



# **HandsMen Threads: Elevating the Art of Sophistication in Men's Fashion Capstone Project**

**In partial fulfillment of the requirements for the subject MNSYSIT**

**Submitted by:**

Charlemagne De Guzman

**Submitted to:**

Ms. Jo Anne M. de la Cuesta &  
Trailhead Virtual Internship Program

**Date of Submission:**

October 29, 2025

# Table of Contents

I.	.....	Abstract
II.	.....	Introduction
III.	.....	Statement of the Problem
IV.	.....	Objectives
V.	.....	Scope and Limitations
VI.	.....	System Overview
VII.	.....	Project Phases
	a.	Phase 1: Architecture & Planning
	b.	Phase 2: Development
	c.	Phase 3: Testing & QA
	d.	Phase 4: Deployment & Training
VIII.	.....	Detailed Execution of Project Components
	a.	Validation Rules
	b.	Profiles, Roles, and Users
	c.	Permission Set
	d.	Classic Email Templates
	e.	Email Alerts & Flows
	f.	Apex Class and Trigger
	g.	Apex Batch Class and Scheduling
IX.	.....	Testing and Quality Assurance
X.	.....	Deployment Summary
XI.	.....	Conclusion

# **Abstract**

The goal of the HandsMen Threads Salesforce implementation project is to automate and modernize a men's fashion company's business processes. The goal of this project is to create a complete Salesforce system that streamlines workflows for inventory management, order tracking, customer management, and communication. HandsMen Threads improve efficiency, data accuracy, and customer engagement by integrating automation tools like Flows, Apex Triggers, Validation Rules, and Batch Jobs. The project exhibits real-world business use case-aligned Salesforce development, configuration, and customization expertise.

# **Introduction**

HandsMen Threads is a high-end men fashion firm which deals with elegant attire and accessories. In an effort to reduce its operations and enhance its customer relationship management strategy, the company has chosen Salesforce as its centralized system. This capstone project aims at developing and designing a complete Salesforce application that helps in supporting day-to-day business operations- including order management, customer loyalty, inventory management, and automated notification.

The essential Salesforce components that are brought out in this project include: declarative tools, automation flows, Apex programming, and system deployment. It shows how Salesforce may be deployed to change manual operations into effective digital processes achieved through business expansion and customer satisfaction.

# **Objectives**

To resolve the specified challenges successfully, it was essential to establish the objectives and deliverables of the Salesforce implementation. The goals are the roadmap that will direct in the development, testing, and implementation of HandsMen Threads CRM solution. All these objectives would help to increase the level of operational efficiency, customer satisfaction, and create a scalable system to serve the business in the future.

1. To model and create a unique Salesforce architecture in accordance with the business model of HandsMen Threads.
2. Automate the major processes in the business with Flows, Validation Rules, and Apex triggers.
3. To adopt an effective order management system and inventory tracking system.
4. To develop customer communication automation by email templates and alerts.
5. To achieve data security, accuracy, and performance by testing and deploying best practices.

# Scope and Limitations

## Scope:

The project will undertake the entire Salesforce implementation framework of HandsMen Threads, such as designing the system, customizing it, automating it and implementing it. It pays attention to automating order processing, inventory tracking, and customer interaction by email notifications and validation conditions. This system guarantees the integrity of data, automation of workflow, and effective management of business processes with the use of Salesforce applications such as Flows, Apex, and Batch Job.

## Limitations:

The project does not entail the third-party system integrations, i.e. payment gateways or ERP systems. Mobile application extension and customer portal are suggested to be enhanced in the future but did not comprise the scope of the current project. Also, despite the automation of key workflows, real-time analytics and AI-driven recommendations were not included because of the lack of time and resources.

# System Overview

The Salesforce solution for HandsMen Threads consists of several interconnected components:

- **Custom Objects:** HandsMen Customer, HandsMen Order, Inventory, Product.
- **Automation Tools:** Record-Triggered Flows, Scheduled Flows, Validation Rules, and Apex Triggers.
- **Batch Processing:** Apex Batch Job for daily inventory synchronization.
- **Email Templates & Alerts:** Automated order confirmations, stock alerts, and loyalty program updates.
- **Profiles, Roles & Permission Sets:** Ensure proper access control and secure data handling.

# Project Phases

## Phase 1: Architecture & Planning

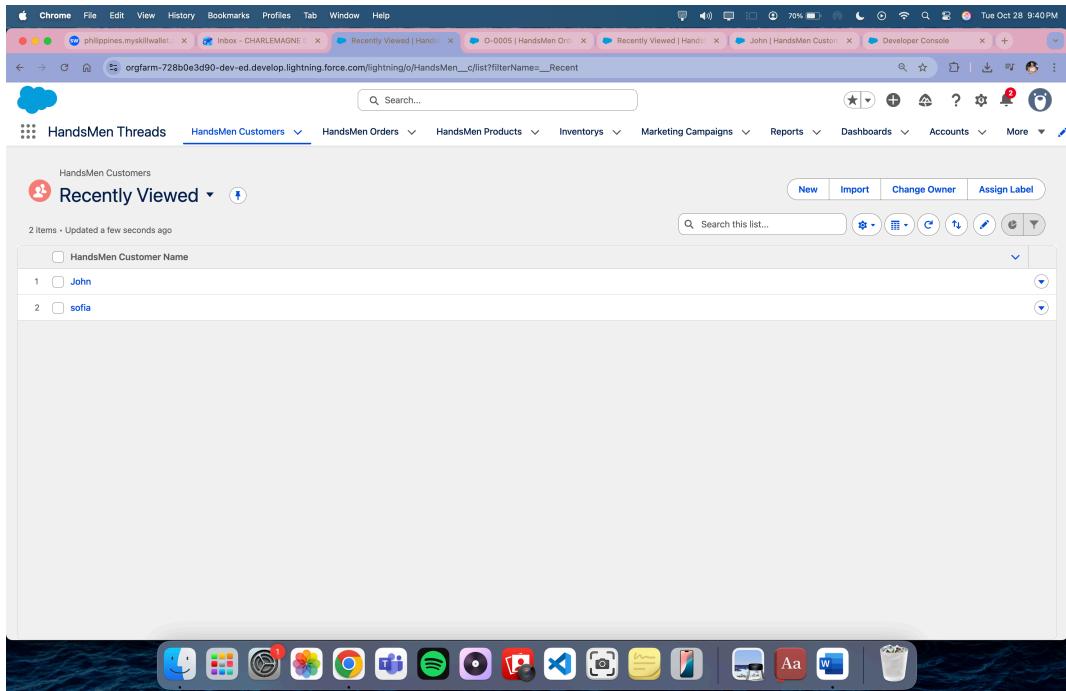
- Defined objects, fields, and relationships between **HandsMen Customer**, **HandsMen Order**, and **Inventory**.
- Established formula fields for calculating totals and loyalty points.
- Designed validation rules to ensure data accuracy.
- Created automation workflows and Apex triggers business logic.
- Designed email templates for notifications and customer communication.

## Phase 2: Development

- Created custom objects and fields.

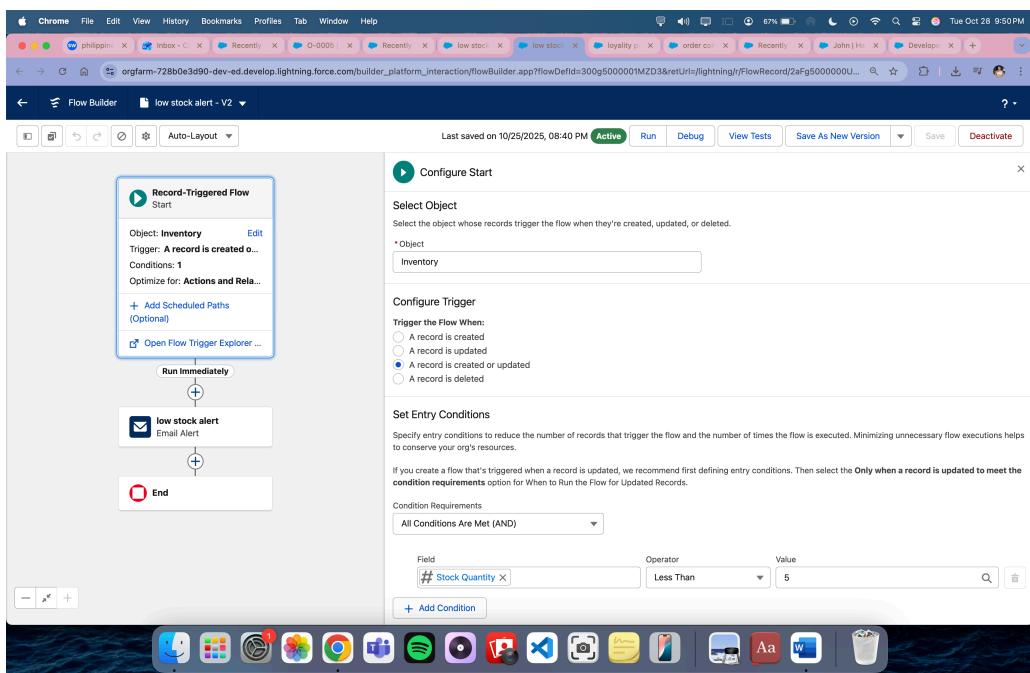
The screenshot shows the Salesforce Object Manager page. The page title is "Object Manager" under the "SETUP" tab. It displays a list of 106+ items, sorted by label. The custom objects created in Phase 1 are highlighted with red arrows:

