

HandsMen Threads — Project Documentation

Elevating the Art of Sophistication in Men's Fashion

Author: Arvin T. De Vera

Date: November 27, 2025

Environment: Salesforce Developer Org / Lightning Experience

Table of Contents

1. Use Case Scenario
2. Project Overview
3. Custom App & Architecture
4. Data Model and Key Fields
5. Creating Data Records & Validation
6. Business Automations (Flows)
7. Apex Development (Triggers & Batch)
8. Email Templates
9. Conclusion

Use Case

I. Description

Developing a premium fashion platform to streamline men's bespoke tailoring and enhance customer experience through personalized styling and seamless order management.

II. User Scenario

- a. HandsMen Threads, a dynamic organization in the fashion industry, is embarking on a Salesforce project designed to revolutionize their data management and enhance customer relations. The project involves building a robust data model tailored to store all pertinent business data, ensuring a seamless flow of information across the organization.

A key aspect of this project is the maintenance of data integrity directly from the user interface (UI). This feature will safeguard the accuracy and consistency of the data, which is crucial for informed decision-making and reliable business operations. The project will integrate several new processes into the business workflow to improve customer service and operational efficiency:

- i. **Automated Order Confirmations:** Post-order confirmation, customers will receive an email update, fostering engagement and strengthening customer relations.
- ii. **Dynamic Loyalty Program:** Customer loyalty statuses will be updated based on purchase history, enabling personalized rewards and promoting repeat business.
- iii. **Proactive Stock Alerts:** When stock levels drop below five units, automatic emails will notify the warehouse team, ensuring timely restocking and preventing stockouts.
- iv. **Scheduled Bulk Order Updates:** Daily midnight, the system will process bulk orders, updating financial records and adjusting inventory, ensuring accurate stock levels for daily operations.

Project Overview

HandsMen Threads is a **custom Salesforce CRM solution** designed to elevate bespoke men's tailoring operations.

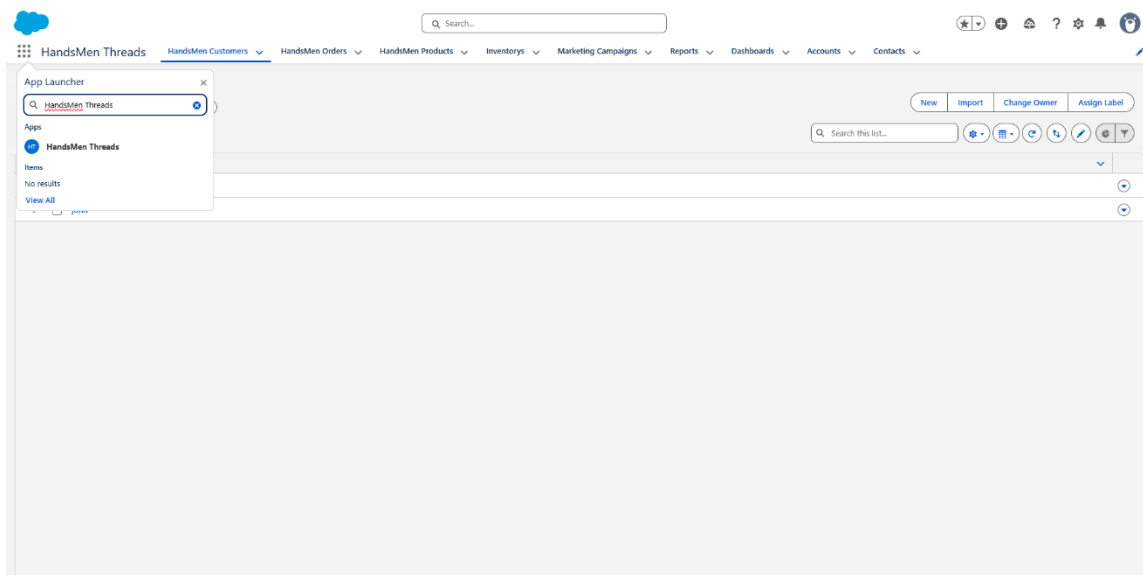
This platform centralizes customer management, loyalty programs, orders, product tracking, inventory deduction, and automated communication workflows.

Key Goals

- Manage customers and track loyalty tiers (Gold, Silver, Bronze).
- Maintain accurate inventory with automated stock deduction.
- Automate communication via Flow email alerts.
- Implement complex business logic using Apex triggers and Batch Apex.
- Enhance overall tailoring workflow from inquiry to delivery.

