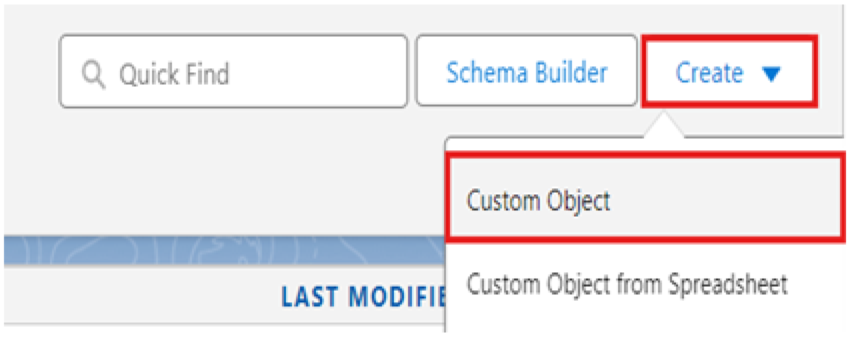
1.In your Salesforce org, click gear icon on the top left and select Setup to open Setup.



2.Click the Object Manager tab. If you don't see it, enter Object Manager in the Quick Find box.



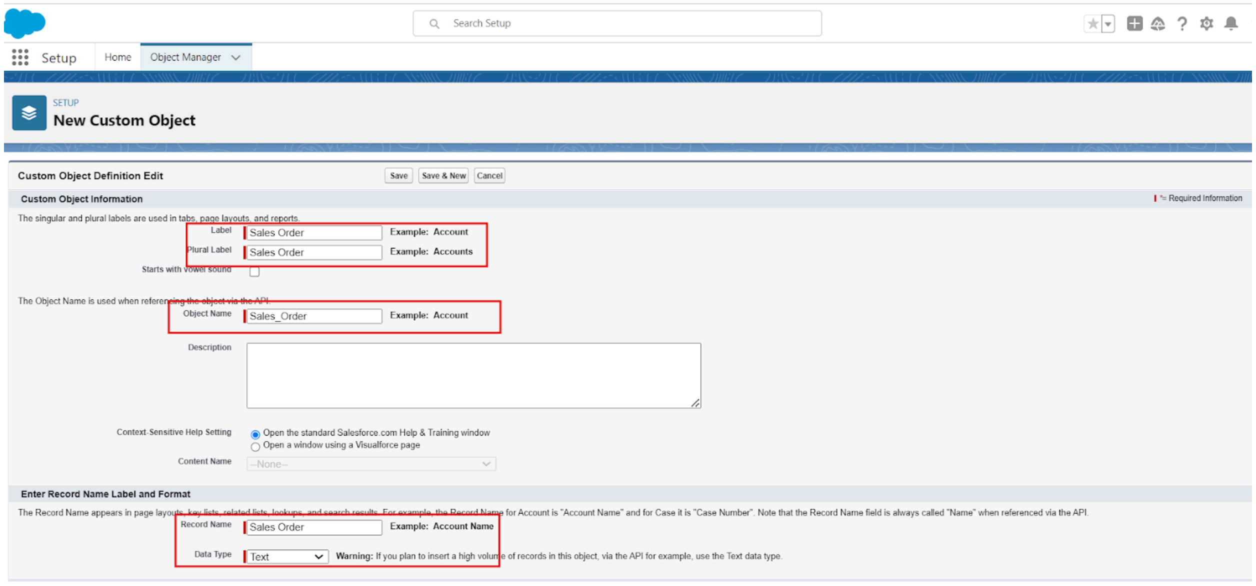
3.On the Object Manager page, click Create | Custom Object.



Activity 1 : Sales Order

1.From the setup page → Click on Object Manager → Click on Create → Click on Custom Object.

- **Label** : Sales Order
- **Plural Label** : Sales Orders
- Enter Record Name Label and Format
- Record Name : Sales Order
- Data Type : Text
- In Optional features : check the boxes for Allow Reports | Allow Activities | Track Field History.
- Search Status : check the box for Allow Search.
- Save



New Custom Object

Custom Object Definition Edit [Save] [Save & New] [Cancel]

Custom Object Information

= Required Information

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

Label	Sales Order	Example: Account
Plural Label	Sales Order	Example: Accounts
Starts with vowel sound	☐	

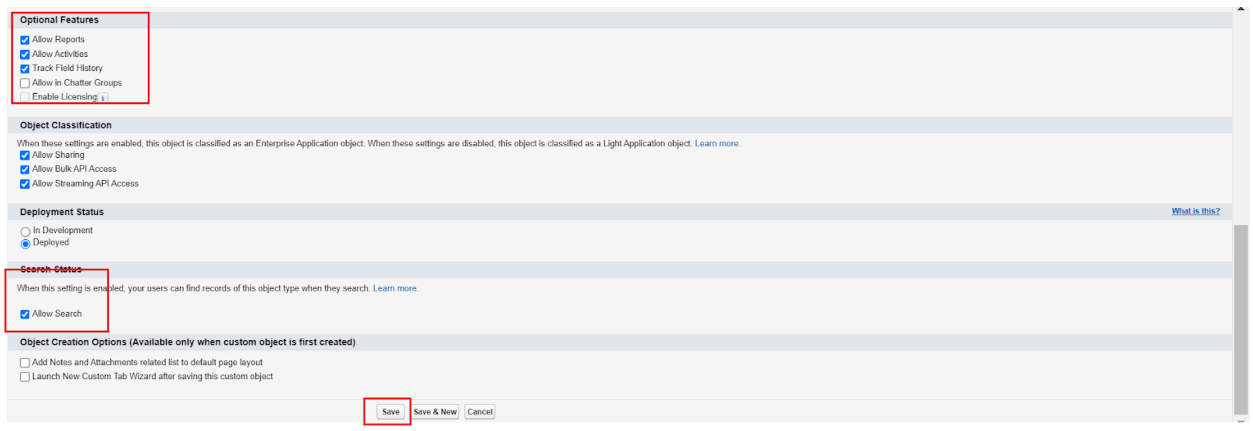
The Object Name is used when referencing the object via the API.

Object Name	Sales_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	--None--	

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	Sales Order	Example: Account Name
Data Type	Text	**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

- [x] Allow Reports
- [x] Allow Activities
- [x] 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.

- [x] Allow Sharing
- [x] Allow Bulk API Access
- [x] Allow Streaming API Access

Deployment Status

What is this?

- ○ In Development
- ● Deployed

Search Status

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

- [x] Allow Search

Object Creation Options (Available only when custom object is first created)

- [ ] Add Notes and Attachments related list to default page layout
- [ ] Launch New Custom Tab Wizard after saving this custom object

Activity 2 : Inventory

2.From the setup page → Click on Object Manager → Click on Create → Click on Custom Object.

- Label : Inventory
- Plural Label : Inventories
- Enter Record Name Label and Format
- Record Name : Inventory Name
- Data Type : Text
- In Optional features : check the boxes for Allow Reports | Allow Activities | Track Field History.
- Search Status : check the box for Allow Search.
- Save.

