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.

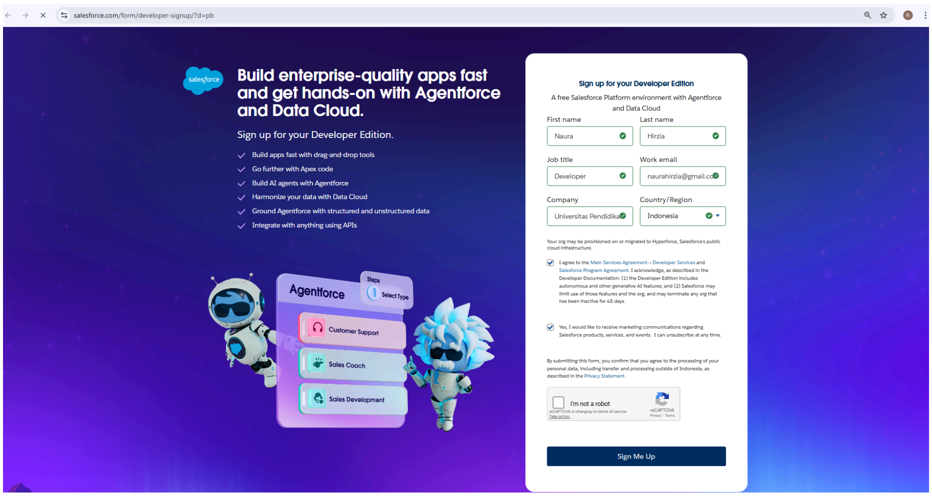
A comprehensive data model was developed, connecting Orders, Products, Customers, and Loyalty Programs, while a security model was implemented with roles, profiles, and field-level access to protect sensitive information and ensure smooth workflows. Primary stakeholders were also selected, which included 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

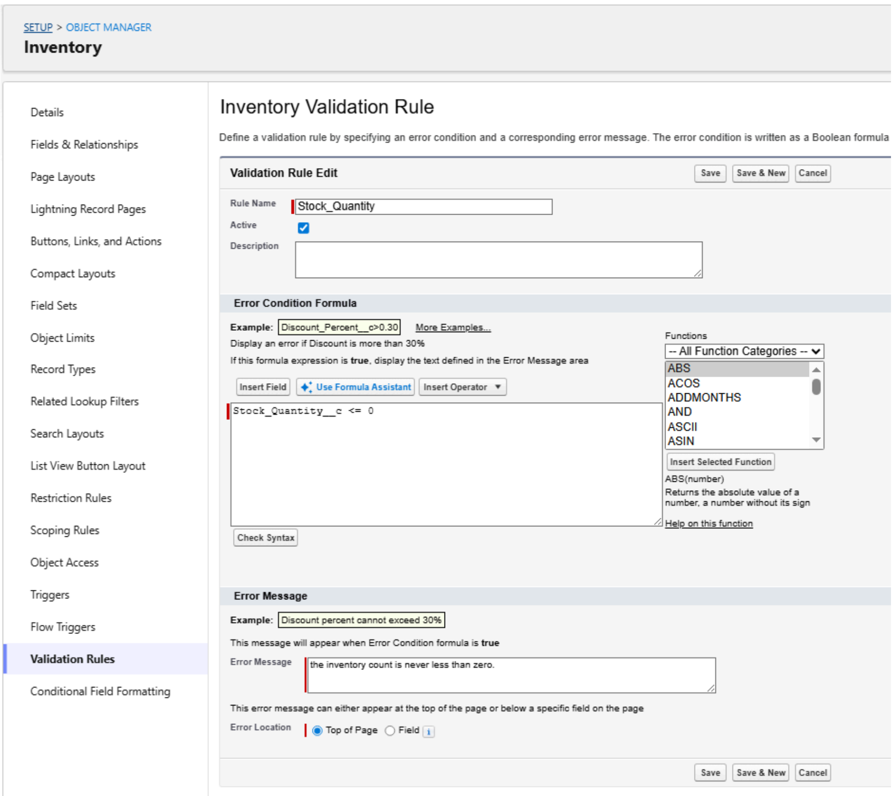
Prevents saving any inventory record with a stock quantity of zero or less. If someone tries, Salesforce stops the action and displays the error message at the top of the page, ensuring inventory data remains accurate.