Object Name	Type	Object Description	Last Modified	Action
HandsMen Customer	Custom Object	HandsMen_c	10/25/2025	▼
HandsMen Order	Custom Object	HandsMen_Order__c	10/25/2025	▼
HandsMen Product	Custom Object	Han__c	10/25/2025	▼
Inventory	Custom Object	Inventory__c	10/25/2025	▼

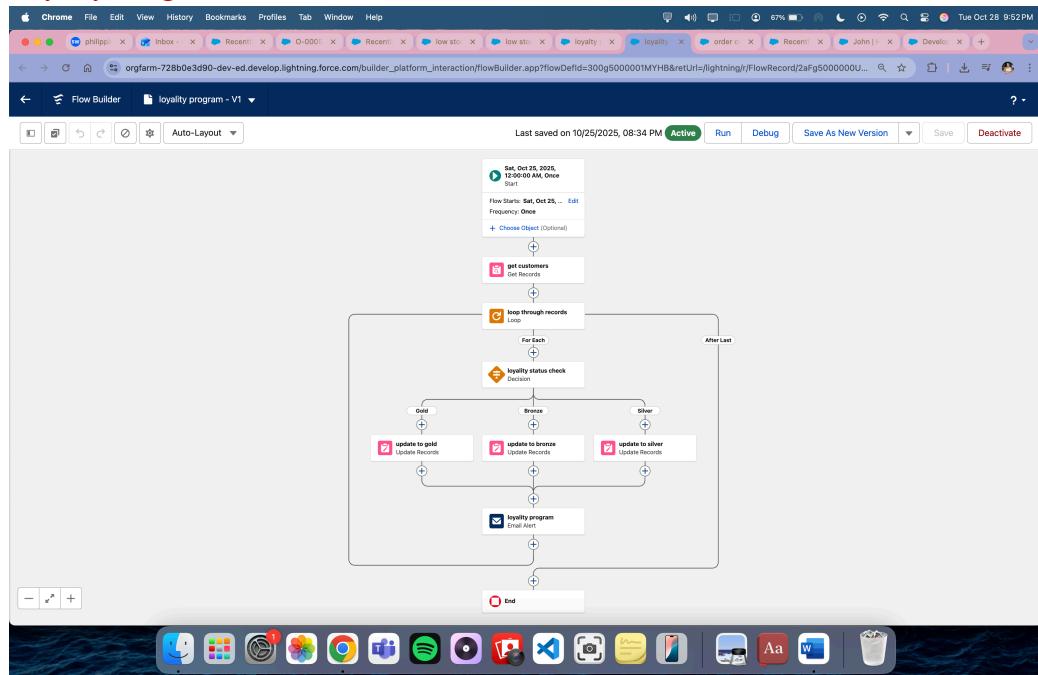


- Implemented automation via Flows, Process Builders, and Apex triggers.

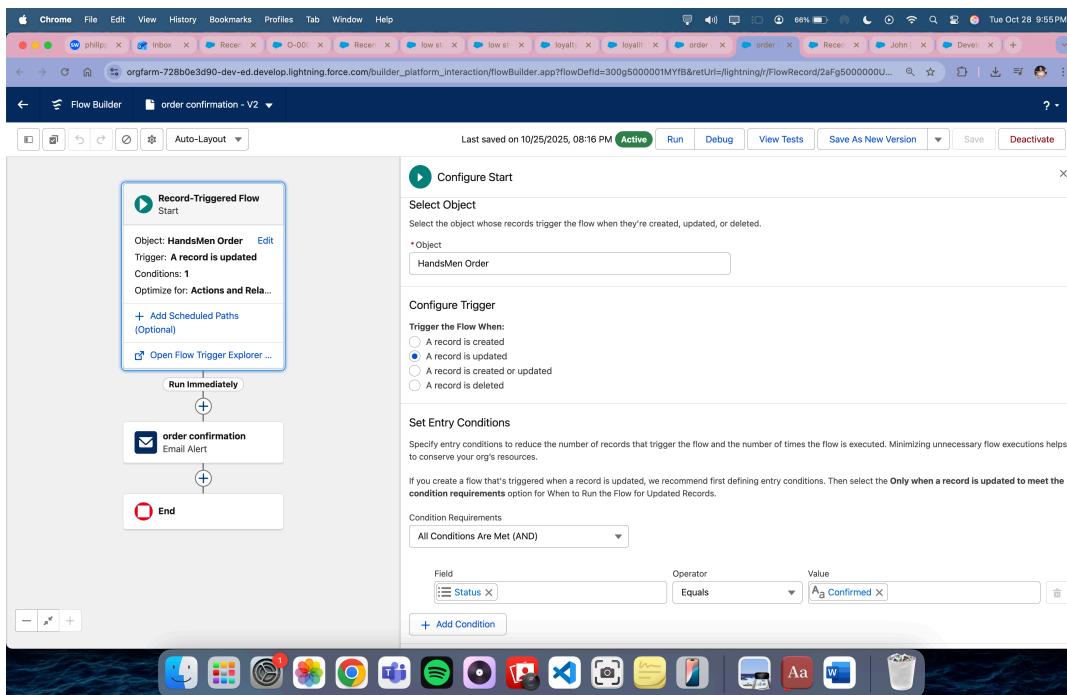
## Low Stock Flow



## Loyalty Program Flow



## Order Confirmation Flow



- Configured data security using Profiles, Roles, and Permission Sets.

The screenshot shows the Salesforce Setup Apex Jobs page. At the top, there's a message: "Click here to see the new Job API jobs pane". Below it, a banner says "Apex Jobs" and "Monitor the status of all Apex jobs, and optionally, abort jobs that are in progress." It also displays "Percent of Asynchronous Apex Used: 0%" and "You have currently used 1 asynchronous Apex operations out of an allowed 24-hour organization limit of 250,000. To learn about how this limit is calculated and what contributes to it, see the Lightning Platform: Apex Limits topic." A table lists one job entry:

Action	Submitted Date	Job Type	Status	Status Detail	Total Matches	Batches Processed	Failures	Submitted By	Completion Date	Apex Class	Apex Method	Apex Job ID
	10/25/2020, 6:07 AM	Scheduled Apex	Success		0	0	0	0		DashInventorySync		707g00000000Ap

- Developed batch jobs for inventory restocking and synchronization.

The screenshot shows the HandsMen Threads Salesforce page. The navigation bar includes "Inventory", "Recently Viewed", "New", "Import", and "Assign Label". The main area displays a list of inventory items under the heading "Inventory Number". There are two items listed: "I-0002" and "I-0001".

- Set up email templates for order confirmation, stock alerts, and loyalty programs.