Activity 3 : Suppliers

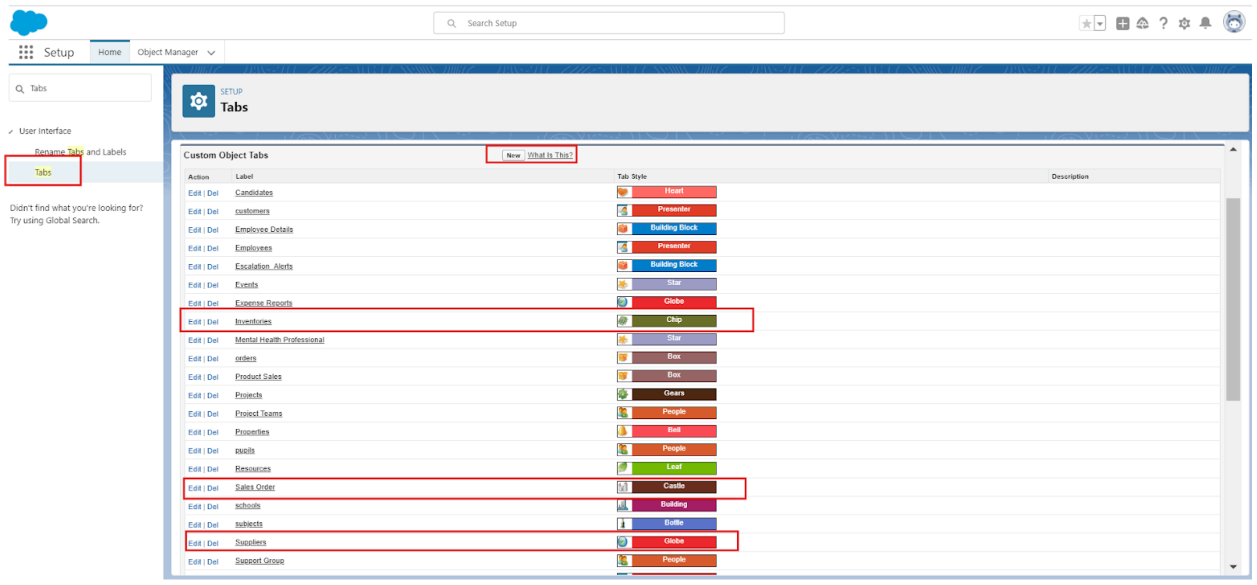
3. From the setup page → Click on Object Manager → Click on Create → Click on Custom Object.

- Label : Supplier
- Plural Label :Suppliers
- Enter Record Name Label and Format
- Record Name : Supplier Name
- Data Type : Text
- In Optional features : check the boxes for Allow Reports | Allow Activities | Track Field History.
- Search Status : check the box for Allow Search.
- Save.

Milestone 3 - Tabs

Activity 1 : Custom tabs creation

1.Go to setup page → type Tabs in Quick Find bar → click on tabs → New (under custom object tab)
2.Select Object(Sales order) → Select the tab style → Next (Add to profiles page) keep it as default → Next (Add to Custom App) uncheck the include tab .
3.Make sure that the Append tab to users' existing personal customizations is checked.
4.Click save.
5.Repeat the same steps for all objects.

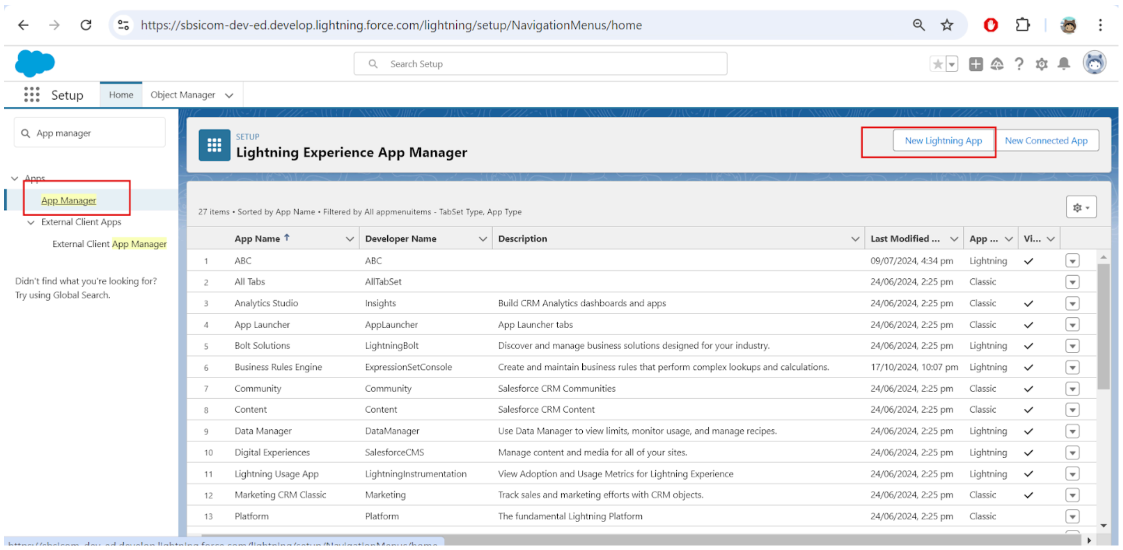


Action	Label	Tab Style	Description
Edit | Del	Candidates	Heart	
Edit | Del	customers	Presenter	
Edit | Del	Employee Details	Building Block	
Edit | Del	Employees	Presenter	
Edit | Del	Escalation_Alerts	Building Block	
Edit | Del	Events	Star	
Edit | Del	Expense Reports	Globe	
Edit | Del	Inventories	Chip	
Edit | Del	Mental Health Professional	Star	
Edit | Del	orders	Box	
Edit | Del	Product Sales	Box	
Edit | Del	Projects	Gears	
Edit | Del	Project Teams	People	
Edit | Del	Properties	Bell	
Edit | Del	pupils	People	
Edit | Del	Resources	Leaf	
Edit | Del	Sales Order	Castle	
Edit | Del	schools	Building	
Edit | Del	subjects	Bottle	
Edit | Del	Suppliers	Globe	
Edit | Del	Support Group	People	

Milestone 4 - The Lightning App

Activity 1 : Create a lightning app

1.Go to setup page → search “app manager” in quick find → select “app manager” → click on New lightning App.


https://sbsicom-dev-ed.develop.lightning.force.com/lightning/setup/NavigationMenus/home

SETUP

Lightning Experience App Manager

27 items • Sorted by App Name • Filtered by All appmenuitems - TabSet Type, App Type