Rule Name: Stock_Quantity

Object: Inventory

Formula: Stock_Quantity__c <= 0

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



2) Customer Email

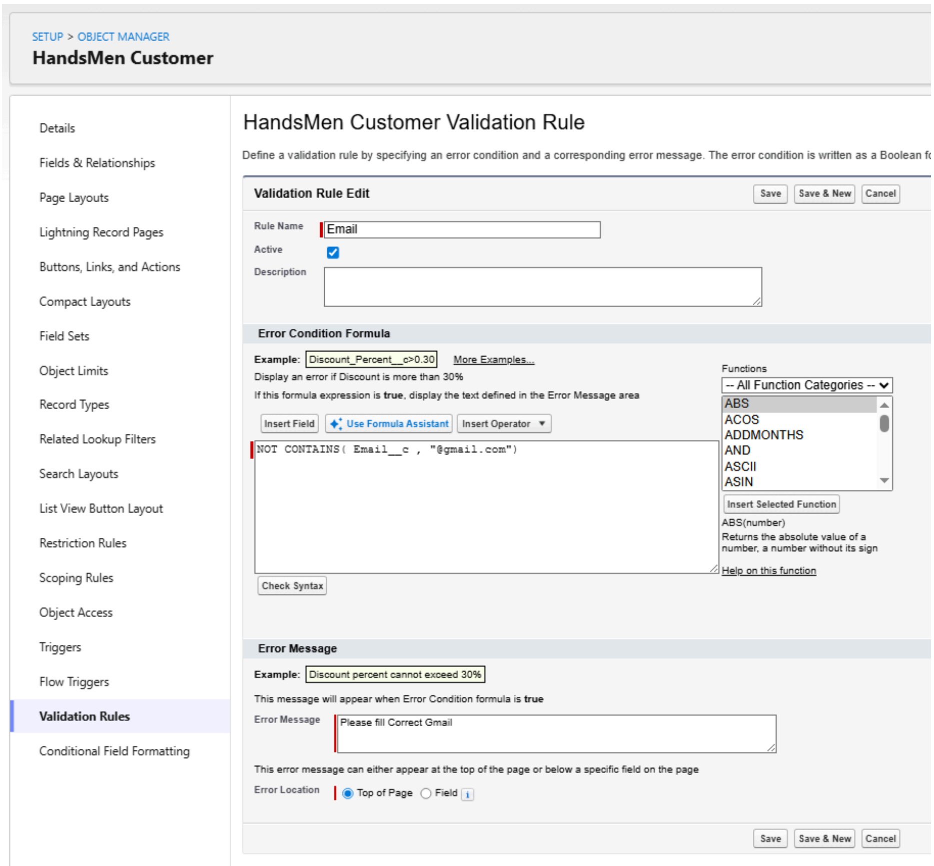
Prevents saving any customer record that does not contain a valid Gmail address. If the entered email lacks “@gmail.com”, Salesforce stops the save and displays the error message, ensuring only valid Gmail addresses are recorded.