Tools Used

- Custom Objects & Tabs
- Lightning App
- Validation Rules
- Record-Triggered Flows & Scheduled Flows
- Apex Triggers & Batch Apex
- Reports & Dashboards



Custom App & Architecture

Name: HandsMen Threads Lightning App

Serves as the operational hub for:

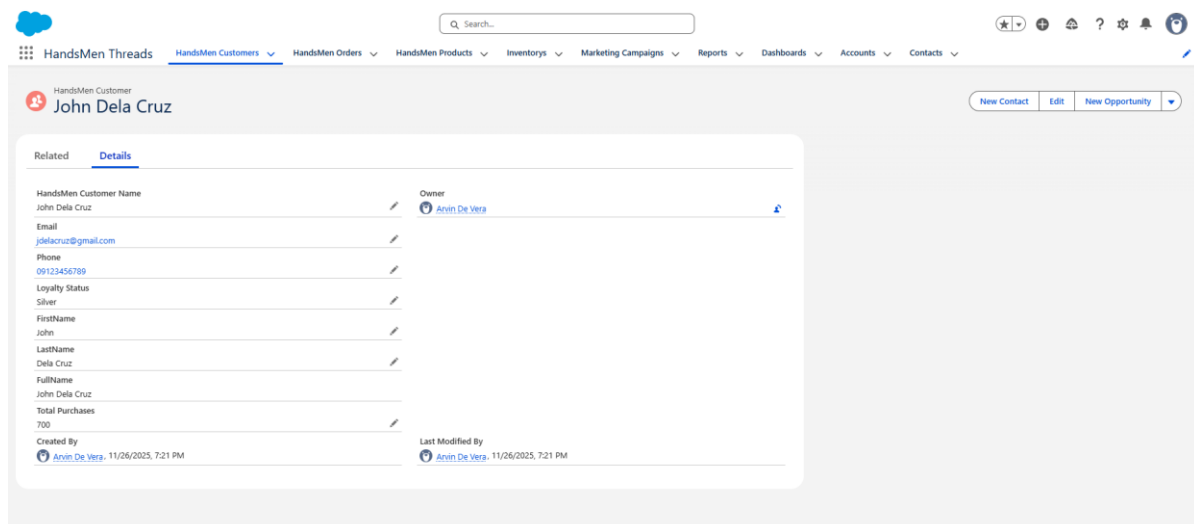
- Sales
- Inventory team
- Admins
- Service team

Custom Objects Implemented

| Object | Description |
|----------------------------|---|
| HandsMen Customers | Stores customer details, Gmail-validated email, loyalty status & purchase history |
| HandsMen Orders | Stores orders, quantities, statuses, total calculation |
| HandsMen Products | Product catalog with Price & SKU |
| Inventory | Tracks stock quantity per product / warehouse |
| Marketing Campaigns | Tracks promotional campaigns & engagement |

Object Relationships

- Customer → Orders
- Orders → Product
- Product → Inventory
- Orders → Inventory (via Apex logic)



HandsMen Threads

HandsMen Customers

HandsMen Orders

HandsMen Products

Inventories

Marketing Campaigns

Reports

Dashboards

Accounts

Contacts

Q Search...

★

+

🏠

?

⚙️

🔔

👤

HandsMen Order

O-0003

New Contact

Edit

New Opportunity

Related

Details

HandsMen OrderNumber

O-0003

HandsMen Product

Shorts

HandsMen Customer

John Dele Cruz

Status

Confirmed

Quantity

600

Total Amount

3,000

Customer Email

devesa000@gmail.com

Created By

Arvin De Vera

Last Modified By

Arvin De Vera

11/26/2023, 7:27 PM

11/26/2023, 7:27 PM

HandsMen Threads

HandsMen Customers

HandsMen Orders

HandsMen Products

Inventories

Marketing Campaigns

Reports

Dashboards

Accounts

Contacts

Q Search...

★

+

🏠

?

⚙️

🔔

👤

HandsMen Product

Shorts

New Contact

Edit

New Opportunity

Related

Details

HandsMen Product Name

Shorts

SKU

0002

Price

\$5

Stock Quantity

2,000

Created By

Arvin De Vera

Last Modified By

Arvin De Vera

11/26/2023, 7:23 PM

11/26/2023, 7:23 PM

HandsMen Threads

HandsMen Customers

HandsMen Orders

HandsMen Products

Inventories

Marketing Campaigns

Reports

Dashboards

Accounts

Contacts

Q Search...

★

+

🏠

?