	App Name	Developer Name	Description	Last Modified ...	App ...	Vi...
1	ABC	ABC		09/07/2024, 4:34 pm	Lightning	✓
2	All Tabs	AllTabSet		24/06/2024, 2:25 pm	Classic	
3	Analytics Studio	Insights	Build CRM Analytics dashboards and apps	24/06/2024, 2:25 pm	Classic	✓
4	App Launcher	AppLauncher	App Launcher tabs	24/06/2024, 2:25 pm	Classic	✓
5	Bolt Solutions	LightningBolt	Discover and manage business solutions designed for your industry.	24/06/2024, 2:25 pm	Lightning	✓
6	Business Rules Engine	ExpressionSetConsole	Create and maintain business rules that perform complex lookups and calculations.	17/10/2024, 10:07 pm	Lightning	✓
7	Community	Community	Salesforce CRM Communities	24/06/2024, 2:25 pm	Classic	✓
8	Content	Content	Salesforce CRM Content	24/06/2024, 2:25 pm	Classic	✓
9	Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	24/06/2024, 2:25 pm	Lightning	✓
10	Digital Experiences	SalesforceCMS	Manage content and media for all of your sites.	24/06/2024, 2:25 pm	Lightning	✓
11	Lightning Usage App	LightningInstrumentation	View Adoption and Usage Metrics for Lightning Experience	24/06/2024, 2:25 pm	Lightning	✓
12	Marketing CRM Classic	Marketing	Track sales and marketing efforts with CRM objects.	24/06/2024, 2:25 pm	Classic	✓
13	Platform	Platform	The fundamental Lightning Platform	24/06/2024, 2:25 pm	Classic	

2.Fill the app name in app details and branding as follow

App Name :Smart Inventory Management

Developer Name : Smart_Inventory_Management

Add image optional (if you want to give any image you can otherwise not mandatory) - Add Primary color Hex or leave it to default.

Then click Next.

3. In App options

Navigation style : Standard navigation

Setup experience : setup

Supporters form factors : Desktop and phone

Then click Next

4. In Utility items

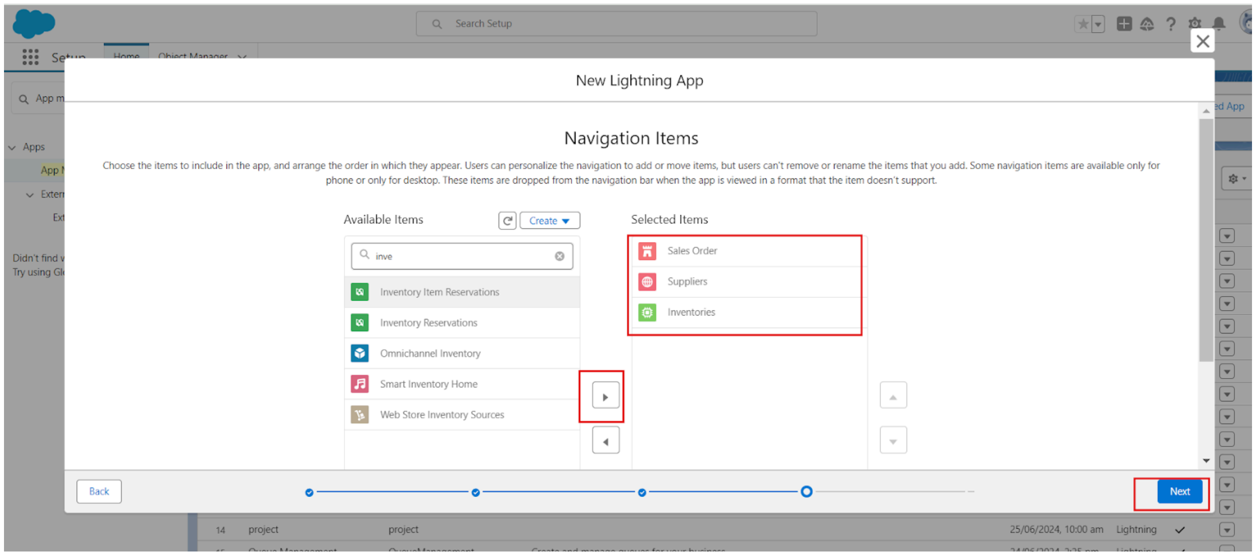
Utility Bar alignment : Default

Then click Next.

5. Navigation items

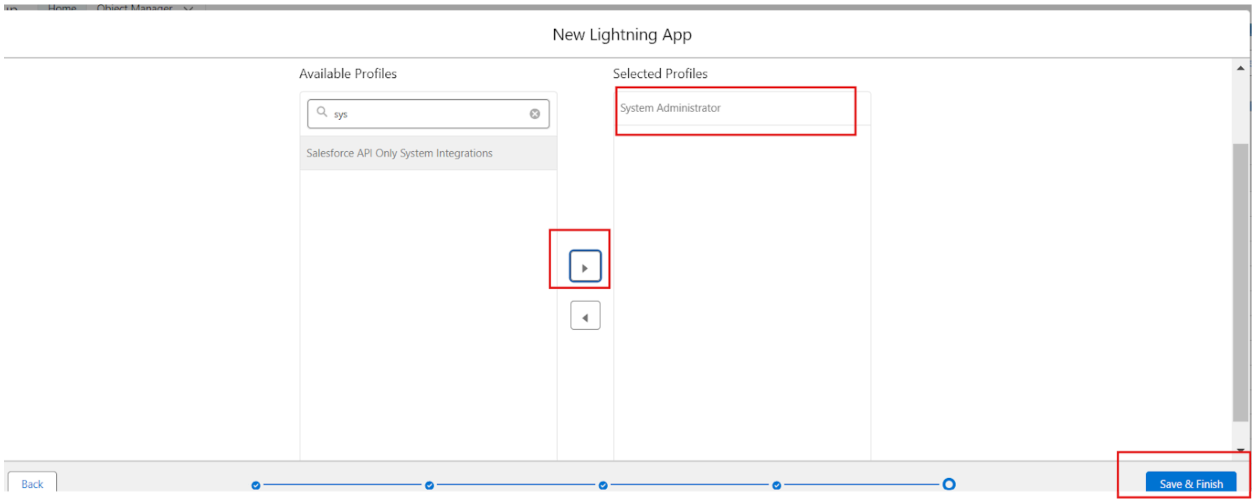
Select the created Custom Objects and required standard objects

- Sales Order
- Inventories
- Supplier


New Lightning App
Navigation Items
Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.
Available Items
Create
inve
Inventory Item Reservations
Inventory Reservations
Omnichannel Inventory
Smart Inventory Home
Web Store Inventory Sources
Selected Items
Sales Order
Suppliers
Inventories
Back
Next

- Click Save

6. To Add User Profiles : System Administrator



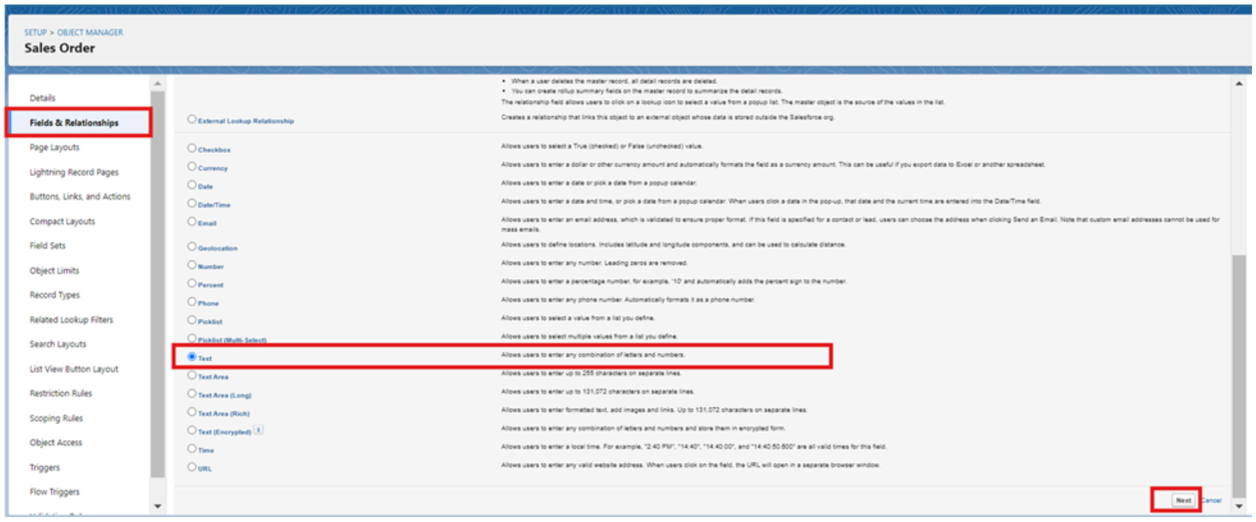
7.Click Save & Finish.

