

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

**Salesforce Talent Accelerator Program
(STAP)**

Submitted by:
Naura Hirzia Nawarika

Universitas Pendidikan Indonesia

A. PROJECT OVERVIEW

To assist with operational efficiency and customer experience, HandsMen Threads built a fully customized Salesforce CRM. Most processes related to customer data, orders, inventory, and marketing campaigns are streamlined through the brand's internal systems.

At the center system's structure is a data model with five main custom objects: Customer, Order, Product, Inventory, and Marketing Campaign. This structure facilitates important workflows and uniformity of data across business activities.

The CRM uses several automation features, such as Record-Triggered Flows, Scheduled Flows, Email Alerts, and Apex processes. These automations enable seamless order handling, timely customer notifications, loyalty program management, and proactive inventory oversight, reducing the need for manual work.

The systems use role-based access for Sales, Inventory, and Marketing teams to secure data and restrict access appropriately. Inventory management is assisted with a Scheduled Apex Batch that updates stock status to automate marginal availability and to keep the uninterrupted status.

By combining automation, structured data management, and secure access controls, the Salesforce CRM implementation strengthens HandsMen Threads' ability to engage customers effectively, optimizes internal operations, and establishes a scalable framework for future growth.

B. OBJECTIVE

This project is focused on implementing a customized Salesforce CRM for HandsMen Threads, designed to centralize customer, order, product, inventory, and marketing data into a single, cohesive platform. The system seeks to streamline operations, enhance data accuracy, and deliver improved customer engagement.

The specific objectives of the project are to:

- **Enhance customer management** by consolidating profiles, purchase history, and interactions to enable more effective tracking and personalized engagement.

- **Streamline order and inventory processes** through automation, including real time stock monitoring and notifications to minimize manual effort.
- **Increase operational efficiency** by standardizing workflows and reducing repetitive tasks across Sales, Inventory, and Marketing teams.
- **Support secure and collaborative team operations** with role-based access controls and coordinated processes to ensure data integrity and smooth interdepartmental collaboration.

C. DETAILED EXECUTION OF PROJECT PHASE

a. Requirement Analysis & Planning

The first step focused on evaluating the challenges the Salesforce CRM can solve. The developer identified essential needs, including accurate tracking of customers, orders, and products, effective inventory management, improved customer communication through timely notifications, a loyalty program, and automation of repetitive tasks to enhance efficiency.

Based on this analysis, the project scope was defined to include customer and order management, inventory tracking, a loyalty program, automated notifications, and reporting dashboards, while excluding physical tailoring work, manual delivery tracking, and unrelated software. The goals of the project include unifying business data on a single system, automating important processes, improving customer communication with reward personalization, and ensuring the company has consistent and reliable data to make better business decisions.

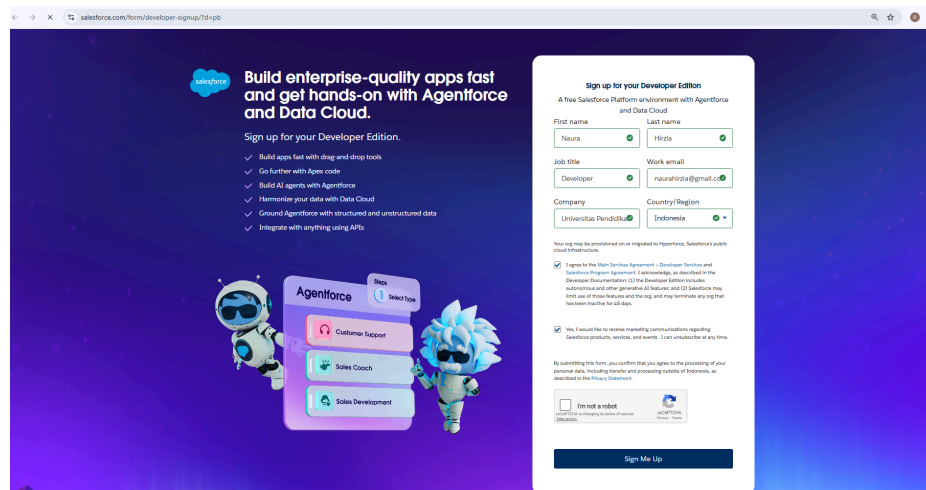
A comprehensive data model was developed, connecting Orders, Products, Customers, and Loyalty Programs, while a security model was implemented with roles, profiles, and field-level access to protect sensitive information and ensure smooth workflows. Primary stakeholders were also selected, which included the Sales Team for operational frontline and Inventory Managers for stock level oversight. This design ensures the CRM enhances operational efficiency, data security, and overall stakeholder satisfaction.

b. Salesforce Development - Backend & Configuration

i. Environment Setup & DevOps Workflow

The project was built and tested within a Salesforce Developer Org, a personal workspace that allows safe configuration of objects, fields, automation, and Apex code.

The developer created the org via [Salesforce Developer Signup](#), completed account verification, and gained access to the Setup interface. This environment served as the main platform for developing, testing, and validating all CRM components before deployment to the live system.



ii. Custom Objects and Fields

- **HandsMen Customer** : Stores customer details: Name, Email, Phone, Loyalty_Status__c (Bronze, Silver, Gold), Total_Purchases__c
- **HandsMen Product** : Stores product information: Name, SKU, Price, Stock_Quantity__c
- **HandsMen Order** : Tracks customer orders: Order_Number, Status (Pending, Confirmed, Rejected), Quantity__c, Total_Amount__c
- **Inventory** : Monitors stock levels: Auto Number, Stock_Quantity__c, Warehouse

- **Marketing Campaign** : Manages promotions: Campaign_Name, Start_Date, End_Date

iii. Validation Rules

1) Inventory Stock Quantity

Prevents saving any inventory record with a stock quantity of zero or less. If someone tries, Salesforce stops the action and displays the error message at the top of the page, ensuring inventory data remains accurate.

Rule Name: Stock_Quantity

Object: Inventory

Formula: Stock_Quantity__c <= 0

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

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
Flow Triggers
Validation Rules
Conditional Field Formatting

Inventory Validation Rule

Define a validation rule by specifying an error condition and a corresponding error message. The error condition is written as a Boolean formula

Validation Rule Edit [Save] [Save & New] [Cancel]

Rule Name:

Active: ☒

Description:

Error Condition Formula

Example: [More Examples...](#)
Display an error if Discount is more than 30%
If this formula expression is true, display the text defined in the Error Message area

Functions
-- All Function Categories --
ABS
ACOS
ADDMONTHS
AND
ASCII
ASIN

ABS(number)
Returns the absolute value of a number, a number without its sign
[Help on this function](#)

Error Message

Example:
This message will appear when Error Condition formula is true

Error Message:

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

Error Location: ☒ Top of Page ☐ Field

[Save] [Save & New] [Cancel]

2) Customer Email

Prevents saving any customer record that does not contain a valid Gmail address. If the entered email lacks “@gmail.com”, Salesforce stops the save and displays the error message, ensuring only valid Gmail addresses are recorded.