⚙️

🔔

👤

Inventory

I-0002

New Contact

Edit

New Opportunity

Related

Details

Inventory Number

I-0002

HandsMen Product

Shorts

Stock Quantity

1,500

Stock Status

Available

Warehouse

HandsMen Shorts Warehouse

Created By

Arvin De Vera

Last Modified By

Arvin De Vera

11/26/2023, 7:24 PM

11/26/2023, 7:24 PM

Data Model and Key Fields

HandsMen Customer

- Full Name
- Email (Validation: must be Gmail)
- Phone
- Loyalty_Status__c (*None, Gold, Silver, Bronze*)
- Total_Purchases__c

HandsMen Product

- SKU__c
- Name
- Price__c
- Stock_Quantity__c

Inventory

- HandsMen_Product__c (*Master-Detail*)
- Stock_Quantity__c
- Stock_Status__c
- Warehouse__c

HandsMen Order

- Name (Order Number)
- HandsMen_Customer__c
- HandsMen_Product__c
- Customer_Email__c
- Quantity__c
- Status__c (*Pending, Confirmed, Rejection*)
- Total_Amount__c (*Apex-calculated*)
- Order_Date__c

Creating Data Records & Validation

I. Customer Creation with Validation

a. A validation rule ensures only Gmail accounts are used.

i. Scenario:

1. **Entered Email:** johndc@asdf.com → Error: "Please fill Correct Gmail"
2. **Corrected to:** johndc@gmail.com → Saved successfully.

ii. Sample Data:

1. **Customer:** John Dela Cruz
2. **Loyalty Status:** Silver

The screenshot shows the 'HandsMen Customer Validation Rule' configuration page. The 'Validation Rule Edit' section has 'Rule Name' set to 'Email' and 'Active' checked. The 'Error Condition Formula' section contains the formula: `NOT CONTAINS(Email, '@gmail.com')`. The 'Error Message' section shows an example message: 'Please fill Correct Gmail'. The 'Error Location' is set to 'Top of Page'.

The screenshot shows the 'New HandsMen Customer' form. The 'Information' section contains fields for 'HandsMen Customer Name' (John Dela Cruz), 'Email' (johndc@asdf.com), 'Phone', 'Loyalty Status' (--None--), 'FirstName', 'LastName', and 'Total Purchases'. The 'Owner' field is set to 'Arvin De Vera'. A validation error message is displayed: 'We hit a snag. Review the errors on this page. Please fill Correct Gmail'. The 'Save' button is disabled, and the 'Cancel' button is visible.

HandsMen Customer Validation Rule and Sample data to be tested for validation

Steps: App Launcher → HandsMen Threads → New Handsman Customer → fill fields → Save Demo note: Enter invalid email "johndc@xyz.com" → validation error "Please fill correct Gmail" → change to "johndc@gmail.com" → Save success.

The screenshot shows the 'HandsMen Customer' page for 'John Dela Cruz'. The page has a top navigation bar with a search bar and various menu items. The main content area is divided into 'Related' and 'Details' tabs. The 'Details' tab is active, showing a form with the following fields:

| Field | Value | Action |
|------------------------|---------------------|--------|
| HandsMen Customer Name | John Dela Cruz | Edit |
| Owner | Arvin De Vera | Edit |
| Email | jdelacruz@gmail.com | Edit |
| Phone | 09123456789 | Edit |
| Loyalty Status | Silver | Edit |
| FirstName | John | Edit |
| LastName | Dela Cruz | Edit |
| FullName | John Dela Cruz | Edit |
| Total Purchases | 700 | Edit |
| Created By | Arvin De Vera | Edit |
| Last Modified By | Arvin De Vera | Edit |

The 'Created By' and 'Last Modified By' fields show the user 'Arvin De Vera' and the timestamp '11/26/2025, 7:21 PM'.

II. Product Creation

- HandsMen Product Name: Shorts
- SKU: 0002
- Price: \$5
- Stock Quantity: 2,000

The screenshot shows the 'HandsMen Product' page for 'Shorts'. The page has a top navigation bar with a search bar and various menu items. The main content area is divided into 'Related' and 'Details' tabs. The 'Details' tab is active, showing a form with the following fields:

| Field | Value | Action |
|-----------------------|---------------|--------|
| HandsMen Product Name | Shorts | Edit |
| Owner | Arvin De Vera | Edit |
| SKU | 0002 | Edit |
| Price | \$5 | Edit |
| Stock Quantity | 2,000 | Edit |
| Created By | Arvin De Vera | Edit |
| Last Modified By | Arvin De Vera | Edit |

The 'Created By' and 'Last Modified By' fields show the user 'Arvin De Vera' and the timestamp '11/26/2025, 7:23 PM'.

III. Inventory Setup

- a. **Inventory Number:** I-0002
- b. **HandsMen Product:** Shorts
- c. **Stock Quantity:** 1,500
- d. **Stock Status:** Available
- e. **Warehouse:** HandsMen Shorts Warehouse

The screenshot shows the 'Inventory' page for item 'I-0002' in the 'HandsMen Threads' application. The page has a top navigation bar with a search bar and various menu items. The main content area is divided into 'Related' and 'Details' tabs. The 'Details' tab is active, showing a form with the following fields: 'Inventory Number' (I-0002), 'HandsMen Product' (Shorts), 'Stock Quantity' (1,500), 'Stock Status' (Available), 'Warehouse' (HandsMen Shorts Warehouse), 'Created By' (Arvin De Vera), and 'Last Modified By' (Arvin De Vera). The 'Created By' and 'Last Modified By' fields show the user's name and the timestamp '11/26/2025, 7:24 PM'.