<u>Milestone 5 : Sales Order</u>

Activity 1 : Create Custom fields for Sales Order object

1.In your Salesforce org, click gear icon on the top left and select Setup to open Setup.
2.Click the Object Manager tab. If you don't see it, enter Object Manager in the Quick Find box.

3.From the object manager page, In the Quick Find box, Search for the custom object you just created : Sales Order
4.From the sidebar, click Fields & Relationships. Notice that there are already some fields there. Those are the standard fields.
5.Click New to create a custom field. Tip: Before creating a new field, do a quick search to make sure a similar one doesn't already exist.
6.Next, choose a data type as Text. Choosing a data type helps you format the field input.



- When a user deletes the master record, all detail records are deleted.
- You can create rollup summary fields on the master record to summarize the detail records.

The relationship field allows users to click on a lookup icon to select a value from a popup list. The master object is the source of the values in the list.

External Lookup Relationship	Creates a relationship that links this object to an external object whose data is stored outside the Salesforce org.
Checkbox	Allows users to select a True (checked) or False (unchecked) value.
Currency	Allows users to enter a dollar or other currency amount and automatically formats the field as a currency amount. This can be useful if you export data to Excel or another spreadsheet.
Date	Allows users to enter a date or pick a date from a popup calendar.
Date/Time	Allows users to enter a date and time, or pick a date from a popup calendar. When users click a date in the pop-up, that date and the current time are entered into the Date/Time field.
Email	Allows users to enter an email address, which is validated to ensure proper format. If this field is specified for a contact or lead, users can choose the address when clicking Send an Email. Note that custom email addresses cannot be used for mass emails.
Geolocation	Allows users to define locations. Includes latitude and longitude components, and can be used to calculate distance.
Number	Allows users to enter any number. Leading zeros are removed.
Percent	Allows users to enter a percentage number, for example, '10' and automatically adds the percent sign to the number.
Phone	Allows users to enter any phone number. Automatically formats it as a phone number.
Picklist	Allows users to select a value from a list you define.
Picklist (Multi-Select)	Allows users to select multiple values from a list you define.
[✓ Text]	Allows users to enter any combination of letters and numbers.
Text Area	Allows users to enter up to 255 characters on separate lines.
Text Area (Long)	Allows users to enter up to 131,072 characters on separate lines.
Text Area (Rich)	Allows users to enter formatted text, add images and links. Up to 131,072 characters on separate lines.
Text (Encrypted)	Allows users to enter any combination of letters and numbers and store them in encrypted form.
Time	Allows users to enter a local time. For example, "2:40 PM", "14:40", "14:40:00", and "14:40:50.600" are all valid times for this field.
URL	Allows users to enter any valid website address. When users click on the field, the URL will open in a separate browser window.

7. Click on next.

8. Data Type : Text
   Field Label: Customer Name
   Field Name : Name
   Length : 50
   Click on Next → Next → Save and new.

1. Repeat the Same steps for remaining fields

2. Data Type : Phone
   Field Label: Customer Mobile
   Field Name : Customer_Mobile__c
   Length : 20
   Click on Next → Next → Save and new.

3. Data Type : E-mail
   Field Label : Customer Email
   Field Name : Customer_Email__c
   Click on Next → Next → Save and new.

4. Data Type : Auto Number
   Field Label : Order Number
   Field Name : Order_Name_c
5. Data Type : Number
   Field Label : Quantity Ordered
   Field Name : Quantity_Ordered__c
   Click on Next → Next → Save and new.

6. Data Type : Date
   Field Label : Order Date
   Field Name : Order_Date__c
   Click on Next → Next → Save and new.

7. Data Type : Picklist
   Field Label : Status
   Field Name : Status_c
   Values : Confirmed ,Shipped ,Completed , Canceled
   Click on Next → Next → Save and new.

8. Data Type : Lookup
   Related To : Inventory
   Field Label : Product
   Field Name : Product__c
   Click on Next → Next → Save and new.

Activity 2 : Create Custom fields for Inventory object

1.Data Type : Look-up Relationship
Related To :Supplier
Field Label: Supplier
Field Name : Supplier_c
Length : 80
Click on Next → Next → Save and new.

2.Data Type : Text
Field Label: Inventory Name
Field Name : Name
Click on Next → Next → Save and new.

3.Data Type : Number
Field Label: Reorder Level
Field Name : Reorder_Level_c
Click on Next → Next → Save and new.

4.Data Type : Number
Field Label: Stock Quantity
Field Name : Stock_Qunatity_c
Click on Next → Next → Save and new.

Activity 3 : Create Custom fields for Supplier Object

1.Data Type : Text
  Field Label: Supplier Name
  Field Name : Name
    Length : 50
    Click on Next → Next → Save and new.

2.Data Type : Email
Field Label: Contact Info
Field Name : Contact_Info__c
Length : 80
Click on Next → Next → Save and new.

<u>Milestone 6 :Triggers</u>

Activity 1 : Create a Trigger for sending reorder notifications to suppliers when stock falls below the reorder level.

Step 1: Login to Salesforce

- **Log in** to your Salesforce account with administrative privileges.

Step 2: Navigate to Developer Console

i) **Navigate to Setup**: Click on the gear icon ⚙ (Setup) at the top-right corner of the screen.
ii) **Open Developer Console**: Select the "Developer Console" option from the Setup menu. The Developer Console will open in a new browser tab or window.