Rule Name: Email

Object: HandsMen Customer

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

Error Message: “Please fill Correct Gmail”



3) Total Amount

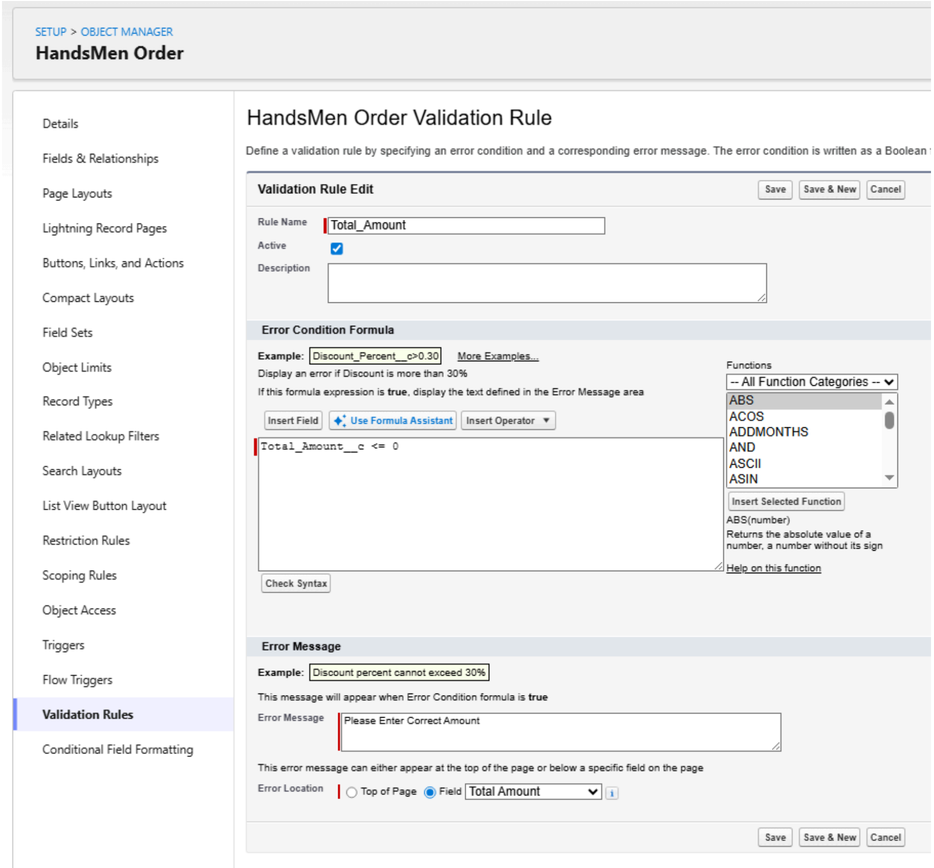
Prevents saving any order with a total amount of zero or less. If a user tries to save such an order, Salesforce stops the action and displays the error message next to the Total Amount field, ensuring all orders have a valid total.

Rule Name: Total_Amount

Object: HandsMen Order

Formula: Total_Amount__c <= 0

Error Message: “Please Enter Correct Amount”