IV. Order Creation

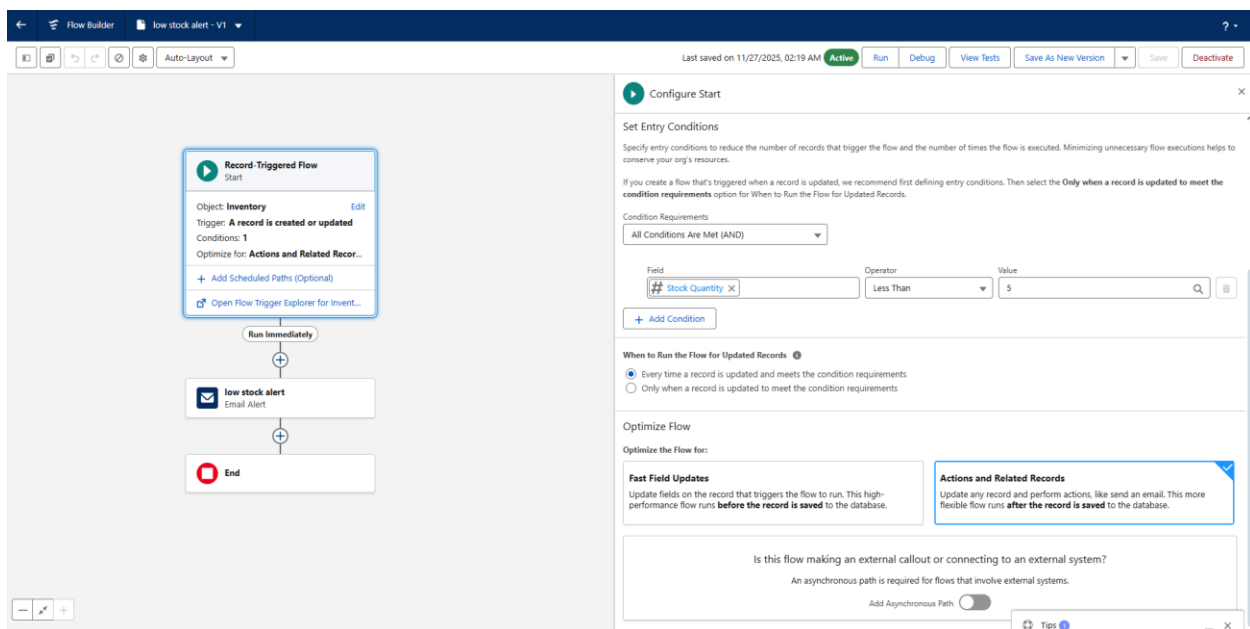
- a. **Order:** O-0003
- b. **HandsMen Product:** Shorts
- c. **HandsMen Customer:** John Dela Cruz
- d. **Status:** Confirmed
- e. **Quantity:** 600
- f. **Total Amount:** 3,000 (Apex calculation)

The screenshot shows the 'HandsMen Order' page for item 'O-0003' in the 'HandsMen Threads' application. The page has a top navigation bar with a search bar and various menu items. The main content area is divided into 'Related' and 'Details' tabs. The 'Details' tab is active, showing a form with the following fields: 'HandsMen OrderNumber' (O-0003), 'Owner' (Arvin De Vera), 'HandsMen Product' (Shorts), 'HandsMen Customer' (John Dela Cruz), 'Status' (Confirmed), 'Quantity' (600), 'Total Amount' (3,000), 'Customer Email' (devera000@gmail.com), 'Created By' (Arvin De Vera), and 'Last Modified By' (Arvin De Vera). The 'Created By' and 'Last Modified By' fields show the user's name and the timestamp '11/26/2025, 7:27 PM'.

Business Automations (Flows)