**Fields & Relationships**  
11 Items, Sorted by Field Label

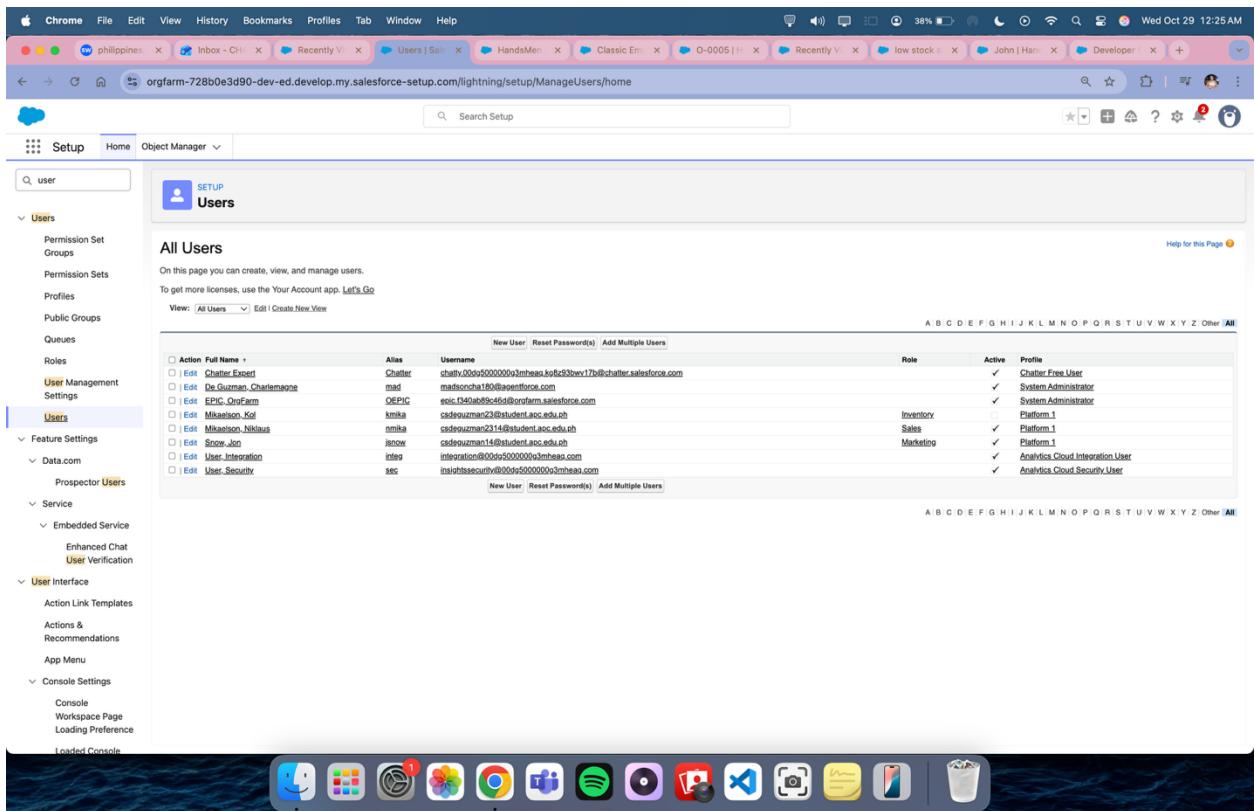
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Email	Email__c	Email		
FirstName	FirstName__c	Text(60)		
FullName	FullName__c	Formula (Text)		
HandsMen Customer Name	Name	Text(80)		
Last Modified By	LastModifiedById	Lookup(User)		
LastName	LastName__c	Text(60)		
Loyalty Status	Loyalty_Status__c	Picklist		
Owner	OwnerId	Lookup(User,Group)		
Phone	Phone__c	Phone		
Total Purchases	Total_Purchases__c	Number(18, 0)		

## Phase 3: Testing & QA

- Performed unit testing for each automation feature.
- Conducted end-to-end testing with sample records for customers, orders, and inventory.
- Verified data consistency and validation of rule enforcement.
- Executed performance testing and basic security validation.

## **Phase 4: Deployment & Training**

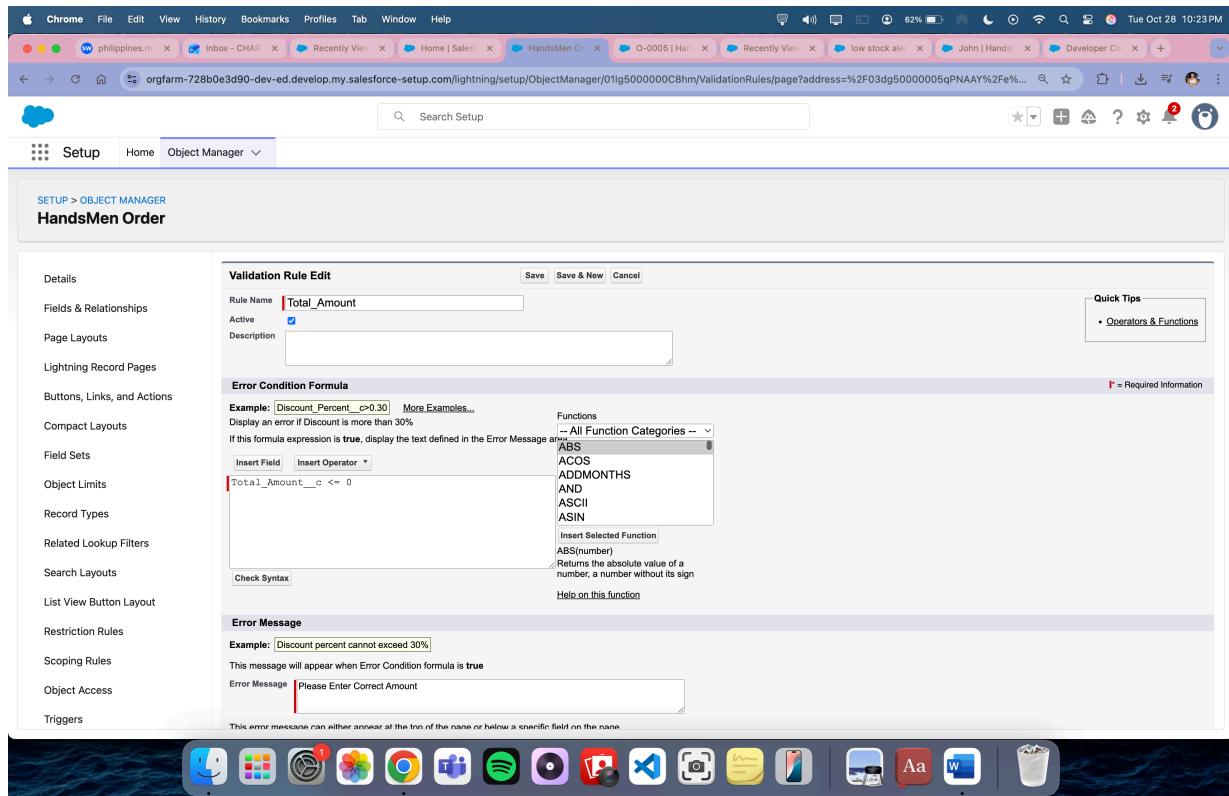
- Implemented configuration and automation in the production environment.
  - Held sales and inventory personnel training.
  - Monitored behavior in the post-deployment system and responded to support.



# Detailed Execution of Project Components

## 1. Validation Rules

Object	Field	Validation Rule	Error Message
HandsMen Order	Total_Amount__c	Total_Amount__c <= 0	"Please Enter Correct Amount"
Inventory__c	Stock_Quantity__c	Stock_Quantity__c <= 0	"The inventory count is never less than zero."
HandsMen Customer	Email	NOT CONTAINS>Email, "@gmail.com")	"Please fill Correct Gmail"



Chrome File Edit View History Bookmarks Profiles Tab Window Help

philippines.m...

orgfarm-728b0e3d90-dev-ed.develop.my.salesforce-setup.com/lightning/setup/ObjectManager/01g5000000C9Pj/ValidationRules/page?address=%2F03dg50000005qQzAAI%2Fe%3...

Tue Oct 28 10:24 PM

Setup Home Object Manager

SETUP > OBJECT MANAGER

## Inventory

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

Validation Rule Edit

Rule Name: Stock\_Quantity

Active:

Description:

Error Condition Formula

Example: Discount\_Percent\_c>0.30 More Examples...

If this formula expression is true, display the text defined in the Error Message area.

Insert Field Insert Operator

Stock\_Quantity\_c <= 0

Functions: ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN

Check Syntax

Error Message

Example: Discount percent cannot exceed 30%

This message will appear when Error Condition formula is true.

Error Message: The inventory count is never less than zero.

Quick Tips: Operators & Functions

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

Chrome File Edit View History Bookmarks Profiles Tab Window Help

philippines.m...

orgfarm-728b0e3d90-dev-ed.develop.my.salesforce-setup.com/lightning/setup/ObjectManager/01g5000000C9Fd/ValidationRules/page?address=%2F03dg50000005qSbAAI%2Fe%3...