Rule Name: Email

Object: HandsMen Customer

Formula: NOT CONTAINS(Email__c , "@gmail.com")

Error Message: “Please fill Correct Gmail”

The screenshot shows the Salesforce Setup interface for the 'HandsMen Customer' object. The left sidebar contains a navigation menu with options like Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, Scoping Rules, Object Access, Triggers, Flow Triggers, Validation Rules (highlighted), and Conditional Field Formatting. The main content area is titled 'HandsMen Customer Validation Rule' and includes a description: 'Define a validation rule by specifying an error condition and a corresponding error message. The error condition is written as a Boolean formula.' Below this is the 'Validation Rule Edit' section with fields for Rule Name (Email), Active (checked), and Description. The 'Error Condition Formula' section shows an example formula 'Discount_Percent__c > 0.30' and the current formula 'NOT CONTAINS(Email__c , "@gmail.com")'. A 'Functions' dropdown menu is open, showing a list of functions including ABS, ACOS, ADDMONTHS, AND, ASCII, and ASIN. The 'Error Message' section shows an example message 'Discount percent cannot exceed 30%' and the current message 'Please fill Correct Gmail'. The 'Error Location' is set to 'Top of Page'.

3) Total Amount

Prevents saving any order with a total amount of zero or less. If a user tries to save such an order, Salesforce stops the action and displays the error message next to the Total Amount field, ensuring all orders have a valid total.

Rule Name: Total_Amount

Object: HandsMen Order

Formula: Total_Amount__c <= 0

Error Message: “Please Enter Correct Amount”

SETUP > OBJECT MANAGER

HandsMen Order

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Scoping Rules

Object Access

Triggers

Flow Triggers

Validation Rules

Conditional Field Formatting

HandsMen Order Validation Rule

Define a validation rule by specifying an error condition and a corresponding error message. The error condition is written as a Boolean formula.

Validation Rule Edit [Save] [Save & New] [Cancel]

Rule Name: Total_Amount

Active: ☒

Description:

Error Condition Formula

Example: Discount_Percent__c > 0.30 [More Examples...](#)

Display an error if Discount is more than 30%

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

[Insert Field] [Use Formula Assistant] [Insert Operator]

Total_Amount__c <= 0

[Check Syntax]

Functions

-- All Function Categories --

ABS

ACOS

ADDMONTHS

AND

ASCII

ASIN

[Insert Selected Function]

ABS(number)

Returns the absolute value of a number, a number without its sign

[Help on this function](#)

Error Message

Example: Discount percent cannot exceed 30%

This message will appear when Error Condition formula is true

Error Message: Please Enter Correct Amount

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

Error Location: ☐ Top of Page ☒ Field Total Amount

[Save] [Save & New] [Cancel]

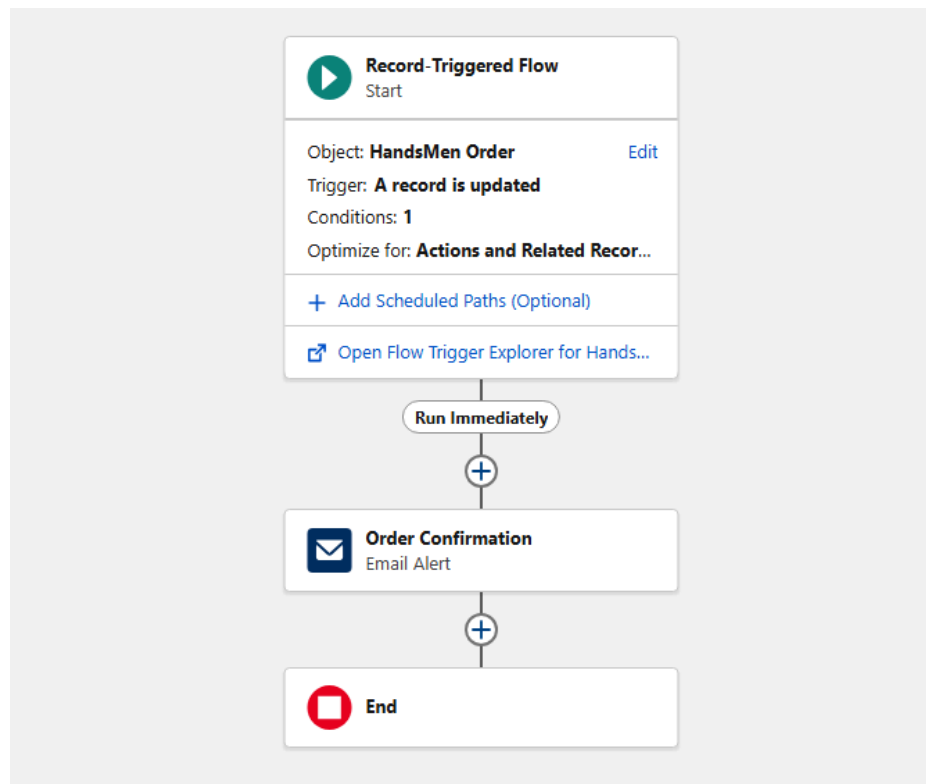
iii. Automation (Flows, Workflow Rules, Approval Processes)

1) Order Confirmation Flow

Automatically sends a confirmation email to the customer once their order is confirmed, ensuring timely notification without manual intervention.