Step 3: Create the Apex Trigger

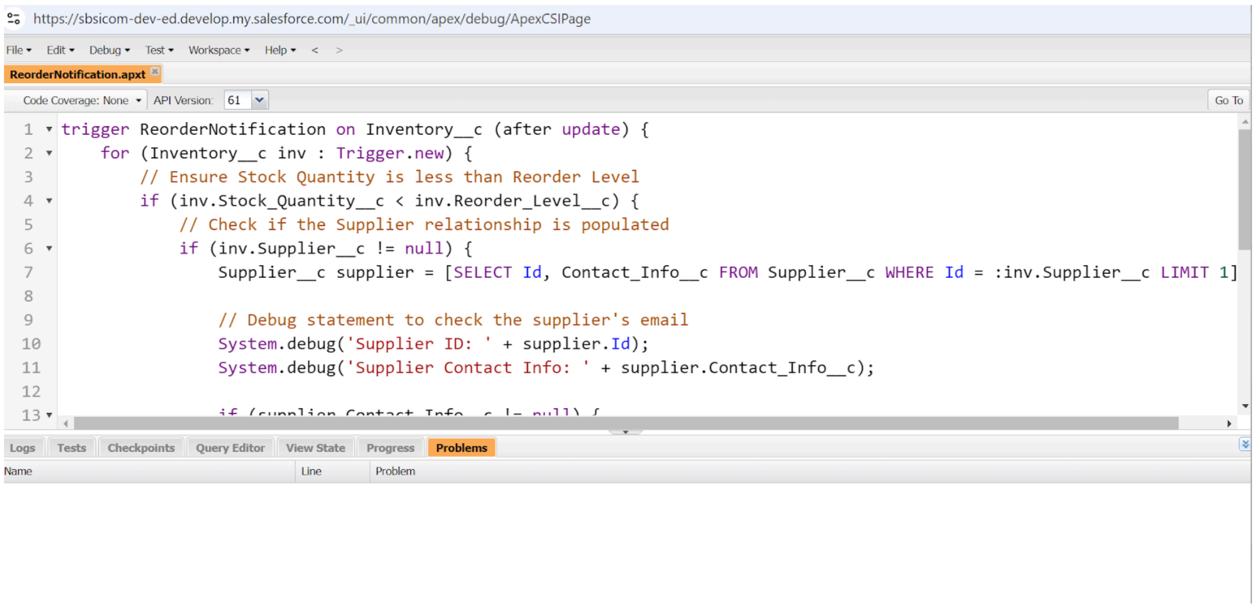
1. In the Developer Console window, click on "File" in the top menu.
2. **Select New**: From the dropdown menu under "File," select "New."
3. **Choose Apex Trigger**: In the submenu, select "Apex Trigger." This opens a new editor tab for writing your trigger.

Step 4: Define Your Trigger

- **Give Trigger Name**: ReorderNotification
- **Select Object**: Inventory_c

```
 trigger ReorderNotification on Inventory__c (after update) {
   for (Inventory__c inv : Trigger.new) {
       // Ensure Stock Quantity is less than Reorder Level
       if (inv.Stock_Quantity__c < inv.Reorder_Level__c) {
           // Check if the Supplier relationship is populated
           if (inv.Supplier__c != null) {
                Supplier__c supplier = [SELECT Id, Contact_Info__c FROM Supplier__c WHERE Id
= :inv.Supplier__c LIMIT 1];

               // Debug statement to check the supplier's email
               System.debug('Supplier ID: ' + supplier.Id);
               System.debug('Supplier Contact Info: ' + supplier.Contact_Info__c);

               if (supplier.Contact_Info__c != null) {
                   Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
                   mail.setToAddresses(new String[] { supplier.Contact_Info__c });
                   mail.setSubject('Reorder Notification');
                   mail.setPlainTextBody('Please reorder product: ' + inv.Name +
                                         '. Current stock is: ' + inv.Stock_Quantity__c);
                   Messaging.sendEmail(new Messaging.SingleEmailMessage[] { mail });
               } else {
                   System.debug('No valid email found for supplier: ' + inv.Supplier__c);
               }
           } else {
               System.debug('Supplier relationship is null for Inventory: ' + inv.Id);
```

```
            }
        }
    }
}
```



```
trigger ReorderNotification on Inventory__c (after update) {
    for (Inventory__c inv : Trigger.new) {
        // Ensure Stock Quantity is less than Reorder Level
        if (inv.Stock_Quantity__c < inv.Reorder_Level__c) {
            // Check if the Supplier relationship is populated
            if (inv.Supplier__c != null) {
                Supplier__c supplier = [SELECT Id, Contact_Info__c FROM Supplier__c WHERE Id = :inv.Supplier__c LIMIT 1]

                // Debug statement to check the supplier's email
                System.debug('Supplier ID: ' + supplier.Id);
                System.debug('Supplier Contact Info: ' + supplier.Contact_Info__c);

                if (supplier.Contact_Info__c != null) {
```

Activity 2: Create a Trigger to manage inventory stock levels based on sales orders.

Step 1: Create the Apex Trigger

1. In the Developer Console window, click on "File" in the top menu.
2. **Select New**: From the dropdown menu under "File," select "New."
3. **Choose Apex Trigger**: In the submenu, select "Apex Trigger." This opens a new editor tab for writing your trigger.

Step 2: Define Your Trigger

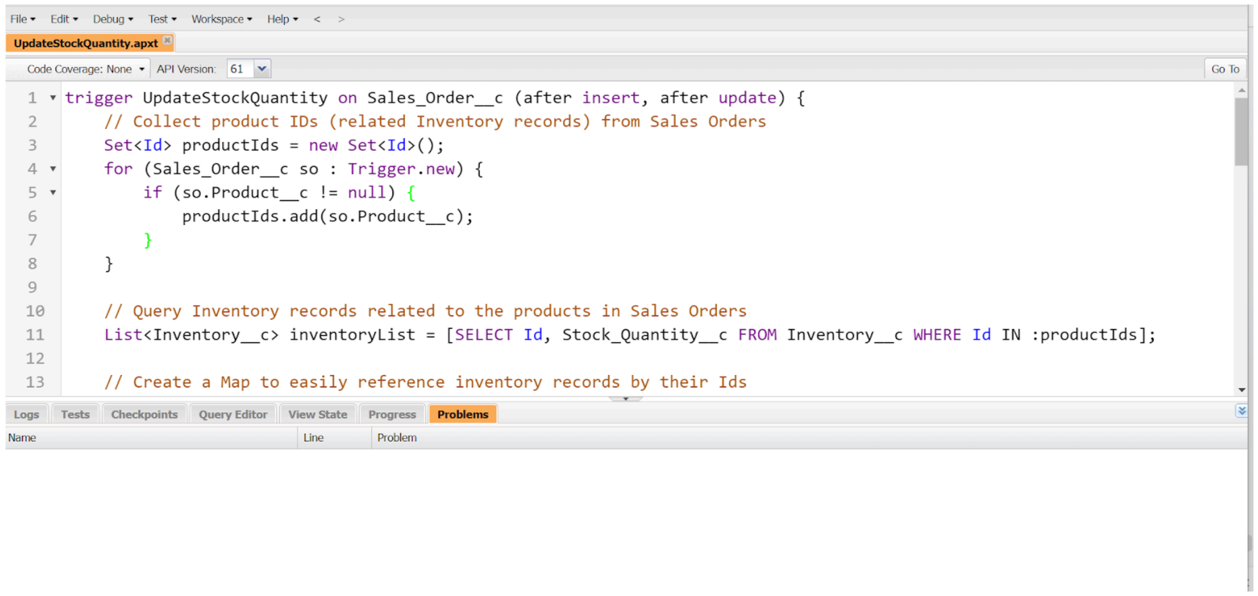
Give Trigger Name: UpdateStockQuantity
Select Object: Sales_Order_c
trigger UpdateStockQuantity on Sales_Order__c (after insert, after update) {