Tue Oct 28 10:22 PM

Setup Home Object Manager

SETUP > OBJECT MANAGER

## HandsMen Customer

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

Validation Rule Edit

Rule Name: Email

Active:

Description:

Error Condition Formula

Example: Discount\_Percent\_c>0.30 More Examples...

If this formula expression is true, display the text defined in the Error Message area.

Insert Field Insert Operator

NOT CONTAINS( Email\_c , "@gmail.com")

Functions: ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN

Check Syntax

Error Message

Example: Discount percent cannot exceed 30%

This message will appear when Error Condition formula is true.

Error Message: Please fill Correct Gmail

Quick Tips: Operators & Functions

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

## 2. Profiles, Roles, and Users

- **Profile:** Cloned *Standard User* → Renamed *Platform 1*.

The screenshot shows the Salesforce Setup interface with the URL `orgfarm-728b0e3d90-dev-ed.develop.my.salesforce-setup.com/lightning/setup/PermSets/page?address=%2FOPSG5000000dSLX`. The left sidebar is open, showing categories like Users, Feature Settings, Data.com, Service, and User Interface. The main content area is titled "Permission Sets" and displays a permission set named "Permission\_Platform\_1". The "Permission Set Overview" section includes fields for Description, License, Session Activation Required, and Permission Set Groups Added To (0). The "API Name" is listed as "Permission\_Platform\_1", and the "Namespace Prefix" is "Permission\_Platform\_1". The "Created By" and "Last Modified By" fields both show "Charlemagne De Guzman" with the timestamp "10/26/2025, 7:45 AM". The "Manage Assignments" and "View Summary" buttons are also present. Below the overview, the "Apps" section lists various app permissions such as Assigned Apps, Assigned Connected Apps, Object Settings, App Permissions, Apex Class Access, Visualforce Page Access, External Data Source Access, Flow Access, Named Credential Access, External Credential Principal Access, Service Presence Status Access, Custom Permissions, Custom Metadata Types, and Custom Setting Definitions. The right side of the interface includes a "Video Tutorial" link and a "Help for this Page" link.

- Granted permissions for HandsMen Products and Inventory objects.
- **Role Hierarchy:** CEO → Sales → Inventory.
- **Users Created:**
  - Niklaus Mikaelson (Role: Sales)
  - Kol Mikaelson (Role: Inventory)
  - Jon Snow (Role: Marketing)

The screenshot shows the Salesforce Setup interface for managing permission sets. The current page is 'Sales permission set' under 'Permission Sets'. The sidebar on the left shows categories like 'Users', 'Permission Set Groups', and 'Custom Code'. The main area displays the 'Current Assignments' table, which lists one user assignment:

Full Name	Active	Role	Profile	User License	Expires On
Niklaus Mikaelson	✓	Sales	Platform 1	Salesforce	

A message at the bottom left says, "Didn't find what you're looking for? Try using Global Search." The top navigation bar shows multiple tabs and the URL <https://orgfarm-728b0e3d90-dev-ed.develop.my.salesforce-setup.com/lightning/setup/PermSets/0PSg5000000ZHcI/PermissionSetAssignment/home>. The bottom of the screen shows the Mac OS X Dock with various application icons.

Chrome File Edit View History Bookmarks Profiles Tab Window Help

philippines.m... Index - CHAT Recently Viewed Home | Sales! Permission Sets O-0005 | Home Recently Viewed low stock alert John | Handl... Developer Co... Tue Oct 28 10:33 PM

orgfarm-728b0e3d90-dev-ed.develop.my.salesforce-setup.com/lightning/setup/PermSets/OPSG5000000ZC5d/PermissionSetAssignment/home

Setup Home Object Manager

PERMISS

Users

Permission Set Groups

**Permission Sets**

Custom Code Custom Permissions

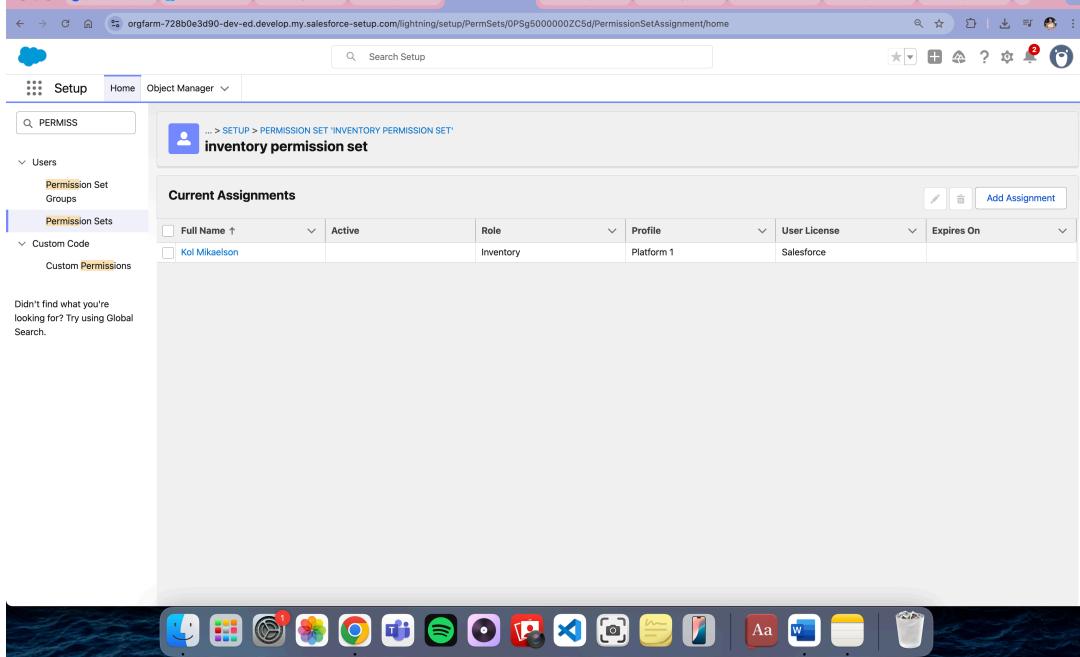
Didn't find what you're looking for? Try using Global Search.

... > SETUP > PERMISSION SET 'INVENTORY PERMISSION SET'  
**inventory permission set**

**Current Assignments**

Full Name	Active	Role	Profile	User License	Expires On
Kol Mikaelson		Inventory	Platform 1	Salesforce	

Add Assignment



Chrome File Edit View History Bookmarks Profiles Tab Window Help

Recently Viewed | HandsMen! Permission Sets | Salesforce New Tab

orgfarm-728b0e3d90-dev-ed.develop.my.salesforce-setup.com/lightning/setup/PermSets/OPSG5000000ZNWP/PermissionSetAssignment/home

Setup Home Object Manager

Q. PERMISS

Users

Permission Set Groups

**Permission Sets**

Custom Code Custom Permissions

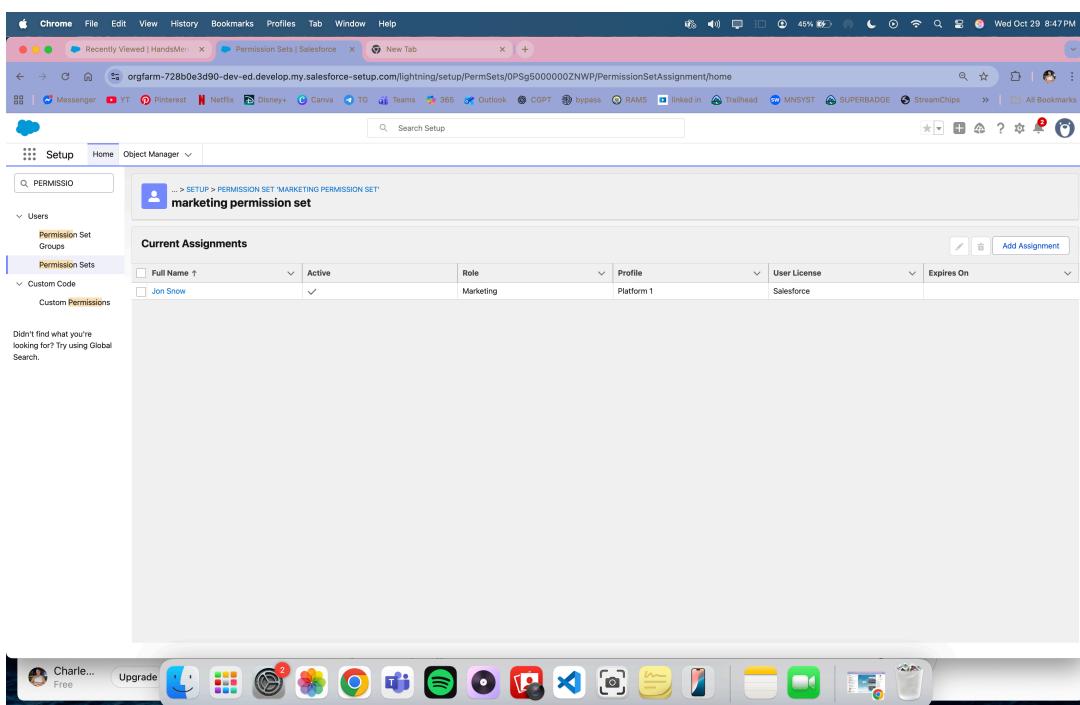
Didn't find what you're looking for? Try using Global Search.

