

## Salesforce CRM Project Documentation

### Project Title:

**Handsmen Threads: Elevating the Art of Sophistication in Men's Fashion**

**Author:** John Vincent Del Rosario

**Date:** November 27, 2025

## Project Overview

Handsmen Threads is a Salesforce Customer Relationship Management/CRM solution designed to streamline customer engagement and sales processes for a premium men's fashion brand. The CRM focuses on personalized customer management, order tracking, and loyalty programs, ensuring that the brand provides a sophistication both in its products and customer experience.

## Objectives

- Enhancing customer relationship management in terms of tracking preferences and purchased history.
- Streamline booking and order processes for bespoke tailoring services.
- Provide actionable insights through dashboards and reports.
- Support scalability for future improvements like AI-driven suggestions and chatbot integration.

## Phase 1: Requirement Analysis & Planning

- **Business Requirements** – customers need a personalized service, efficient order tracking, and loyalty rewards.

- **Project Scope** – build a CRM with modules specifically for customer profiles, booking, payments and loyalty programs.
- **Data & Security Model** – custom objects for customers, orders, loyalty points (role-based access for sales staff and admins).
- **Stakeholders** – sales team, tailoring staff, management and IT support.
- **Execution Roadmap** – Requirement gathering, Development, Testing, Deployment lastly is Maintenance.

## Phase 2: Salesforce Development – Backend & Configurations

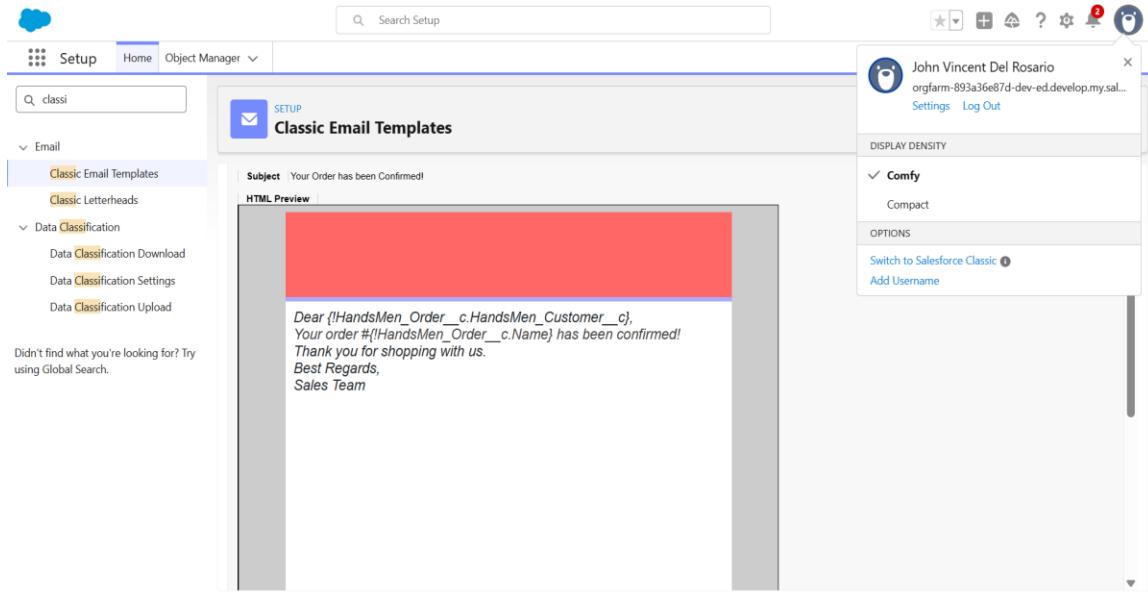
**Environment Setup:** Sandbox and DevOps workflow established.