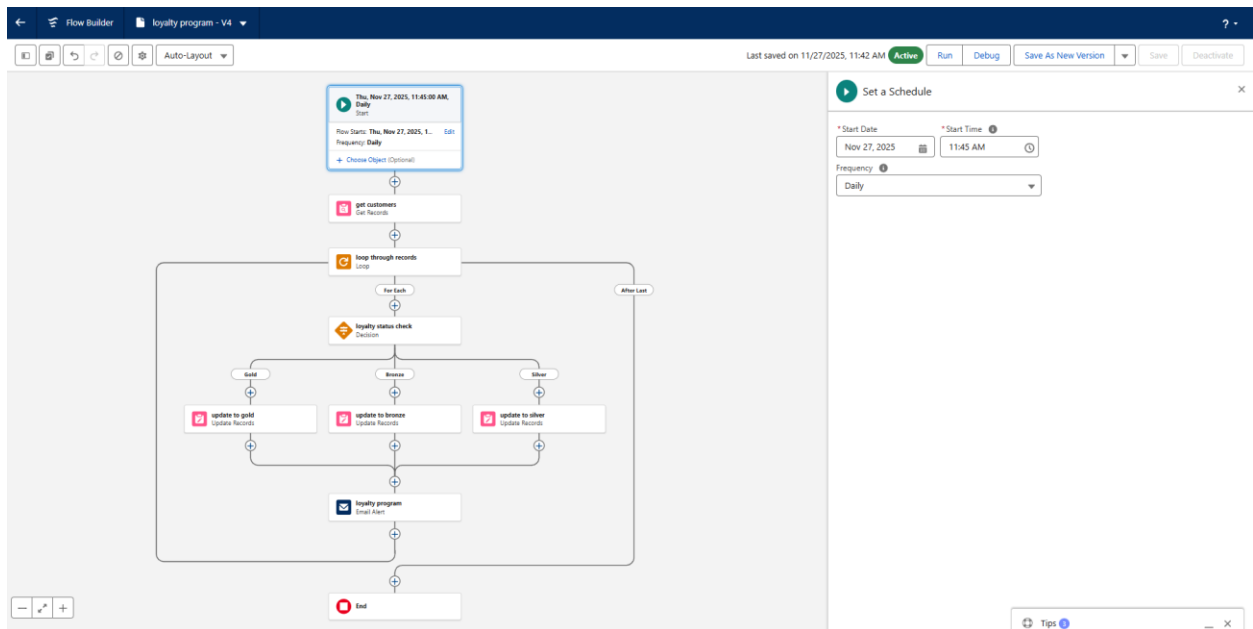
I. Low Stock Alert (Record-Triggered Flow)

- a. **Trigger:** When Inventory is updated.
- b. **Logic:** If Stock Quantity < 5 → send email.
- c. **Email:** “Dear Inventory Manager, This is to inform you that the stock for the following product is running low: Product Name: {!Inventory__c.HandsMen_Product__c} Current Stock Quantity: {!Inventory__c.Stock_Quantity__c} Please take the necessary steps to restock this item immediately. Best Regards, Inventory Monitoring System”



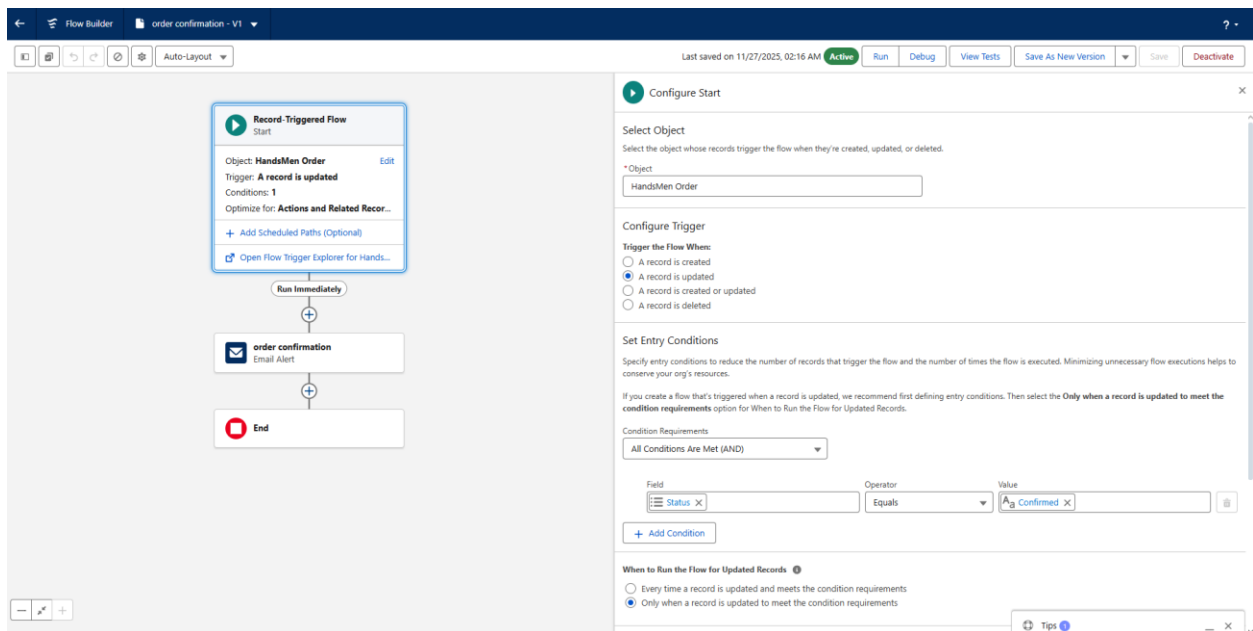
II. Loyalty Program (Scheduled Flow)

- a. **Runs:** Daily at 11:45 AM
- b. **Logic:**
 - i. >1000 → **Gold**
 - ii. 501–1000 → **Silver**
 - iii. ≤500 → **Bronze**
- c. **Email:** “Congratulations! You are now a Gold member and you are eligible for our Loyalty Rewards Program. Enjoy exclusive discounts, early access to offers, and special member benefits. Thank you for your continued Support.”