Type: Record-Triggered Flow

Trigger: When an order's status is Confirmed

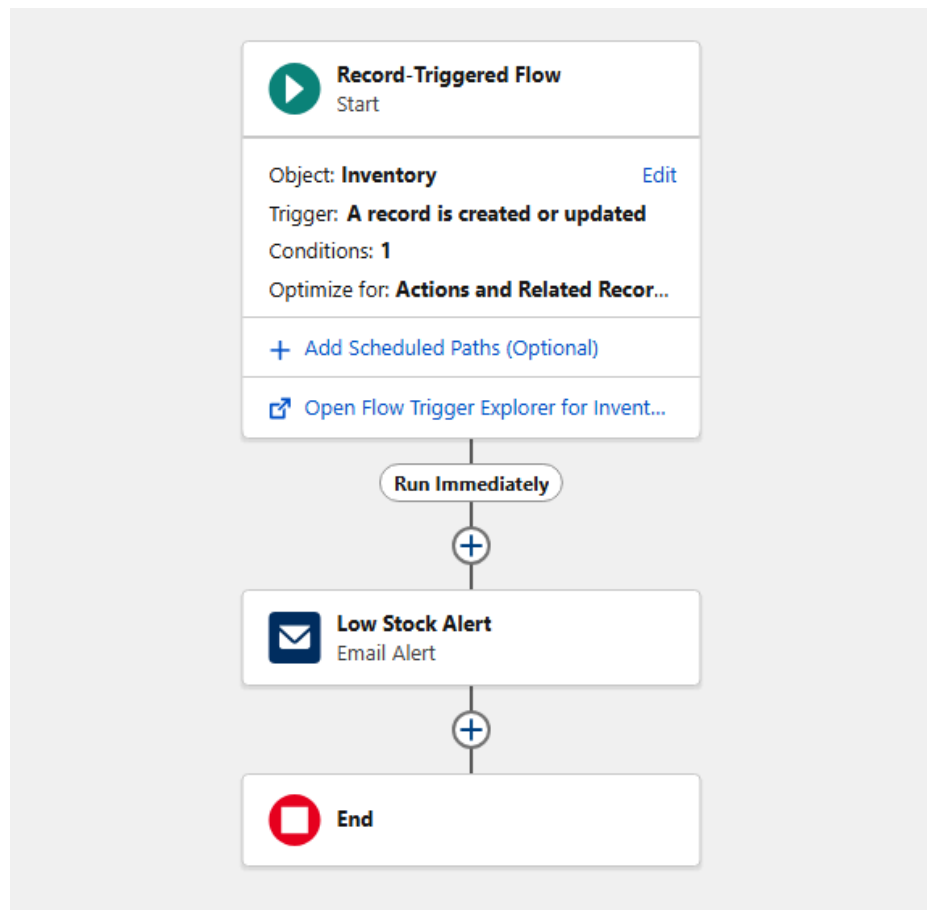


2) Stock Alert Flow

Sends an alert to the staff to restock the item before it runs out, preventing stockouts and ensuring smooth operations.

Type: Record-Triggered Flow

Trigger: When an inventory item's stock quantity drops below 5

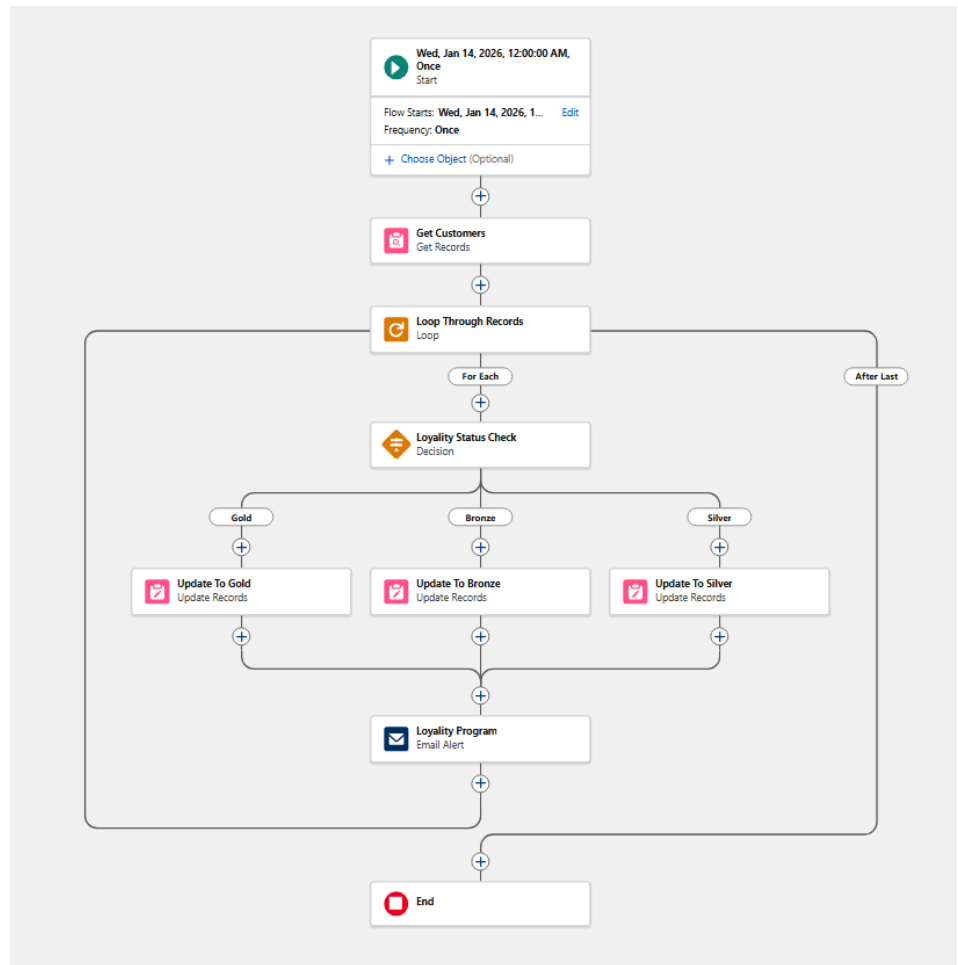


3) Loyalty Status Update Flow

Automatically updates the Loyalty_Status__c field for customers based on their points or activity, keeping the loyalty program current without manual input.

Type: Scheduled Flow

Trigger: Runs daily

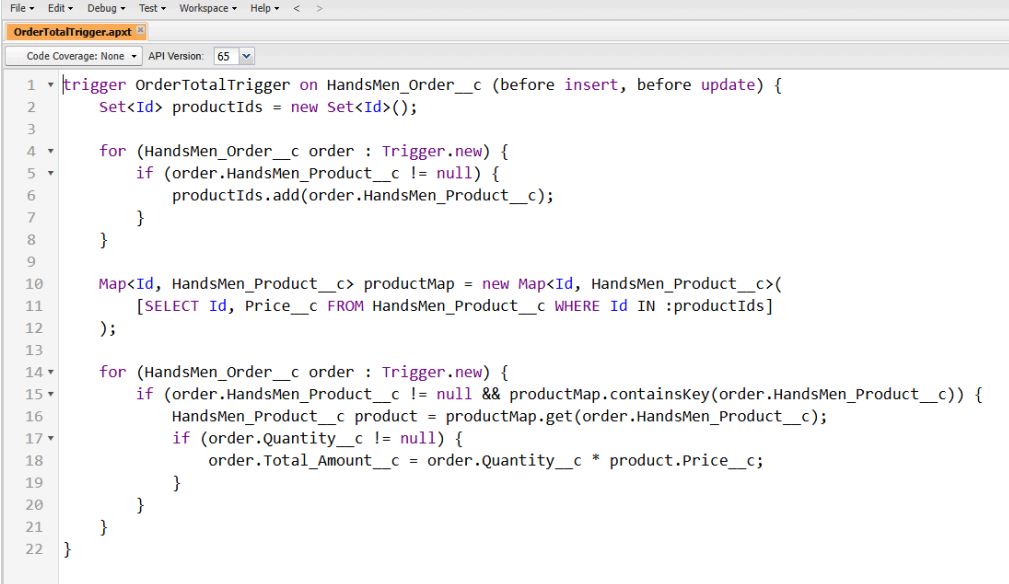


iv. Apex Classes & Triggers

1) OrderTotalTrigger

- Purpose:** Automatically calculates the total amount of each order based on product price and quantity. Ensures the Total_Amount__c field is accurate before saving, preventing manual errors.
- How It Works:**
 - Collect Product IDs:** The trigger scans incoming orders during insert or update. Product IDs from orders are added to a set to identify which products need pricing information.