**Customizations:**

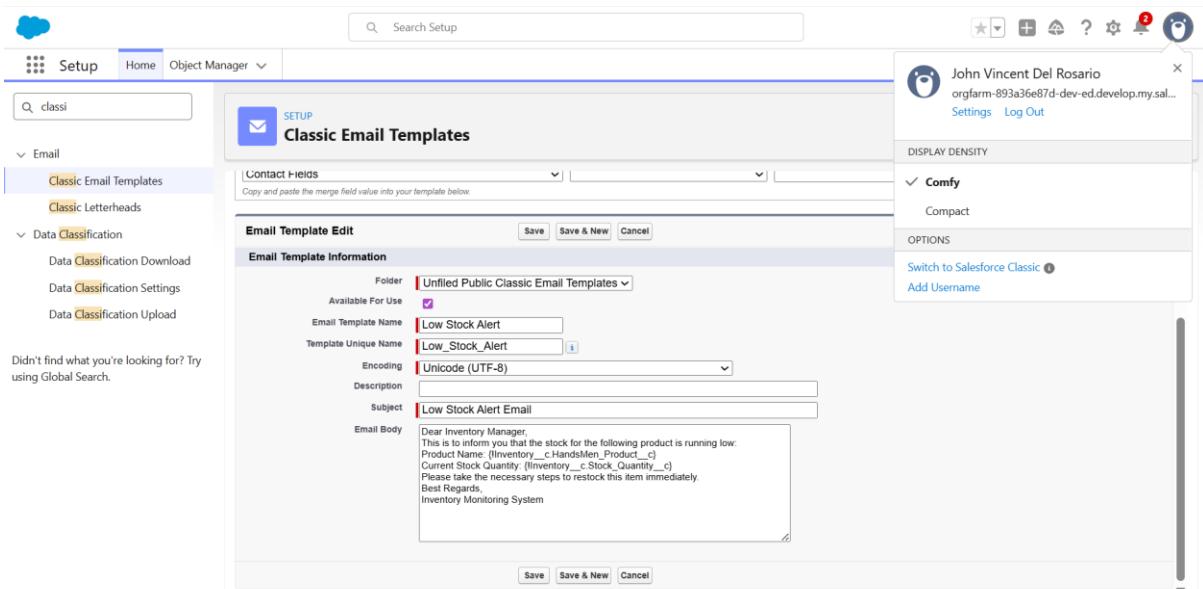
- **Custom objects:**  
*Customer, Booking, Payment, Loyalty Points.*
- **Validation rules:**  
Ensure accurate customer measurements and payment details.
- **Automation:**  
Flows for loyalty point updates, approval processes for bespoke orders.
- **Apex Development:**  
Triggers for automatic loyalty point calculation.
- **Screenshots:**  
(Insert relevant Salesforce setup screenshots).

## Here's the Relevant Screenshots:

### 1. Order Confirmation Email



### 2. Low Stock Alert Email



### 3. Loyalty Points Email

The screenshot shows the Salesforce Setup interface. On the left, the navigation pane is open with the following structure:

- Setup
- Home
- Object Manager
- Email
  - Classic Email Templates
  - Classic Letterheads
  - Data Classification
    - Data Classification Download
    - Data Classification Settings
    - Data Classification Upload

In the center, the "Classic Email Templates" page is displayed. A new template is being edited, titled "Loyalty Program Email". The "Text-Only Email Content" section contains the following text:

```
Loyalty Program Email  
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.
```

On the right side of the screen, there is a sidebar with user information and settings:

- User: John Vincent Del Rosario  
Settings Log Out
- DISPLAY DENSITY: Compact
- OPTIONS: Switch to Salesforce Classic, Add Username

### 4. Order Total Trigger

The screenshot shows the Salesforce Developer Console with the file "OrderTotalTrigger.apex" selected. The code is as follows:

```
trigger OrderTotalTrigger on HandsMen_Order__c (before insert, before update) {
    Set<Id> productIds = new Set<Id>();
    for (HandsMen_Order__c order : Trigger.new) {
        if (order.HandsMen_Product__c != null) {
            productIds.add(order.HandsMen_Product__c);
        }
    }
    Map<Id, HandsMen_Product__c> productMap = new Map<Id, HandsMen_Product__c>(
        [SELECT Id, Price__c FROM HandsMen_Product__c WHERE Id IN :productIds]
    );
    for (HandsMen_Order__c order : Trigger.new) {
        if (order.HandsMen_Product__c != null && productMap.containsKey(order.HandsMen_Product__c)) {
            HandsMen_Product__c product = productMap.get(order.HandsMen_Product__c);
            if (order.Quantity__c != null) {
                order.Total_Amount__c = order.Quantity__c * product.Price__c;
            }
        }
    }
}
```

Below the code editor, there is a logs table and a filter bar at the bottom.