// Collect product IDs (related Inventory records) from Sales Orders

Set<Id> productIds = new Set<Id>();

for (Sales_Order__c so : Trigger.new) {

```
        if (so.Product__c != null) {
            productIds.add(so.Product__c);
        }
    }

    // Query Inventory records related to the products in Sales Orders
        List<Inventory__c> inventoryList = [SELECT Id, Stock_Quantity__c FROM
Inventory__c WHERE Id IN :productIds];

    // Create a Map to easily reference inventory records by their Ids
    Map<Id, Inventory__c> inventoryMap = new Map<Id, Inventory__c>(inventoryList);

    // Loop through each Sales Order to update the respective inventory
    for (Sales_Order__c so : Trigger.new) {
        if (so.Product__c != null && inventoryMap.containsKey(so.Product__c)) {
            Inventory__c inventory = inventoryMap.get(so.Product__c);

            // Log inventory and order details for debugging
            System.debug('Inventory Before Update: ' + inventory.Stock_Quantity__c);
            System.debug('Sales Order Quantity Ordered: ' + so.Quantity_Ordered__c);

            // Check if this is an update operation
            if (Trigger.isUpdate && Trigger.oldMap.containsKey(so.Id)) {
                Decimal previousQuantity = Trigger.oldMap.get(so.Id).Quantity_Ordered__c;
                // Restore the previous stock quantity (add back the previous quantity)
                if (previousQuantity != null) {
                    inventory.Stock_Quantity__c += previousQuantity;
                    System.debug('Restored Stock Quantity: ' + inventory.Stock_Quantity__c);
                }
            }

            // Reduce the stock by the new quantity (for both insert and update)
            Decimal newQuantity = so.Quantity_Ordered__c;
            if (newQuantity != null) {
                inventory.Stock_Quantity__c -= newQuantity;
                System.debug('Updated Stock Quantity: ' + inventory.Stock_Quantity__c);
            }
        } else {
              System.debug('Product not found in Inventory Map for Sales Order Product__c: '
+ so.Product__c);
        }
    }

    // Update inventory records in bulk
    if (!inventoryList.isEmpty()) {
        try {
            update inventoryList;
            System.debug('Inventory records updated successfully.');
        } catch (Exception e) {
            System.debug('Error updating inventory: ' + e.getMessage());
        }
    }
}
```



File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

UpdateStockQuantity.apxt

Code Coverage: None ▾ API Version: 61

```
trigger UpdateStockQuantity on Sales_Order__c (after insert, after update) {
    // Collect product IDs (related Inventory records) from Sales Orders
    Set<Id> productIds = new Set<Id>();
    for (Sales_Order__c so : Trigger.new) {
        if (so.Product__c != null) {
            productIds.add(so.Product__c);
        }
    }

    // Query Inventory records related to the products in Sales Orders
    List<Inventory__c> inventoryList = [SELECT Id, Stock_Quantity__c FROM Inventory__c WHERE Id IN :productIds];

    // Create a Map to easily reference inventory records by their Ids
```

Milestone 7 : Email Alert

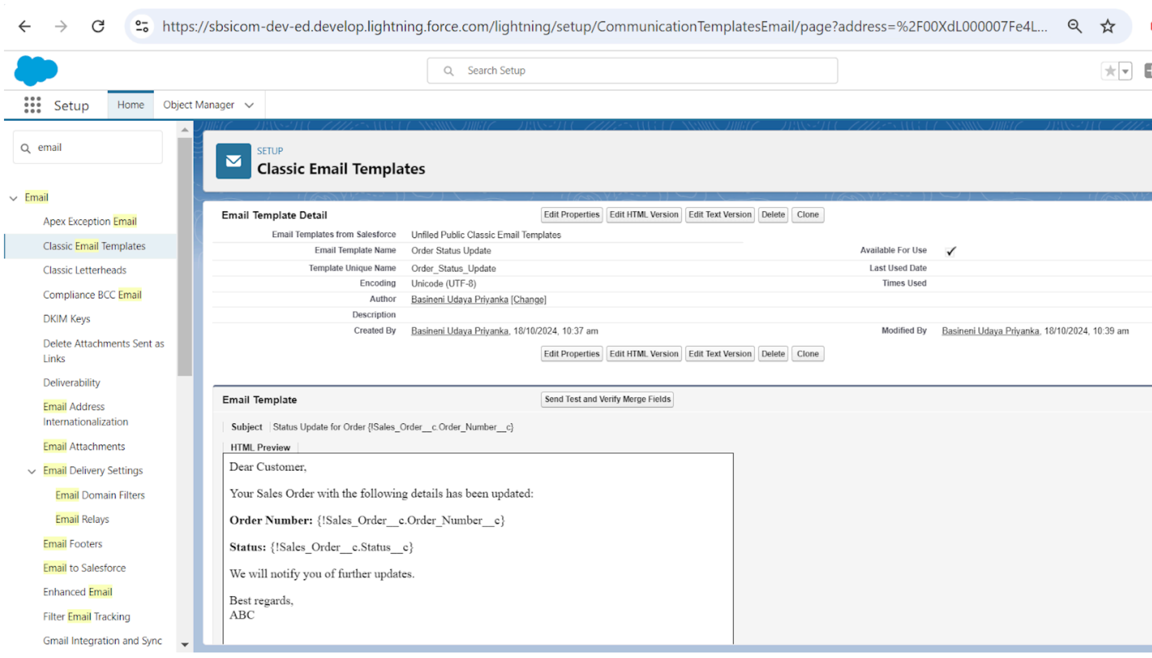
Activity 1 : Creating an Email Template

This step involves designing a custom email template that is used to send notifications to customers regarding the status of their orders. It ensures consistent, professional communication.

Step 1: Create Email Template

1. **Login to Salesforce**: Use administrative credentials to log in.
2. **Navigate to Email Templates**:
    - Go to **Setup**.
    - In the Quick Find box, type **Classic Email Templates**.
    - Click **Classic Email Templates**.
3. **Create New Email Template**:

- Click **New Template**.
- Choose the type of template (e.g., **Custom HTML**, **Text**, or **Visualforce**).
- Enter details like:
  - **Email Template Name**: `Order_Status_Notification`.
  - **Subject**: `Order Status Update for Your Order {!Sales_Order_c.Name}`.
- **Email Body**: Include placeholders for fields such as `{!Sales_Order_c.Order_Status_c}`, `{!Sales_Order_c.Quantity_Ordered_c}`, `{!Sales_Order_c.Product_c}`.



https://sbsicom-dev-ed.develop.lightning.force.com/lightning/setup/CommunicationTemplatesEmail/page?address=%2F00XdL000007Fe4L...

SETUP

Classic Email Templates

Email Template Detail

Email Templates from Salesforce	Unfiled Public Classic Email Templates		
Email Template Name	Order Status Update	Available For Use	✓
Template Unique Name	Order_Status_Update	Last Used Date	
Encoding	Unicode (UTF-8)	Times Used	
Author	Basineni Udaya Priyanka [Change]		
Description			
Created By	Basineni Udaya Priyanka, 18/10/2024, 10:37 am	Modified By	Basineni Udaya Priyanka, 18/10/2024, 10:39 am

Email Template

Subject: Status Update for Order {!Sales_Order__c.Order_Number__c}

HTML Preview

Dear Customer,

Your Sales Order with the following details has been updated:

Order Number: {!Sales_Order__c.Order_Number__c}

Status: {!Sales_Order__c.Status__c}

We will notify you of further updates.