2. **Query Product Prices:** Products in the set are queried for their Price__c values. Prices are stored in a map (productMap) with product IDs as keys for efficient lookup.
3. **Calculate Total Amount:** The trigger loops through orders again, retrieves product prices from the map, and multiplies by the order quantity to compute the total amount.



```

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 }

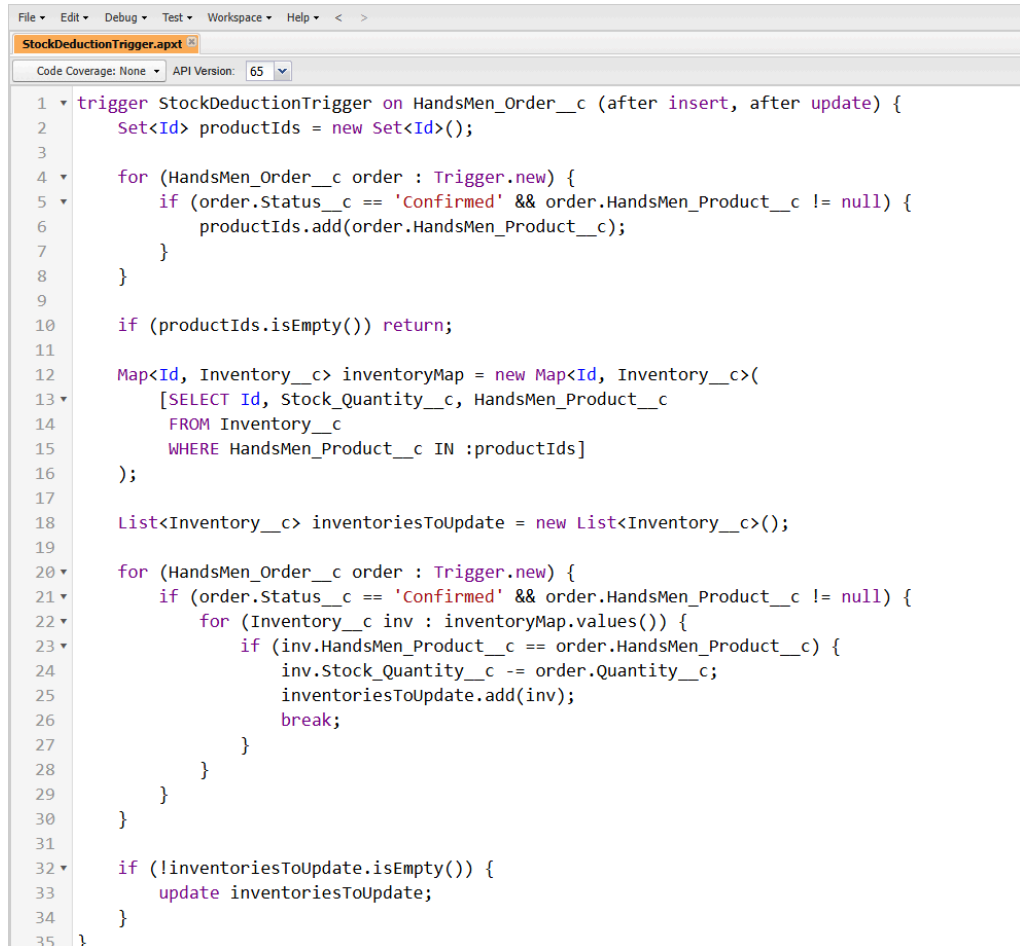
```

2) StockDeductionTrigger

- **Purpose:** Automatically reduces inventory stock when an order is confirmed, ensuring real-time stock accuracy.
- **How It Works:**
 1. **Collect Product IDs:** Scan incoming orders during insert or update; confirmed orders with selected products have their product IDs added to a set to identify relevant inventory records.
 2. **Retrieve Inventory Records:** Queries all corresponding Inventory__c records and stores them in a map keyed by product ID for efficient access.
 3. **Deduct Stock Quantities:** Loops through confirmed orders, subtracts the order quantity from the inventory's

Stock_Quantity__c, and adds updated records to a list.

4. **Update Inventory:** Performs a single update operation for all modified inventory records to maintain accurate stock levels without manual intervention.



```
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    Map<Id, Inventory__c> inventoryMap = new Map<Id, Inventory__c>{
13        [SELECT Id, Stock_Quantity__c, HandsMen_Product__c
14         FROM Inventory__c
15         WHERE HandsMen_Product__c IN :productIds]
16    };
17
18    List<Inventory__c> inventoriesToUpdate = new List<Inventory__c>();
19
20    for (HandsMen_Order__c order : Trigger.new) {
21        if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
22            for (Inventory__c inv : inventoryMap.values()) {
23                if (inv.HandsMen_Product__c == order.HandsMen_Product__c) {
24                    inv.Stock_Quantity__c -= order.Quantity__c;
25                    inventoriesToUpdate.add(inv);
26                    break;
27                }
28            }
29        }
30    }
31
32    if (!inventoriesToUpdate.isEmpty()) {
33        update inventoriesToUpdate;
34    }
35 }
```

v. User Roles & Profiles

A new profile, Platform1, was created by copying the Standard User profile and granted access to all necessary custom objects.

Roles were defined for different departments to control data visibility:

- Sales Manager
- Inventory Manager
- Marketing Team

Setup Profiles												
Custom Object Permissions												
	Custom Objects				Custom Permissions				Custom Permissions			
	Read	Create	Edit	Delete	View All Records	Modify All Records	View All Fields		Read	Create	Edit	Delete
Handoff Customers	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Handoff Orders	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Handoff Products	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Session Settings												

vi. User Accounts

New users were created in Salesforce and assigned appropriate roles and profiles:

- Niklaus Mikaelson – Sales Manager
- Kol Mikaelson – Inventory Manager
- Lila Mikaelson – Marketing

These role and profile configurations ensure that users can access only the data relevant to their responsibilities, maintaining system security and proper workflow.

SETUP

Users

All Users

On this page you can create, view, and manage users.