... > SETUP > PERMISSION SET MARKETING PERMISSION SET  
**marketing permission set**

**Current Assignments**

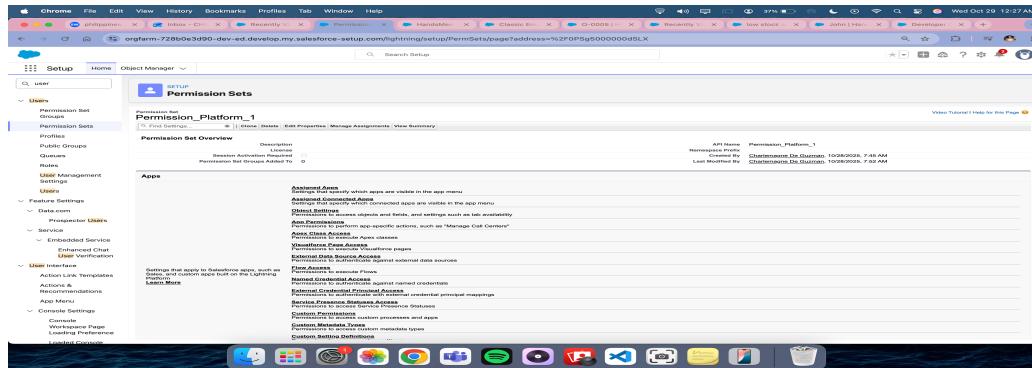
Full Name	Active	Role	Profile	User License	Expires On
Jon Snow		Marketing	Platform 1	Salesforce	

Add Assignment



### 3. Permission Set (Permission\_Platform\_1)

- Created permission set named **Permission\_Platform\_1**.



- Granted **Read, Create, Edit, Delete** access for *HandsMen Customer* and *HandsMen Order* objects.
- Assigned permission set to *Platform 1* users.

The image contains two side-by-side screenshots of the Salesforce Setup interface under the 'Permission Sets' tab, showing the configuration of object settings for 'HandsMen Customers' and 'HandsMen Orders'.

**Left Screenshot (HandsMen Customers):**

- Tab Settings:** Shows 'Available' and 'Visible' tabs. Under 'Available', there is one checkbox. Under 'Visible', there is one checkbox.
- Object Permissions:** A table showing permissions for 'HandsMen Customers':

Permission Name	Enabled
Read	<input checked="" type="checkbox"/>
Create	<input checked="" type="checkbox"/>
Edit	<input checked="" type="checkbox"/>
Delete	<input checked="" type="checkbox"/>
View All Records	<input type="checkbox"/>
Modify All Records	<input type="checkbox"/>
View All Fields	<input type="checkbox"/>

**Right Screenshot (HandsMen Orders):**

- Tab Settings:** Shows 'Available' and 'Visible' tabs. Under 'Available', there is one checkbox. Under 'Visible', there is one checkbox.
- Object Permissions:** A table showing permissions for 'HandsMen Orders':

Permission Name	Enabled
Read	<input checked="" type="checkbox"/>
Create	<input checked="" type="checkbox"/>
Edit	<input checked="" type="checkbox"/>
Delete	<input checked="" type="checkbox"/>
View All Records	<input type="checkbox"/>
Modify All Records	<input type="checkbox"/>
View All Fields	<input type="checkbox"/>

## 4. Classic Email Templates

Created three email templates for customer communication:

### 1. Order Confirmation Email

Subject: *Your Order has been Confirmed!*

Subject | Your Order has been Confirmed!

HTML Preview |

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

Plain Text Preview |

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

## 2. Low Stock Alert

Automated notification to inventory managers when stock is low.

Text Email Template

### Low Stock Alert

Preview your email template below.

#### Email Template Detail

[Edit](#) [Delete](#) [Clone](#)

Email Templates from Salesforce	Unfiled Public Classic Email Templates
Email Template Name	Low Stock Alert
Template Unique Name	Low_Stock_Alert
Encoding	Unicode (UTF-8)
Author	<a href="#">Charlemagne De Guzman</a> [Change]
Description	
Created By	<a href="#">Charlemagne De Guzman</a> , 10/25/2025, 4:38 AM

[Edit](#) [Delete](#) [Clone](#)

#### Email Template

[Send Test and Verify Merge Fields](#)