Best regards,
ABC

4. **Save the Template**.

Activity 2 : Configuring an Email Alert

Setting up an email alert links the template to the Sales Order object, automating the sending of order status notifications based on specific triggers or conditions

Step 1 : Create Email Alert

1. **Navigate to Email Alerts**:
   - In **Setup**, search for **Email Alerts**.

- Click **New Email Alert**.

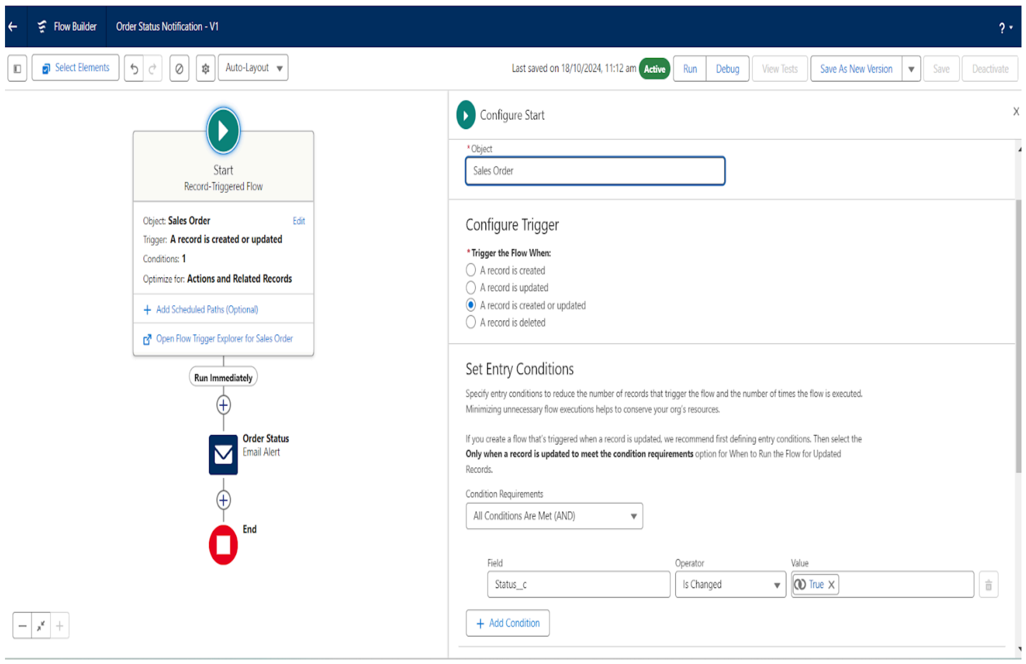
2. **Configure the Email Alert**:
   - **Description**: `Order Status Notification Alert`.
   - **Email Template**: Select the email template you created (`Order_Status_Notification`).
   - **Object**: Choose `Sales_Order_c`.
   - **Recipients**: Choose **Related Contact** and select the field that holds the customer's email, such as `Customer_Email_c` from Sales Order.
   - Click **Save**.

<u>Milestone 8 : Building a Flow</u>

The flow automates the process of sending an email alert whenever a Sales Order's status is updated. It helps streamline communication and ensures customers are informed promptly about their order status.

Step 1: Build the Flow

1. **Navigate to Flows**:
   - In **Setup**, search for **Flows**.
   - Click **Flows** and then **New Flow**.
2. **Select Flow Type**:
   - Choose **Record-Triggered Flow**.
   - Set **Object**: Select **Sales_Order__c**.
   - Set the flow to trigger **after the record is Created and Updated** .
3. **Define Entry Conditions**:
   - Set the condition: **Status__c** is changed True
     Choose **All Conditions are Met (AND)** if you want the flow to run only when all conditions are met.



Flow Builder | Order Status Notification - V1

Last saved on 18/10/2024, 11:12 am | Active

Start
Record-Triggered Flow
Object: Sales Order
Trigger: A record is created or updated
Conditions: 1
Optimize for: Actions and Related Records

Run Immediately → Order Status (Email Alert) → End

Configure Start

*Object: Sales Order

Configure Trigger

*Trigger the Flow When:
- A record is created
- A record is updated
- (●) A record is created or updated
- A record is deleted

Set Entry Conditions

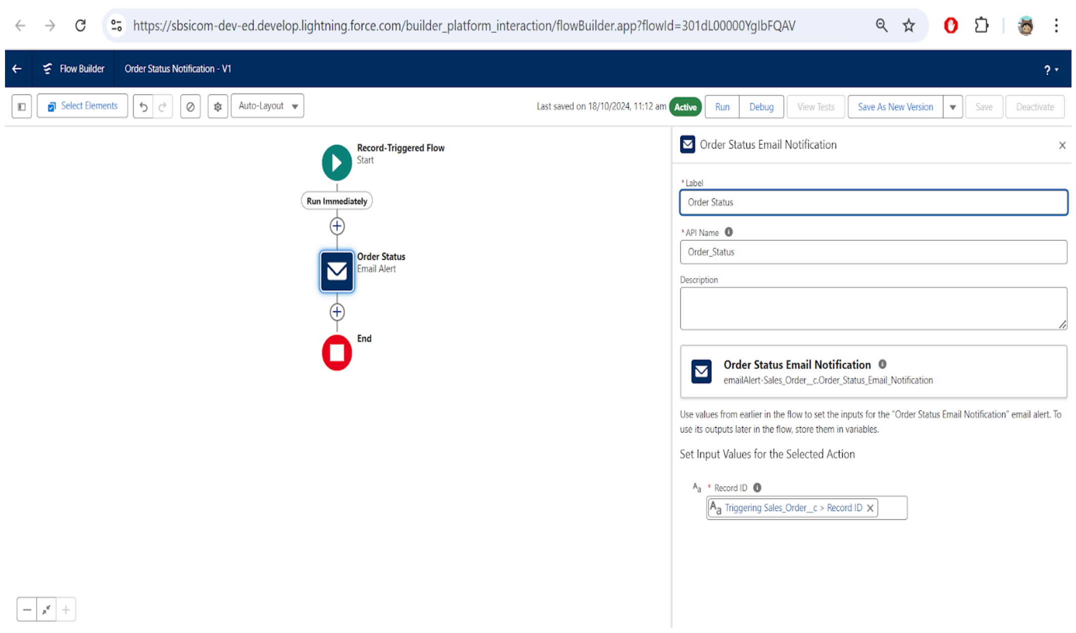
Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

Condition Requirements: All Conditions Are Met (AND)

Field	Operator	Value
Status__c	Is Changed	True

4. **Add Action for Sending Email**:
   - Click the + icon and select **Action**.
   - In the Action type, search for **Send Email Alert**.
   - Choose the **Email Alert** you created earlier (e.g., `Order Status Notification Alert`).



https://sbsicom-dev-ed.develop.lightning.force.com/builder_platform_interaction/flowBuilder.app?flowId=301dL00000YgIbFQAV

Flow Builder | Order Status Notification - V1

Last saved on 18/10/2024, 11:12 am | Active

Record-Triggered Flow (Start) → Run Immediately → Order Status (Email Alert) → End

Order Status Email Notification

*Label: Order Status

*API Name: Order_Status

Description:

Order Status Email Notification
emailAlert-Sales_Order__c.Order_Status_Email_Notification

Use values from earlier in the flow to set the inputs for the "Order Status Email Notification" email alert. To use its outputs later in the flow, store them in variables.