To get more licenses, use the Your Account app. [Let's Go](#)

View:

All Users

 | [Edit](#) | [Create New User](#)

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

<input type="checkbox"/>	Action	Full Name	Alias	Username	Role	Active	Profile
<input type="checkbox"/>	E04	Charles Fawcett	Charles	charles.0045000001@salesforce.com		<input checked="" type="checkbox"/>	Charles Fawcett User
<input type="checkbox"/>	E04	Charles Fawcett	Charles	charles.1421564040@salesforce.com		<input checked="" type="checkbox"/>	System Administrator
<input type="checkbox"/>	E04	Mikaelson, Niklaus	Niklaus	niklaus.0045000001@salesforce.com		<input checked="" type="checkbox"/>	System Administrator
<input type="checkbox"/>	E04	Mikaelson, Kol	Kol	kol.0045000001@salesforce.com	Inventory	<input checked="" type="checkbox"/>	Platform_1
<input type="checkbox"/>	E04	Mikaelson, Lila	Lila	lila.0045000001@salesforce.com	Marketing	<input checked="" type="checkbox"/>	Platform_1
<input type="checkbox"/>	E04	Mikaelson, Niklaus	Niklaus	niklaus.0045000001@salesforce.com	Sales	<input checked="" type="checkbox"/>	Platform_1
<input type="checkbox"/>	E04	User, Integration	Integration	integration.0045000001@salesforce.com		<input checked="" type="checkbox"/>	Analytics Cloud Integration User
<input type="checkbox"/>	E04	User, Security	Security	security.0045000001@salesforce.com		<input checked="" type="checkbox"/>	Analytics Cloud Security User

New User

Reset Password(s)

Add Multiple Users

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

vii. Email Templates & Alerts

- **Order Confirmation Email:** Sent automatically when an order status changes to Confirmed.
- **Low Stock Alert:** Sent when an inventory item's stock drops below 5 units.
- **Loyalty Program Update:** Sent when a customer's loyalty status is updated.

Automation: Email alerts were linked to the corresponding flows to ensure notifications are sent accurately and on time.

The screenshot shows the 'Classic Email Templates' setup page in Salesforce. The template is named 'Order_Confirmation_Email'. It is a 'Unified Public Classic Email Template' that is 'Available For Use'. The 'Template Unique Name' is 'Order_Confirmation_Email', the 'Classic Letterhead' is 'HandsMen Threads', and the 'Email Layout' is 'Free Form Letter'. The 'Encoding' is 'Unicode (UTF-8)' and the 'Author' is 'Naura Hogg'. The 'Description' is 'Naura Hogg (Chatter)'. It was 'Created By' Naura Hogg on 1/13/2020 at 4:01 PM and 'Modified By' Naura Hogg on 1/13/2020 at 4:01 PM. Below the details, there is an 'Email Template' section with a 'Send Test and Verify Merge Fields' button. The 'HTML Preview' shows a confirmation email with the subject 'Your Order has been Confirmed!', a greeting 'Dear (HandsMen_Order__c.HandsMen_Customer__c)', and a message stating 'Your order #(HandsMen_Order__c.Name) has been confirmed!'. It also includes 'Thank you for shopping with us.', 'Best Regards,', and 'Sales Team'.

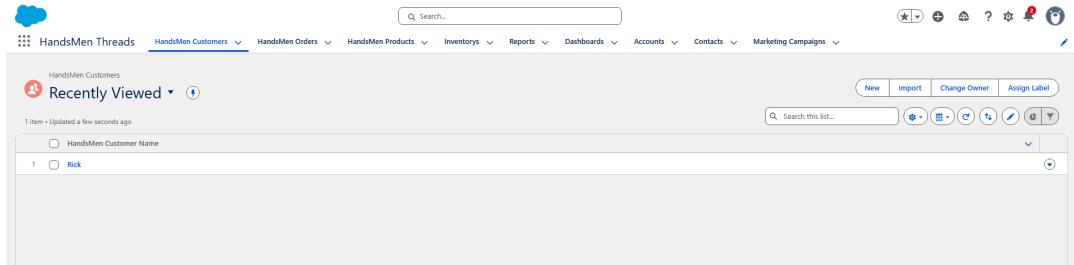
The screenshot shows the 'Email Alerts' setup page in Salesforce. The alert is named 'Order Confirmation Email Alert'. It is linked to the 'Order_Confirmation_Email' template. The 'Description' is 'Order Confirmation Email Alert', the 'Unique Name' is 'Order_Confirmation_Email_Alert', the 'From Email Address' is 'Current User's email address', and the 'Recipients' are 'Email Field: Customer Email'. It was 'Created By' Naura Hogg on 1/13/2020 at 4:06 PM and 'Modified By' Naura Hogg on 1/13/2020 at 4:06 PM. Below the details, there are sections for 'Rules Using This Email Alert', 'Approval Processes Using This Email Alert', and 'Entitlement Processes Using This Email Alert', all of which currently show no associated rules or processes. At the bottom, there is a 'Flows Using This Email Alert' table with one entry: 'Order Confirmation' (Version 1, Description: 'Order Confirmation', Object: '01H0000081evty', Active: checked).

c. UI/UX Development & Customization

A custom Lightning App named HandsMen Threads was created using Salesforce App Manager to centralize access to all CRM objects and features. The app was configured with a meaningful name and icon, while default colors and utility items were maintained.

Navigation items, including HandsMen Customer, HandsMen Order, Inventory, HandsMen Product, Reports, Dashboards, Account, Contact, and Marketing Campaign, were added to provide users with quick access to relevant data.

The System Administrator profile was granted access to the app, ensuring that authorized users could efficiently view and use its functionality, resulting in organized navigation and streamlined CRM operations.



D. PROJECT EXPLANATION WITH REAL-WORLD EXAMPLE

a. Customer Registration

A customer, Rika Sancheez visits the store

Customer Name : Rika Sancheez

Email : rika332@delta.com

Other Details : Add phone number

Validation Rule : “Please fill Correct Gmail” (e.g., must contain @gmail.com)

Beth entered naurahirzia@gmail.com , which meets the Gmail validation requirement

b. Product Setup

The admin adds product like Leather Jacket, Windbreaker, etc., into the Product__c object.

Each product has a price and other details.