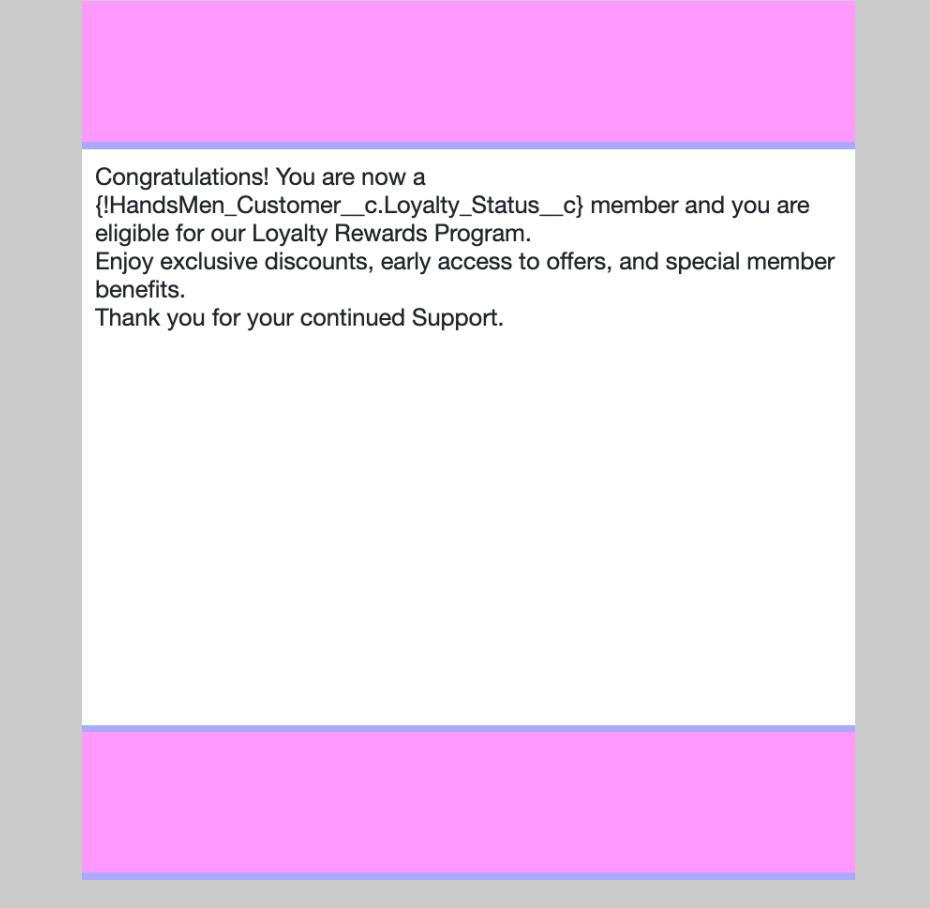
**Subject** | Low Stock Alert Email

#### 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

### 3. Loyalty Program Email

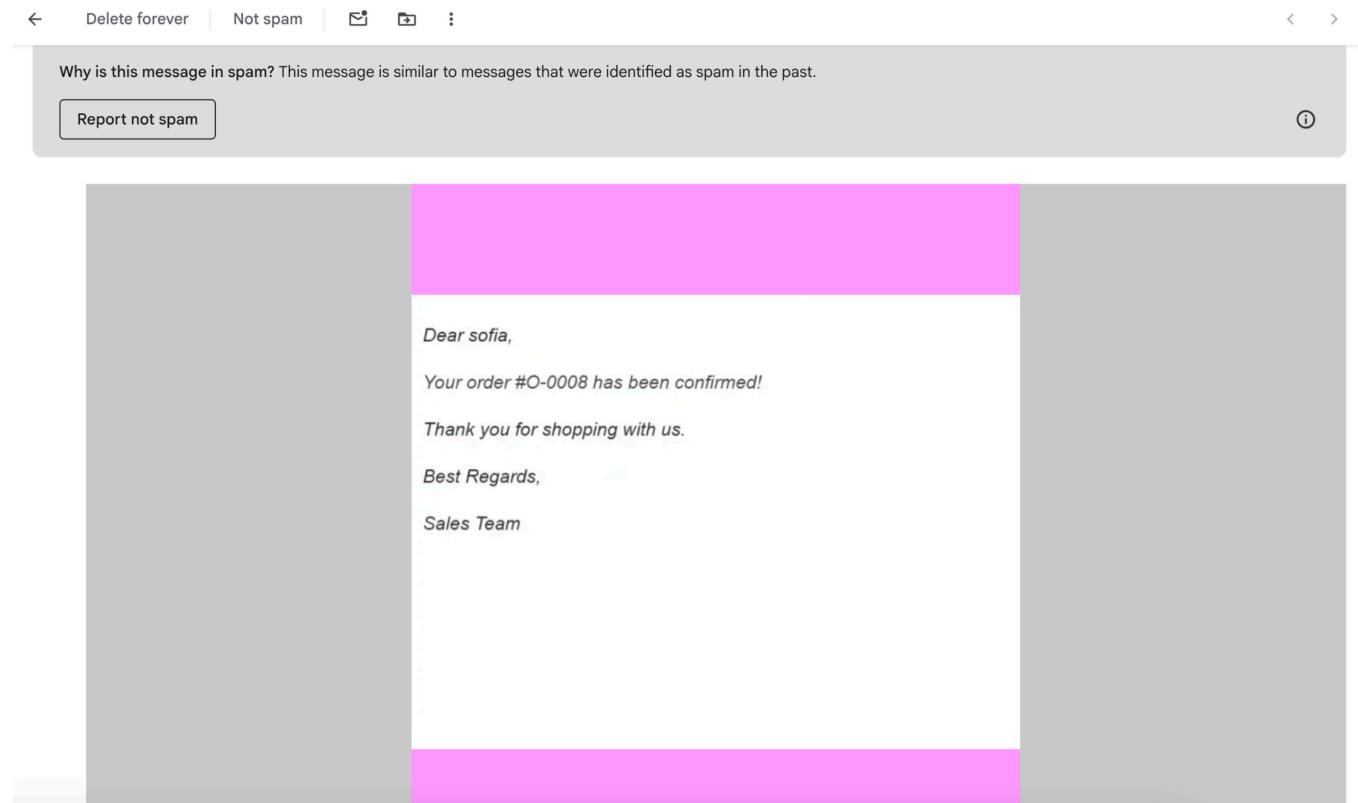
Sent to customers when their loyalty tier is updated.

Email Template	<a href="#">Send Test and Verify Merge Fields</a>
<p><b>Subject</b>   Loyalty Program Email</p> <p><b>HTML Preview</b></p>  <p>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.</p> <p><b>Plain Text Preview</b></p> <p>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.</p>	

## 5. Email Alerts & Flows

### Order Confirmation Flow (Record-Triggered):

- Object: *Order\_\_c*
- Trigger: *When record is updated*
- Condition: *Status = “Confirmed”*
- Action: Send *Order Confirmation Email Alert*



## **Stock Alert Flow (Record-Triggered):**

- Object: *Inventory\_c*
  - Condition: *Stock\_Quantity\_c < 5*
  - Action: Send *Low Stock Alert Email*

Low Stock Alert Email Inbox x

OrgFarm EPIC <epic.orgfarm@salesforce.... Mon, Oct 27, 6:14 PM (1 day ago) star smile left more

to me ▾

Dear Inventory Manager,

This is to inform you that the stock for the following product is running low:

Product Name: T-Shirt

Current Stock Quantity:4

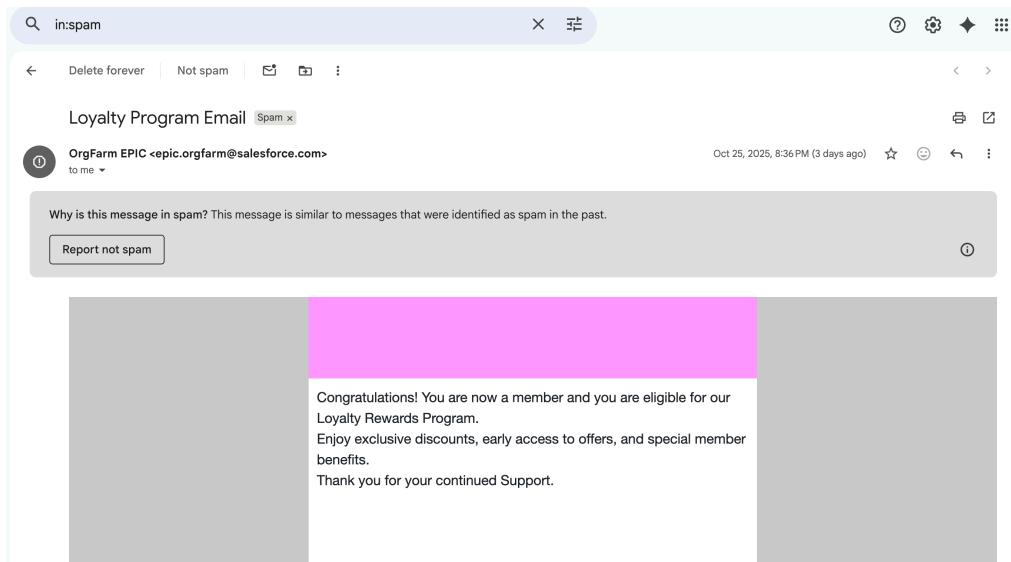
Please take the necessary steps to restock this item immediately.

Best Regards,

Inventory Monitoring System

## Loyalty Status Update (Scheduled Flow):

- Runs daily to update customer loyalty tiers:
    - *Gold*: Total Purchases > 1000
    - *Silver*: Between 500–1000
    - *Bronze*: < 500



## Outcomes

For each path the automation can take, create an outcome. For each outcome, specify the conditions that must be met for the automation to take that path.

OUTCOME ORDER	OUTCOME DETAILS	Delete Outcome
Gold	* Outcome Label Gold	* Outcome API Name Gold
Bronze	Condition Requirements to Execute Outcome All Conditions Are Met (AND)	
Silver	* Resource # ...ugh records > Total Purchases X	
	Operator Greater Than	Value 1000

## Outcomes

For each path the automation can take, create an outcome. For each outcome, specify the conditions that must be met for the automation to take that path.