Set Input Values for the Selected Action

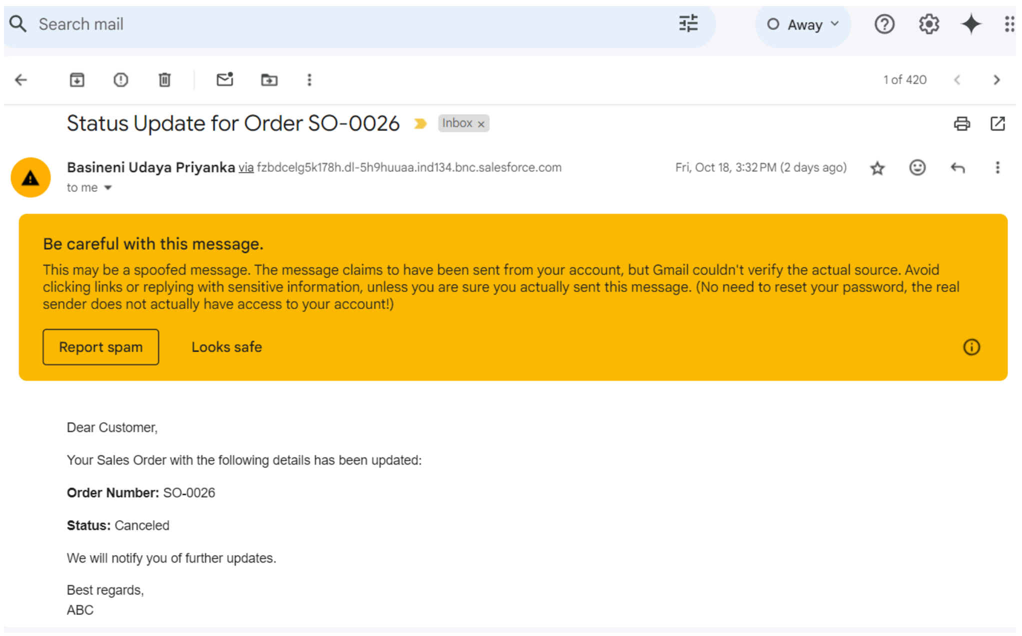
*Record ID: Triggering Sales_Order__c > Record ID

5. **Save and Activate the Flow**:

- Click **Save**, give your flow a meaningful name like `Order Status Notification Flow`.
- Click **Activate** to start the flow.

Milestone 9 : Result

Notifications for status update and reorder :



Status Update for Order SO-0026

Basineni Udaya Priyanka via fzbdcelg5k178h.dl-5h9huuaa.ind134.bnc.salesforce.com
to me
Fri, Oct 18, 3:32 PM (2 days ago)

Be careful with this message.
This may be a spoofed message. The message claims to have been sent from your account, but Gmail couldn't verify the actual source. Avoid clicking links or replying with sensitive information, unless you are sure you actually sent this message. (No need to reset your password, the real sender does not actually have access to your account!)

Dear Customer,

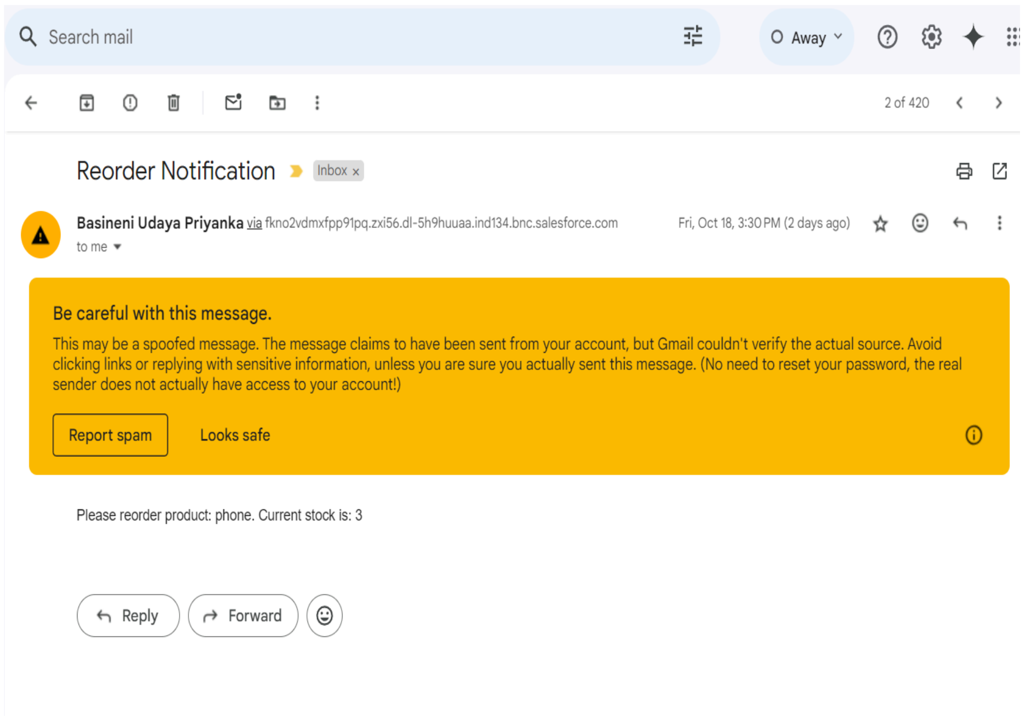
Your Sales Order with the following details has been updated:

Order Number: SO-0026

Status: Canceled

We will notify you of further updates.

Best regards,
ABC



Reorder Notification

Basineni Udaya Priyanka via fkno2vdmxfpp91pq.zxi56.dl-5h9huuaa.ind134.bnc.salesforce.com
to me
Fri, Oct 18, 3:30 PM (2 days ago)

Be careful with this message.
This may be a spoofed message. The message claims to have been sent from your account, but Gmail couldn't verify the actual source. Avoid clicking links or replying with sensitive information, unless you are sure you actually sent this message. (No need to reset your password, the real sender does not actually have access to your account!)

Please reorder product: phone. Current stock is: 3

Milestone 10 : Conclusion

The Smart Inventory Management project utilizing Salesforce with Apex triggers and flows effectively streamlines inventory processes and enhances supplier communication. By automating reorder notifications and stock updates, the system ensures optimal inventory levels are maintained while minimizing the risk of stockouts. The project also facilitates timely order status notifications to customers, improving overall customer satisfaction. This solution exemplifies how Salesforce's robust automation features can transform inventory management, offering a scalable and efficient approach to managing stock, suppliers, and customer interactions in a dynamic retail environment.