Inventory is also created to manage stock for the products.

c. Order Placement

Rika decides to **buy 50 Wool Jacket (\$42 per piece)**

In Salesforce, a new order record is created

Apex Trigger: Automatically calculates `Total_Amount__c` = $50 \times 42 = \$2100$

d. Inventory Update

Apex Trigger: reduced Wool Jacket stock by 50 pieces

Validation Rule: Ensure stock never goes below 0

e. Loyalty Program

Rika now has a **total purchase of \$2100**

A trigger on Customer checks her total purchases

Based on the value:

<\$500 Bronze

\$500-\$1000 Silver

\$1000 Gold

Apex Trigger: Evaluates cumulative purchases and updates `Loyalty_Status__c` field

So, Beth becomes a **Gold member**

f. Email Notification

When a new order is placed or royalty status is updated:

Flow + Email Alerts is triggeres

E. SCREENSHOTS

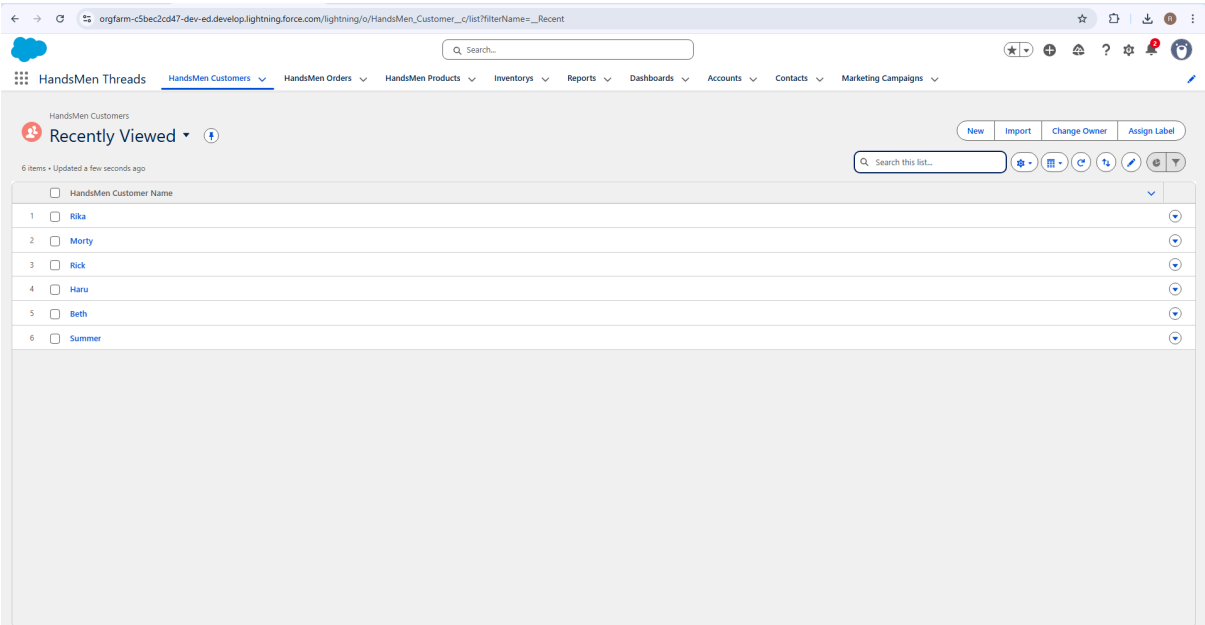


Fig: Custom App for HandsMen Threads

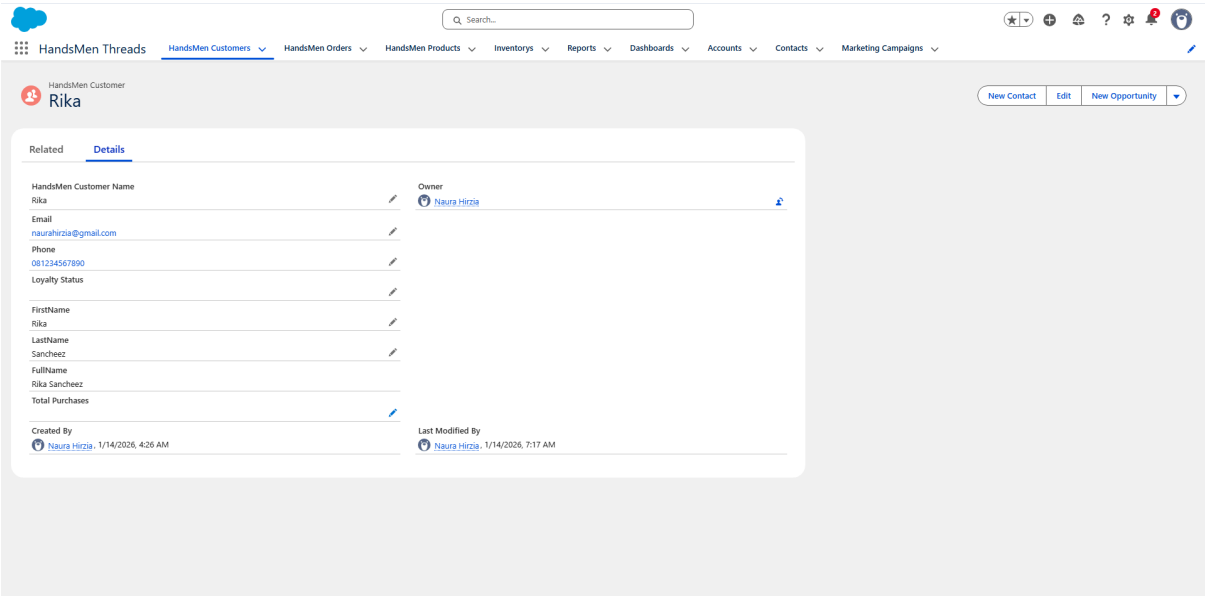


Fig: Customer Creation in HandsMen Threads

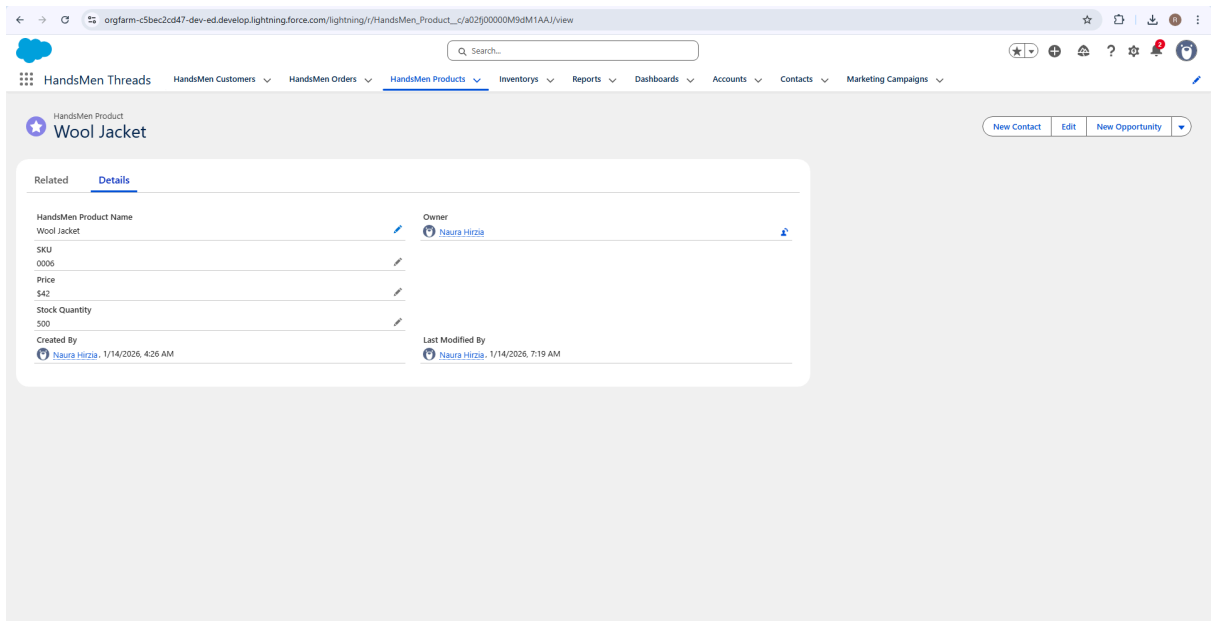


Fig: Products in HandsMen Threads

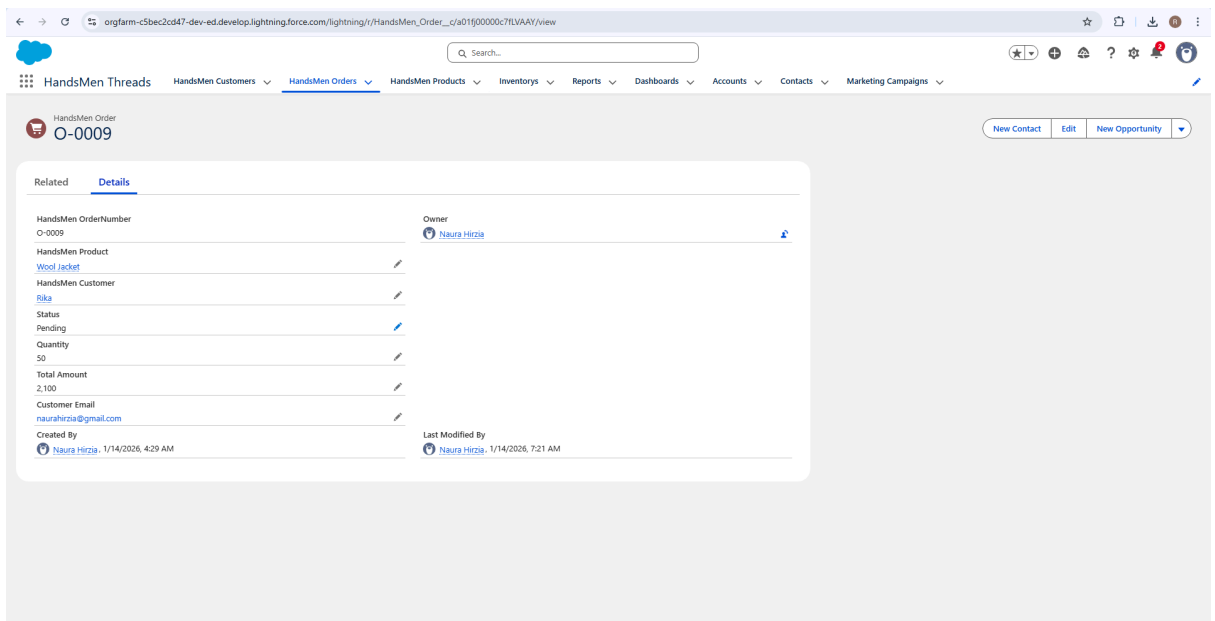


Fig: Order Confirmation