## 5. Stock Deduction Trigger

The screenshot shows the Salesforce IDE interface. The top navigation bar includes File, Edit, Debug, Test, Workspace, and Help. The tabs at the top are 'StockDeductionTrigger.apex' (active), 'InventoryBatchJob.apex', and 'InventoryBatchJob.cls'. The code editor contains the following Apex code:

```
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
33    if (!inventoriesToUpdate.isEmpty()) {
34        update inventoriesToUpdate;
35    }
36}
```

The bottom part of the interface shows the 'Log' tab, which is currently selected. It displays a table with columns: User, Application, Operation, Time, Status, and Result. There are no entries in the log.

## 6. Inventory Batch Job

The screenshot shows the Salesforce IDE interface. The top navigation bar includes File, Edit, Debug, Test, Workspace, and Help. The tabs at the top are 'InventoryBatchJob.apex' (active), 'StockDeductionTrigger.apex', and 'StockDeductionTrigger.cls'. The code editor contains the following Apex code:

```
1 global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {
2
3     global Database.QueryLocator start(Database.BatchableContext BC) {
4         return Database.getQueryLocator(
5             'SELECT Id, Stock_Quantity__c FROM Product__c WHERE Stock_Quantity__c < 10'
6         );
7     }
8
9     global void execute(Database.BatchableContext BC, List<SObject> records) {
10
11         List<HandsMen_Product__c> productsToUpdate = new List<HandsMen_Product__c>();
12
13         // Cast SObject list to Product__c list
14         for (SObject record : records) {
15             HandsMen_Product__c product = (HandsMen_Product__c) record;
16             product.Stock_Quantity__c += 50; // Restock logic
17             productsToUpdate.add(product);
18         }
19
20         if (!productsToUpdate.isEmpty()) {
21             try {
22                 update productsToUpdate;
23             } catch (DmlException e) {
24                 System.debug('Error updating inventory: ' + e.getMessage());
25             }
26         }
27     }
28
29     if (!productsToUpdate.isEmpty()) {
30
31         try {
32             update productsToUpdate;
33         } catch (DmlException e) {
34             System.debug('Error updating inventory: ' + e.getMessage());
35         }
36     }
37 }
```

The bottom part of the interface shows the 'Log' tab, which is currently selected. It displays a table with columns: User, Application, Operation, Time, Status, and Result. There are no entries in the log.

## **7. Inventory Executing**

### **8. Inventory Executing Success**

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

OrderTotalTrigger.apxt StockDeductionTrigger.apxt **InventoryBatchJob.apxc**

Code Coverage: None API Version: 64

```
1 * global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {  
2  
3     * global Database.QueryLocator start(Database.BatchableContext BC) {  
4  
5         return Database.getQueryLocator(  
6  
6             'SELECT Id, Stock_Quantity__c FROM Product__c WHERE Stock_Quantity__c < 10'  
7  
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 }
```

Logs Tests Checkpoints Query Editor View State Progress Problems

User	Application	Operation	Time	Status
Education Matters	Unknown	/services/data/v64.0/tooling/executeA...	7/15/2025, 12:37:32 AM	Success

Filter Click here to filter the log list

## Phase 3: UI/UX Development & Customization

- **Lightning App:**  
**Handsmen Threads CRM app created via App Manager.**
- **Page Layouts & Dynamic Forms:**  
**Tailored layouts for customer profiles and bookings.**
- **Reports & Dashboards:**  
**Sales performance, customer loyalty trends, booking pipeline.**
- **LWC Development:**  
**Optional components for real-time order status.**

## Phase 4: Data Migration, Testing & Security

- **Data Migration:**  
**Imported customer records via Data Loader.**
- **Security:**  
**Profiles for Admin, Sales, Tailor; role hierarchy for managers.**
- **Testing:**  
**Test cases for booking creation, loyalty point updates, approval workflows.**

# Date Model and Key Fields

## Handsman Customer

- **FullName**
- **Email**
- **Phone**
- **Loyalty\_Status\_c**
- **Total Purchases\_c**

## Handsman Product

- **SKU\_c**
- **Name**
- **Price\_c**
- **Stock\_Quantity\_c**

## Inventory

- **HandsMen\_Product\_c(Master Detail)**
- **CreatedById**
- **Warehouse**
- **Stock\_Quantity\_c**
- **Stock\_Status\_c**

## Handsman Orders

- **Quantity\_c**
- **Customer\_Email\_c**
- **HandsMen\_Customer\_c (lookup)**
- **Status\_c (Pending Confirmed Cancelled)**
- **HandsMen\_Product\_c (lookup)**
- **Name (Order Number)**