OUTCOME ORDER	OUTCOME DETAILS	Delete Outcome
Gold	* Outcome Label	
Bronze	* Outcome Label Bronze	* Outcome API Name Bronze
Silver	Condition Requirements to Execute Outcome All Conditions Are Met (AND)	
	* Resource # ...ugh records > Total Purchases X	
	Operator Less Than or Equal	Value 500

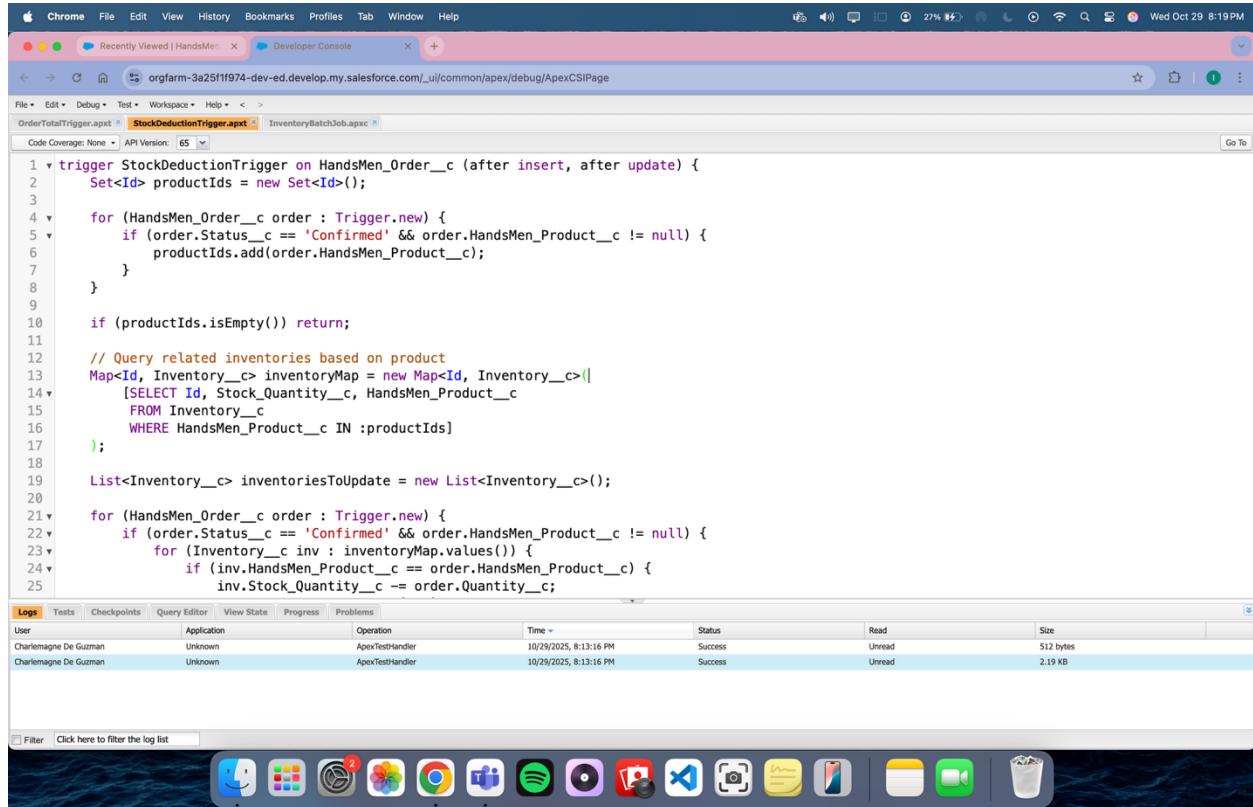
## Outcomes

For each path the automation can take, create an outcome. For each outcome, specify the conditions that must be met for the automation to take that path.

OUTCOME ORDER	OUTCOME DETAILS
Gold	* Label Silver
Bronze	
Silver	

## 6. Apex Class and Trigger

### Apex Class: StockDeductionTrigger



```
trigger StockDeductionTrigger on HandsMen_Order__c (after insert, after update) {
    Set<Id> productIds = new Set<Id>();
    for (HandsMen_Order__c order : Trigger.new) {
        if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
            productIds.add(order.HandsMen_Product__c);
        }
    }
    if (productIds.isEmpty()) return;
    // Query related inventories based on product
    Map<Id, Inventory__c> inventoryMap = new Map<Id, Inventory__c>(
        [SELECT Id, Stock_Quantity__c, HandsMen_Product__c
         FROM Inventory__c
         WHERE HandsMen_Product__c IN :productIds]
    );
    List<Inventory__c> inventoriesToUpdate = new List<Inventory__c>();
    for (HandsMen_Order__c order : Trigger.new) {
        if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
            for (Inventory__c inv : inventoryMap.values()) {
                if (inv.HandsMen_Product__c == order.HandsMen_Product__c) {
                    inv.Stock_Quantity__c -= order.Quantity__c;
                    inventoriesToUpdate.add(inv);
                }
            }
        }
    }
}
```

User	Application	Operation	Time	Status	Read	Size
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	512 bytes
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	2.19 KB

```
trigger StockDeductionTrigger on HandsMen_Order__c (after insert, after update) { Set<Id> productIds = new Set<Id>();
```

```
for (HandsMen_Order__c order : Trigger.new) {
    if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
        productIds.add(order.HandsMen_Product__c);
    }
}
```

```
if (productIds.isEmpty()) return;
```

```
// Query related inventories based on product
Map<Id, Inventory__c> inventoryMap = new Map<Id, Inventory__c>(
    [SELECT Id, Stock_Quantity__c, HandsMen_Product__c
     FROM Inventory__c
     WHERE HandsMen_Product__c IN :productIds]
```

```
);
```

```
List<Inventory__c> inventoriesToUpdate = new List<Inventory__c>();  
  
for (HandsMen_Order__c order : Trigger.new) {  
    if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {  
        for (Inventory__c inv : inventoryMap.values()) {  
            if (inv.HandsMen_Product__c == order.HandsMen_Product__c) {  
                inv.Stock_Quantity__c -= order.Quantity__c;  
                inventoriesToUpdate.add(inv);  
                break;  
            }  
        }  
    }  
}  
  
if (!inventoriesToUpdate.isEmpty()) {  
    update inventoriesToUpdate;  
}  
}
```

## Apex Trigger: OrderTotalTrigger

The screenshot shows the Salesforce Developer Console interface. At the top, the browser title is "orgfarm-3a25f1f974-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCsPage". The tabs include "OrderTotalTrigger.apxt", "StockDeductionTrigger.apxt", and "InventoryBatchJob.apxt". The code editor displays the Apex trigger "OrderTotalTrigger" which calculates the total amount for each order based on quantity and product price. Below the code is a table of test logs:

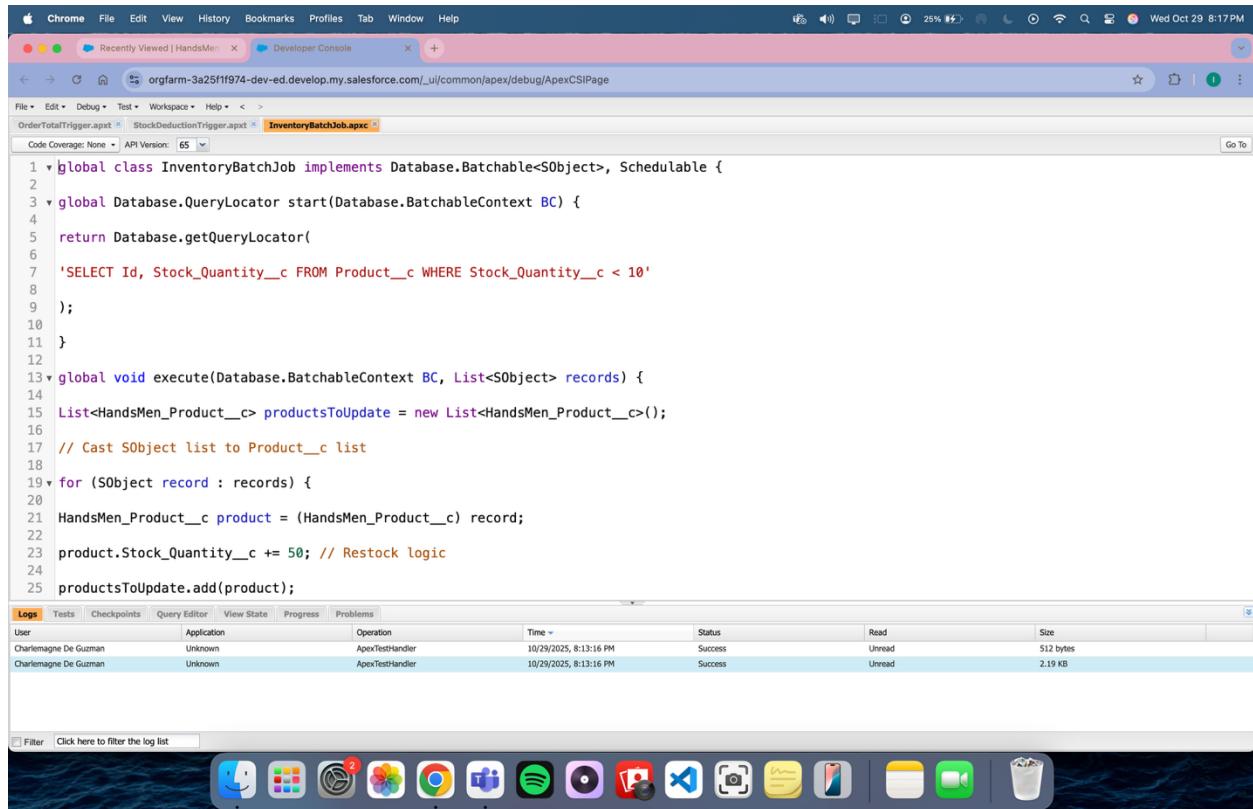
User	Application	Operation	Time	Status	Read	Size
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	512 bytes
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	2.19 KB