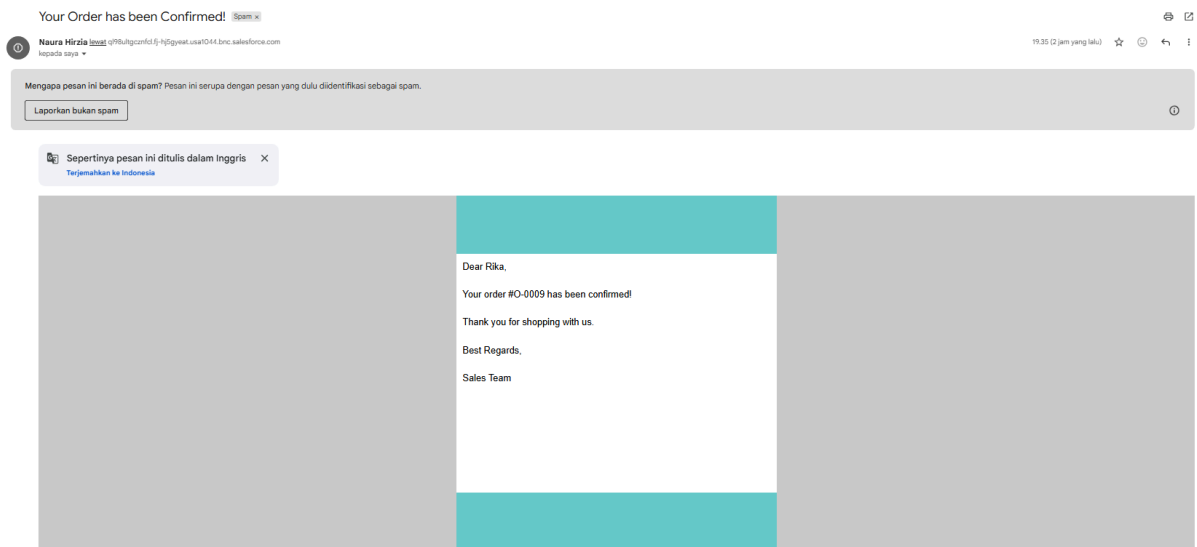


Fig: Order Confirmation Email

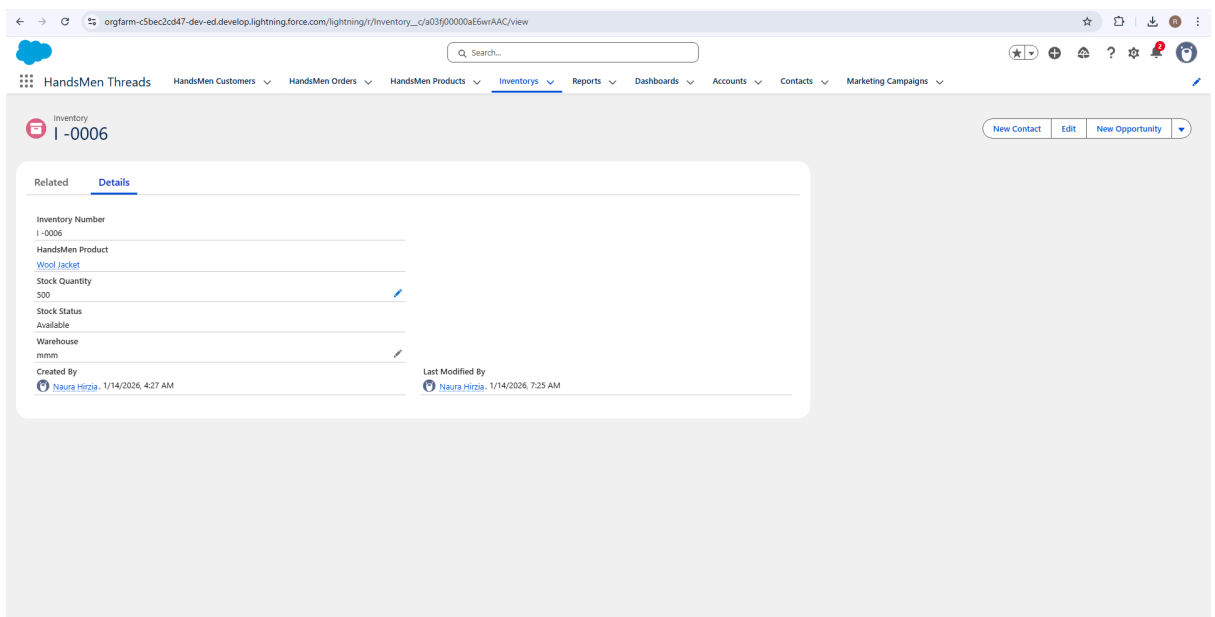


Fig: Inventory Creation

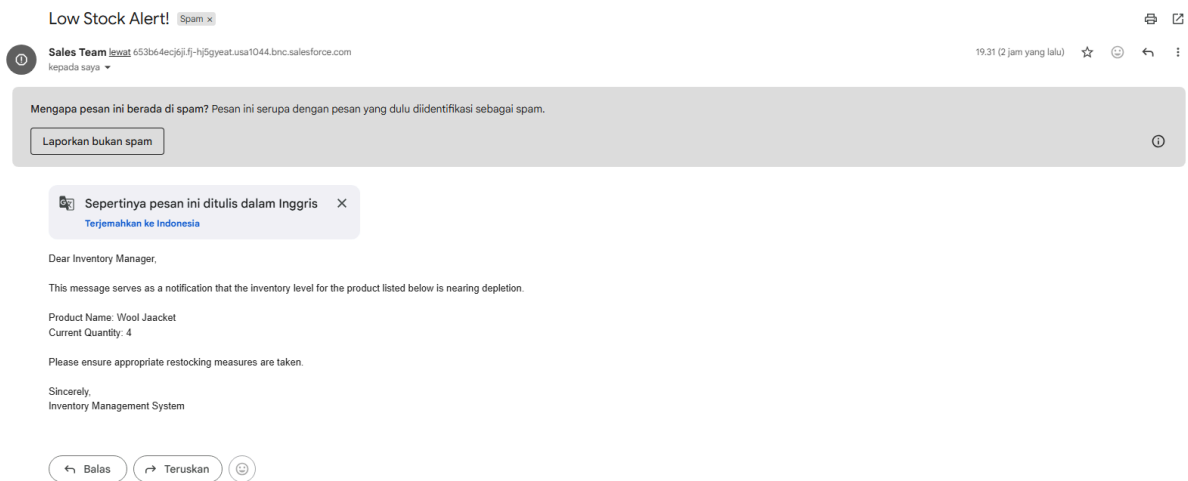


Fig: Low Stock Alert Email

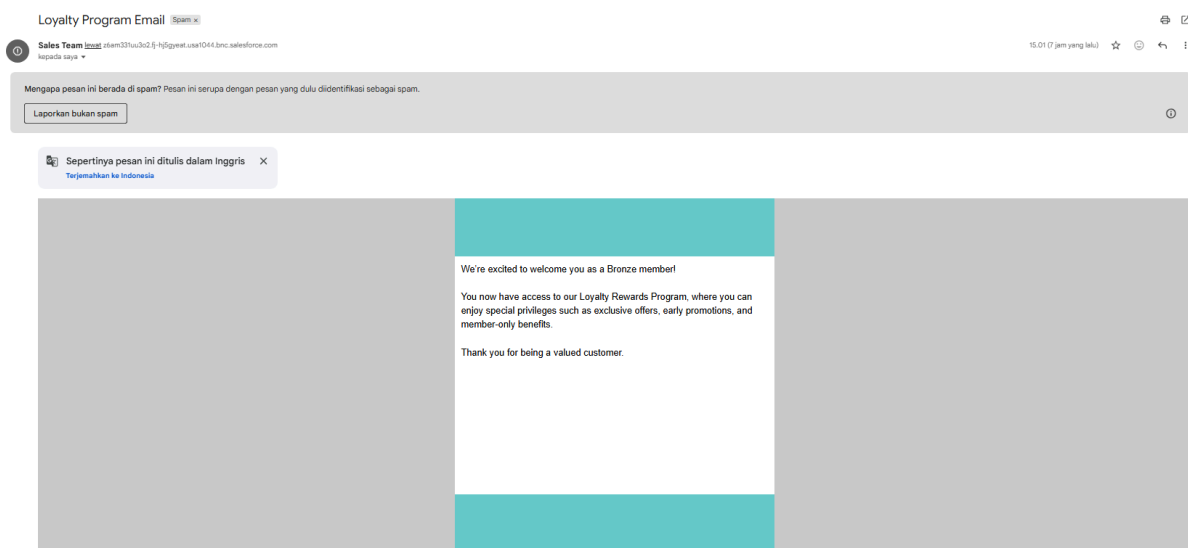


Fig: Loyalty Program Email

F. CONCLUSION

The HandsMen Threads Salesforce CRM project delivered a customized system that centralizes customer, product, order, inventory, and marketing data while automating key business processes. Using structured data models, validation rules, and automations such as Flows, Apex Triggers, and Email

Alerts, the system ensures accurate order processing, timely customer communication, and consistent inventory and loyalty updates.

The CRM also strengthens operational efficiency through role-based access controls, secure data management, and centralized reporting. Overall, the solution provides a scalable and reliable platform that enhances decision-making, improves customer engagement, and supports the long-term growth of HandsMen Threads.

G. FUTURE SCOPE

a. AI Chatbots and Customer Self-Service

Deploy intelligent chatbots and self-service portals to assist customers with inquiries, order tracking, and product recommendations, enhancing customer convenience and support efficiency.

b. Supplier and Procurement Integration

Connect the CRM with supplier systems to automate restocking alerts, purchase orders, and supply chain visibility.

c. E-commerce Platform Integration

Integrate the CRM with online sales platforms to enable real-time order synchronization, automated inventory updates, and seamless customer purchase tracking.

d. Mobile CRM Enablement

Introduce mobile-optimized features that allow sales and inventory teams to manage orders, track stock levels, and access customer information in real time from any location.

e. Advanced Marketing Automation

Strengthen marketing capabilities through automated email campaigns, social media integration, customer segmentation, and performance analytics to improve campaign effectiveness.

f. Customer Chat and Messaging Integration

Integrate chat-based communication channels such as Salesforce Messaging, live chat, or third-party platforms (e.g., WhatsApp) to enable real-time customer support, order updates, and direct engagement within the CRM system.