III. Order Confirmation Flow

- Trigger:** Order Status → Confirmed
- Action:** Send confirmation email.
- Sample Text:** “Dear John, Your order #O-0002 has been confirmed!
Thank you for shopping with us. Best Regards, Sales Team”



Apex Development

I. OrderTotalTrigger — Order Amount Calculation

- a. **Type:** before insert, before update on Handsman Orders
- b. **Logic:** $\text{order.Total_Amount_c} = \text{order.Quantity_c} * \text{product.Price_c}$; $\text{Total_Amount_c} = \text{Product.Price_c} \times \text{Quantity_c}$

```
File • Edit • Debug • Test • Workspace • Help • < >
OrderTotalTrigger.apex • StockInductionTrigger.apex • OrderTrigger.apex • InventoryBatchJob.apex • OrderTriggerHandler.apex
Code Coverage: None • API Version: 65 • Go To

1 • trigger OrderTotalTrigger on HandsMen_Order__c (before insert, before update) {
2   Set<Id> productIds = new Set<Id>();
3
4   for (HandsMen_Order__c order : Trigger.new) {
5     if (order.HandsMen_Product__c != null) {
6       productIds.add(order.HandsMen_Product__c);
7     }
8   }
9
10  Map<Id, HandsMen_Product__c> productMap = new Map<Id, HandsMen_Product__c>{
11    [SELECT Id, Price__c FROM HandsMen_Product__c WHERE Id IN :productIds]
12  };
13
14  for (HandsMen_Order__c order : Trigger.new) {
15    if (order.HandsMen_Product__c != null && productMap.containsKey(order.HandsMen_Product__c)) {
16      HandsMen_Product__c product = productMap.get(order.HandsMen_Product__c);
17      if (order.Quantity__c != null) {
18        order.Total_Amount__c = order.Quantity__c * product.Price__c;
19      }
20    }
21  }
22 }
```

II. OrderTrigger — Quantity Validation Dispatcher

- a. **Type:** before insert, before update
- b. **Logic:**
 - i. **Confirmed** → Quantity > 500
 - ii. **Pending** → Quantity > 200
 - iii. **Rejection** → Quantity = 0

```
File • Edit • Debug • Test • Workspace • Help • < >
OrderTotalTrigger.apex • StockInductionTrigger.apex • OrderTrigger.apex • InventoryBatchJob.apex • OrderTriggerHandler.apex
Code Coverage: None • API Version: 65 • Go To

1 • trigger OrderTrigger on HandsMen_Order__c (before insert, before update) {
2
3   if (Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
4
5     OrderTriggerHandler.validateOrderQuantity(Trigger.new);
6
7   }
8
9 }
```

III. OrderTriggerHandler — Quantity Validation

a. Business Rules:

| Status | Required Quantity |
|-----------|-------------------|
| Confirmed | > 500 |
| Pending | > 200 |

```
File • Edit • Debug • Test • Workspace • Help • < >
OrderTotalTrigger.appt • StockDeductionTrigger.appt • OrderTrigger.appt • InventoryBatchJob.appt • OrderTriggerHandler.appt
Code Coverage: None • API Version: 65 • Go To

1 • public class OrderTriggerHandler {
2 •     public static void validateOrderQuantity(List<HandsMen_Order__c> orderList) {
3 •         for (HandsMen_Order__c order : orderList) {
4 •             if (order.Status__c == 'Confirmed') {
5 •                 if (order.Quantity__c == null || order.Quantity__c <= 500) {
6 •                     order.Quantity__c.addError('For Status "Confirmed", Quantity must be more than 500.');
```