At the bottom, there is a toolbar with various application icons.

```
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 }
```

## 7. Apex Batch Class and Scheduling

### Apex Class: InventoryBatchJob



```
global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {
    global Database.QueryLocator start(Database.BatchableContext BC) {
        return Database.getQueryLocator(
            'SELECT Id, Stock_Quantity__c FROM Product__c WHERE Stock_Quantity__c < 10'
        );
    }
    global void execute(Database.BatchableContext BC, List<SObject> records) {
        List<HandsMen_Product__c> productsToUpdate = new List<HandsMen_Product__c>();
        // Cast SObject list to Product__c list
        for (SObject record : records) {
            HandsMen_Product__c product = (HandsMen_Product__c) record;
            product.Stock_Quantity__c += 50; // Restock logic
            productsToUpdate.add(product);
        }
    }
}
```

Logs	Tests	Checkpoints	Query Editor	View State	Progress	Problems
User	Application	Operation	Time	Status	Read	Size
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	512 bytes
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	2.19 KB

```
global class InventoryBatchJob implements Database.Batchable, Schedulable {
```

```
    global Database.QueryLocator start(Database.BatchableContext BC) {
```

```
        return Database.getQueryLocator(
```

```
            'SELECT Id, Stock_Quantity__c FROM Product__c WHERE Stock_Quantity__c < 10'
```

```
    );
```

```
}
```

```
    global void execute(Database.BatchableContext BC, List<SObject> records) {
```

```
        List<HandsMen_Product__c> productsToUpdate = new List<HandsMen_Product__c>();
```

```
// Cast SObject list to Product__c list

for (SObject record : records) {

    HandsMen_Product__c product = (HandsMen_Product__c) record;

    product.Stock_Quantity__c += 50; // Restock logic

    productsToUpdate.add(product);

}

if (!productsToUpdate.isEmpty()) {

    try {

        update productsToUpdate;

    } catch (DmlException e) {

        System.debug('Error updating inventory: ' + e.getMessage());

    }

}

}

global void finish(Database.BatchableContext BC) {

    System.debug('Inventory Sync Completed');

}

// Scheduler Method

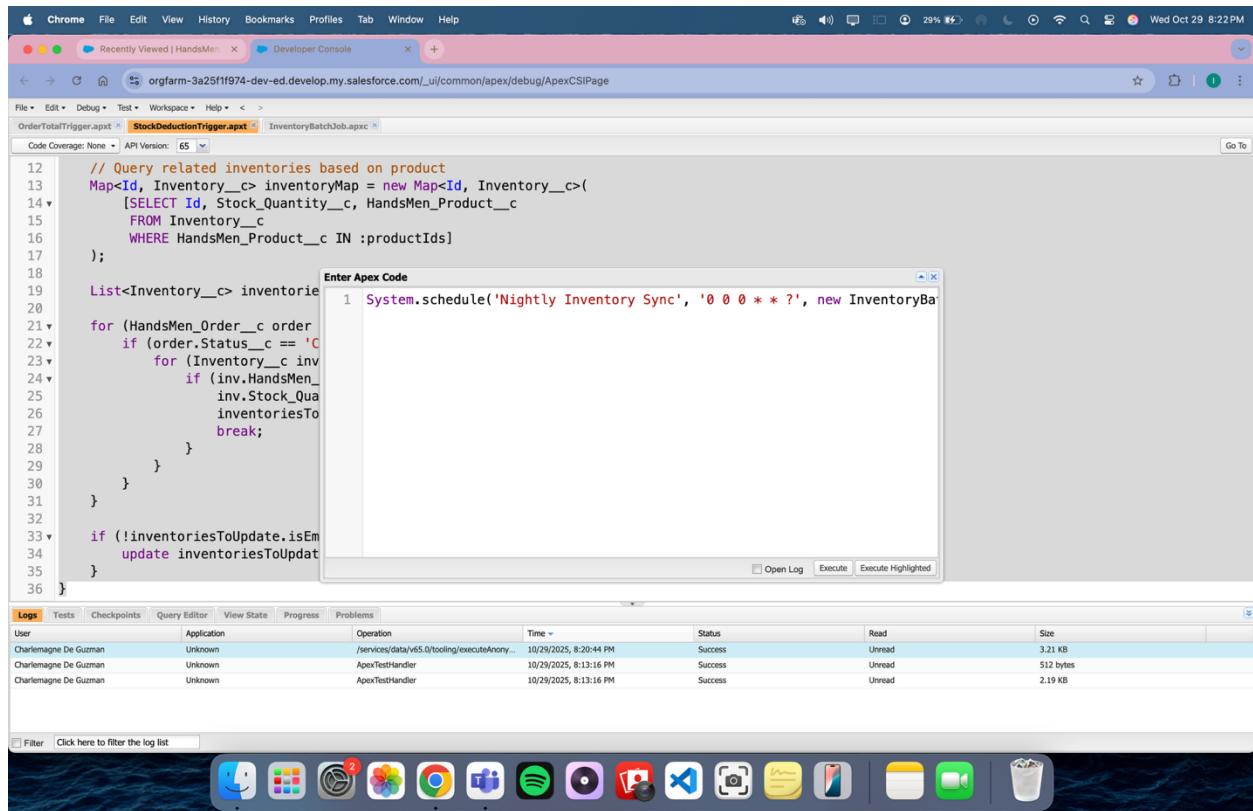
global void execute(SchedulableContext SC) {

    InventoryBatchJob batchJob = new InventoryBatchJob();

    Database.executeBatch(batchJob, 200)

}
```

## Scheduling Command:



The screenshot shows the Salesforce Developer Console interface. A modal window titled "Enter Apex Code" is open, displaying the line of code: `System.schedule('Nightly Inventory Sync', '0 0 0 * * ?', new InventoryBatchJob());`. The background shows a trigger named "StockDeductionTrigger.apxt" with some Apex code. Below the code editor is a log viewer showing three entries from a user named "Charlemagne De Guzman". At the bottom, there's a dock with various application icons.

```
// Query related inventories based on product
Map<Id, Inventory__c> inventoryMap = new Map<Id, Inventory__c>(
    [SELECT Id, Stock_Quantity__c, HandsMen_Product__c
     FROM Inventory__c
     WHERE HandsMen_Product__c IN :productIds]
);
List<Inventory__c> inventories;
for (HandsMen_Order__c order
     if (order.Status__c == 'C'
         for (Inventory__c inv
              if (inv.HandsMen_
                  inv.Stock_Qua
                  inventoriesTo
                  break;
            }
        }
    }
}
if (!inventoriesToUpdate.isEmpty()
    update inventoriesToUpdate
}
```

User	Application	Operation	Time	Status	Read	Size
Charlemagne De Guzman	Unknown	/services/data/v65.0/tooling/executeAnony...	10/29/2025, 8:20:44 PM	Success	Unread	3.21 KB
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	512 bytes
Charlemagne De Guzman	Unknown	ApexTestHandler	10/29/2025, 8:13:16 PM	Success	Unread	2.19 KB

`System.schedule('Daily Inventory Sync', '0 0 0 * * ?', new InventoryBatchJob());`

## **Testing and Quality Assurance**

- All the validation rules were applied to valid and invalid data.
- Test records were checked using email alerts.
- Classes that had apex code passed without runtime errors.
- The executed batch job was successful and automatically updated low-stock products.
- For proper access control, user roles and permission sets were checked.
- 

## **Deployment Summary**

- Successfully deployed the full Salesforce configuration to production.
- User training was conducted for Sales and Inventory personnel.
- Post-deployment monitoring confirmed stable operations with accurate data flow.

## **Conclusion**

- The HandsMen Threads Salesforce implementation project was a successful project that turned traditional business operations into an intelligent automated CRM environment. The system enhanced order management, minimized human error, and increased overall productivity through the integration of Apex programming, Flows and declarative tools.
- Along with enhancing technical expertise in Salesforce, this capstone project illustrates a possibility of cloud CRM solutions in facilitating digital change in the retail fashion sector.