III. StockDeductionTrigger — Inventory Deduction

- a. Type: after insert, after update on Inventory
- b. Logic: inv.Stock_Quantity__c -= order.Quantity__c;

```
File • Edit • Debug • Test • Workspace • Help • < >
OrderTotalTrigger.appt • StockDeductionTrigger.appt • OrderTrigger.appt • InventoryBatchJob.appt • OrderTriggerHandler.appt
Code Coverage: None • API Version: 65 • Go To

1 • trigger StockDeductionTrigger on HandsMen_Order__c (after insert, after update) {
2 •     Set<Id> productIds = new Set<Id>();
3 •
4 •     for (HandsMen_Order__c order : Trigger.new) {
5 •         if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
6 •             productIds.add(order.HandsMen_Product__c);
7 •         }
8 •     }
9 •
10 •     if (productIds.isEmpty()) return;
11 •
12 •     // Query related inventories based on product
13 •     Map<Id, Inventory__c> inventoryMap = new Map<Id, Inventory__c>{
14 •         [SELECT Id, Stock_quantity__c, HandsMen_Product__c
15 •          FROM Inventory__c
16 •          WHERE HandsMen_Product__c IN :productIds]
17 •     };
18 •
19 •     List<Inventory__c> inventoriesToUpdate = new List<Inventory__c>();
20 •
21 •     for (HandsMen_Order__c order : Trigger.new) {
22 •         if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
23 •             for (Inventory__c inv : inventoryMap.values()) {
24 •                 if (inv.HandsMen_Product__c == order.HandsMen_Product__c) {
25 •                     inv.Stock_Quantity__c -= order.Quantity__c;
26 •                     inventoriesToUpdate.add(inv);
27 •                     break;
28 •                 }
29 •             }
30 •         }
31 •     }
32 • }
```

IV. InventoryBatchJob — Batch Apex

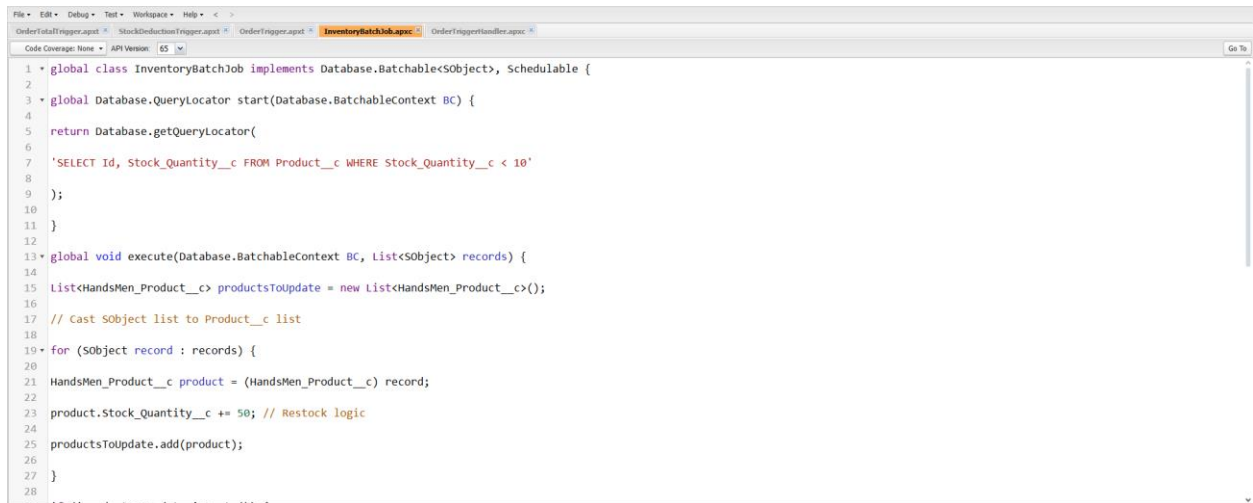
a. Implements:

- i. Database.Batchable
- ii. Schedulable

b. Query: Select products where stock < 18.

c. Purpose:

- i. Bulk restocking
- ii. Reporting



```
1 • global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {
2
3 • global Database.QueryLocator start(Database.BatchableContext BC) {
4
5     return Database.getQueryLocator(
6
7     'SELECT Id, Stock_Quantity__c FROM Product__c WHERE Stock_Quantity__c < 18'
8
9     );
10
11 }
12
13 • global void execute(Database.BatchableContext BC, List<SObject> records) {
14
15     List<HandsMen_Product__c> productsToUpdate = new List<HandsMen_Product__c>();
16
17     // Cast SObject list to Product__c list
18
19 • for (SObject record : records) {
20
21     HandsMen_Product__c product = (HandsMen_Product__c) record;
22
23     product.Stock_Quantity__c += 50; // Restock logic
24
25     productsToUpdate.add(product);
26
27 }
28 }
```

Email Templates

I. Low Stock Alert

- a. **Subject:** Current Stock Quantity: {!Inventory__c.Stock_Quantity__c}
- b. **Body sample:** Dear Inventory Manager, This is to inform you that the stock for the following product is running low: Product Name: {!Inventory__c.HandsMen_Product__c} Current Stock Quantity: {!Inventory__c.Stock_Quantity__c} Please take the necessary steps to restock this item immediately. Best Regards, Inventory Monitoring System

Text Email Template

Low Stock Alert

Help for this Page

Preview your email template below

Email Template Detail

Email Templates from Salesforce

Unified Public Classic Email Templates

EditDetailsClone

| | | | |
|----------------------|------------------------|-------------------|---------------------|
| Email Template Name | Low Stock Alert | Available For Use | ✓ |
| Template Unique Name | Low_Stock_Alert | Last Used Date | |
| Encoding | Unicode (UTF-8) | Times Used | |
| Author | Aryin De Vera (Change) | | |
| Description | | | |
| Created By | Aryin De Vera | Created By | Aryin De Vera |
| | 11/28/2025, 7:54 AM | Modified By | Aryin De Vera |
| | | | 11/28/2025, 8:01 AM |

EditDetailsClone

Email Template

Send Text and Verify Merge Fields

Subject

Low Stock Alert Email

Plain Text Preview

Dear Inventory Manager,
This is to inform you that the stock for the following product is running low:
Product Name: {!Inventory__c.HandsMen_Product__c}
Current Stock Quantity: {!Inventory__c.Stock_Quantity__c}
Please take the necessary steps to restock this item immediately.
Best Regards,
Inventory Monitoring System

Attachments

Attach File

No records to display

II. Loyalty Status Update

- a. **Subject:** Congratulations! You are now a {!HandsMen_Customer__c.Loyalty_Status__c} member
- b. **Body sample:** Congratulations! You are now a {!HandsMen_Customer__c.Loyalty_Status__c} member and you are eligible for our Loyalty Rewards Program. Enjoy exclusive discounts, early access to offers, and special member benefits. Thank you for your continued support

HTML Email Template

Loyalty Program Email

Help for this Page

Preview your email template below

Email Template Detail

Email Templates from Salesforce

Unified Public Classic Email Templates

Edit PropertiesEdit HTML VersionEdit Text VersionDetailsClone

| | | | |
|----------------------|------------------------|-------------------|---------------------|
| Email Template Name | Loyalty Program Email | Available For Use | ✓ |
| Template Unique Name | Loyalty_Program_Email | Last Used Date | |
| Classic Letterhead | Standard Template | Times Used | |
| Email Layout | Free Form Letter | | |
| Encoding | Unicode (UTF-8) | | |
| Author | Aryin De Vera (Change) | | |
| Description | | | |
| Created By | Aryin De Vera | Modified By | Aryin De Vera |
| | 11/28/2025, 7:55 AM | | 11/28/2025, 7:55 AM |

Edit PropertiesEdit HTML VersionEdit Text VersionDetailsClone

Email Template

Send Text and Verify Merge Fields

Subject

Loyalty Program Email

HTML Preview

Congratulations! You are now a {!HandsMen_Customer__c.Loyalty_Status__c} member and you are eligible for our Loyalty Rewards Program. Enjoy exclusive discounts, early access to offers, and special member benefits. Thank you for your continued support.

III. Order Confirmation

- a. **Subject:** Your order #{!HandsMen_Order__c.Name} has been confirmed!
- b. **Body sample:** Dear {!HandsMen_Order__c.HandsMen_Customer__c},
Your order #{!HandsMen_Order__c.Name} has been confirmed! Thank you for shopping with us. Best Regards, Sales Team

The screenshot shows the Salesforce Email Template Editor interface. At the top, the title is 'Order Confirmation Email'. Below the title, there are buttons for 'Edit Properties', 'Edit HTML Version', 'Edit Text Version', 'Preview', and 'Close'. The main area is divided into two sections: 'Email Template Detail' and 'Email Template'.

Email Template Detail: This section contains a table with the following information:

| Field | Value |
|----------------------|--|
| Email Template Name | Unified Public Classic Email Templates |
| Template Unique Name | Order_Confirmation_Email |
| Classic Label Name | HandsMen_Threads |
| Email Layout | Free Form Layout |
| Encoding | Unicode (UTF-8) |
| Folder | Apex On-Step (Standard) |
| Description | |
| Created By | Apex On-Step |
| Created Date | 11/28/2024, 7:52 AM |
| Modified By | Apex On-Step |
| Modified Date | 11/28/2024, 7:40 PM |

Below the table, there are buttons for 'Edit Properties', 'Edit HTML Version', 'Edit Text Version', 'Preview', and 'Close'.

Email Template: This section contains a 'Subject' field with the text 'Your Order has been Confirmed!' and an 'HTML Preview' section. The HTML preview shows a red header bar, followed by the text: 'Dear {!HandsMen_Order__c.HandsMen_Customer__c}, Your order #{!HandsMen_Order__c.Name} has been confirmed! Thank you for shopping with us. Best Regards, Sales Team'. Below the text is a red footer bar.

Conclusion

The HandsMen Threads CRM successfully streamlines bespoke tailoring operations by integrating customer management, order processing, inventory tracking, loyalty automation, and Apex-driven logic into one cohesive platform. Through the combined use of Flows, Validation Rules, Email Alerts, and Apex Triggers, the system delivers accurate data handling, reduced manual work, and enhanced customer engagement.

Overall, the solution demonstrates how Salesforce can effectively support and optimize business processes for a modern, custom-tailoring business.