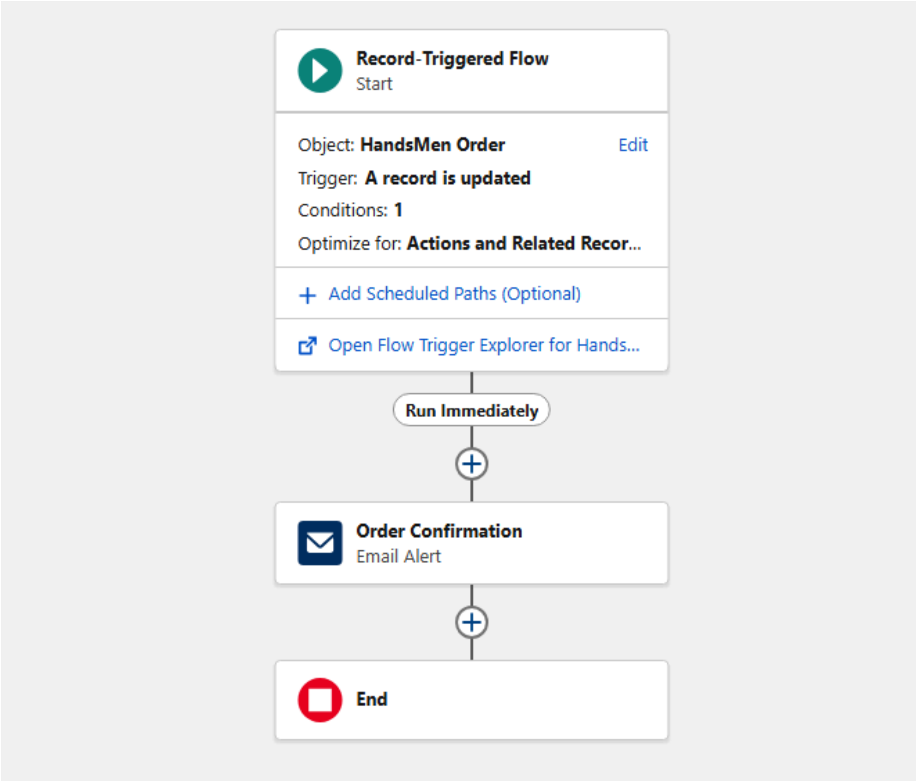


iiii. Automation (Flows, Workflow Rules, Approval Processes)

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

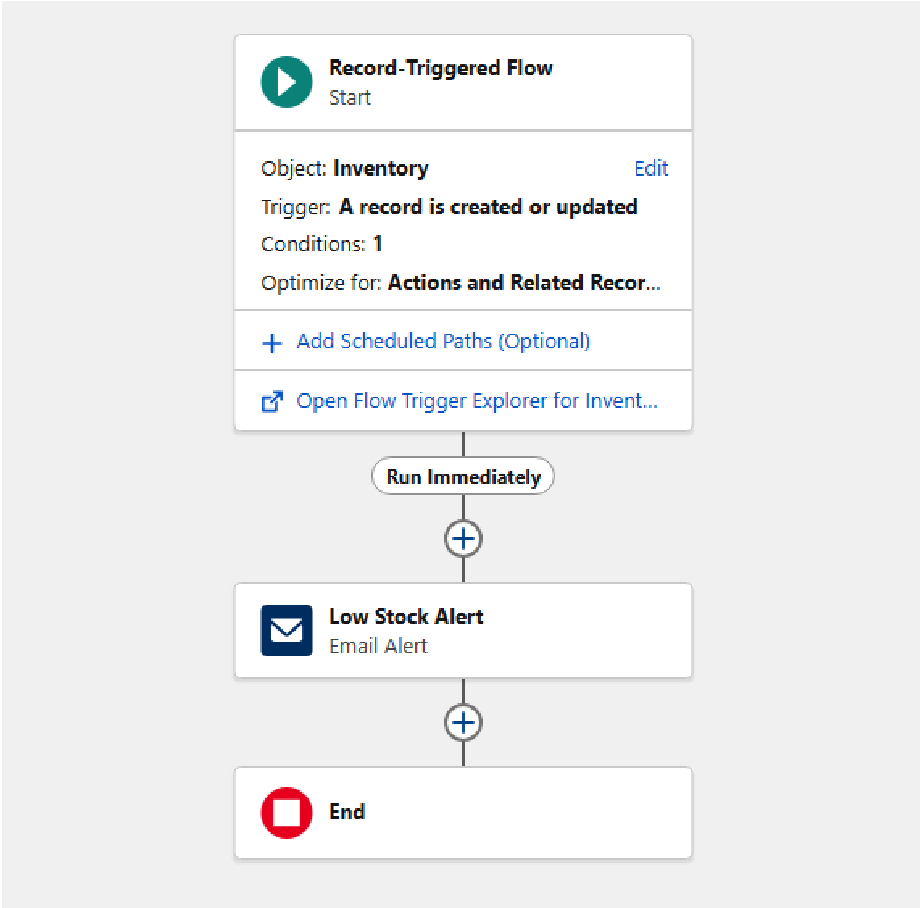


2) Stock Alert Flow

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

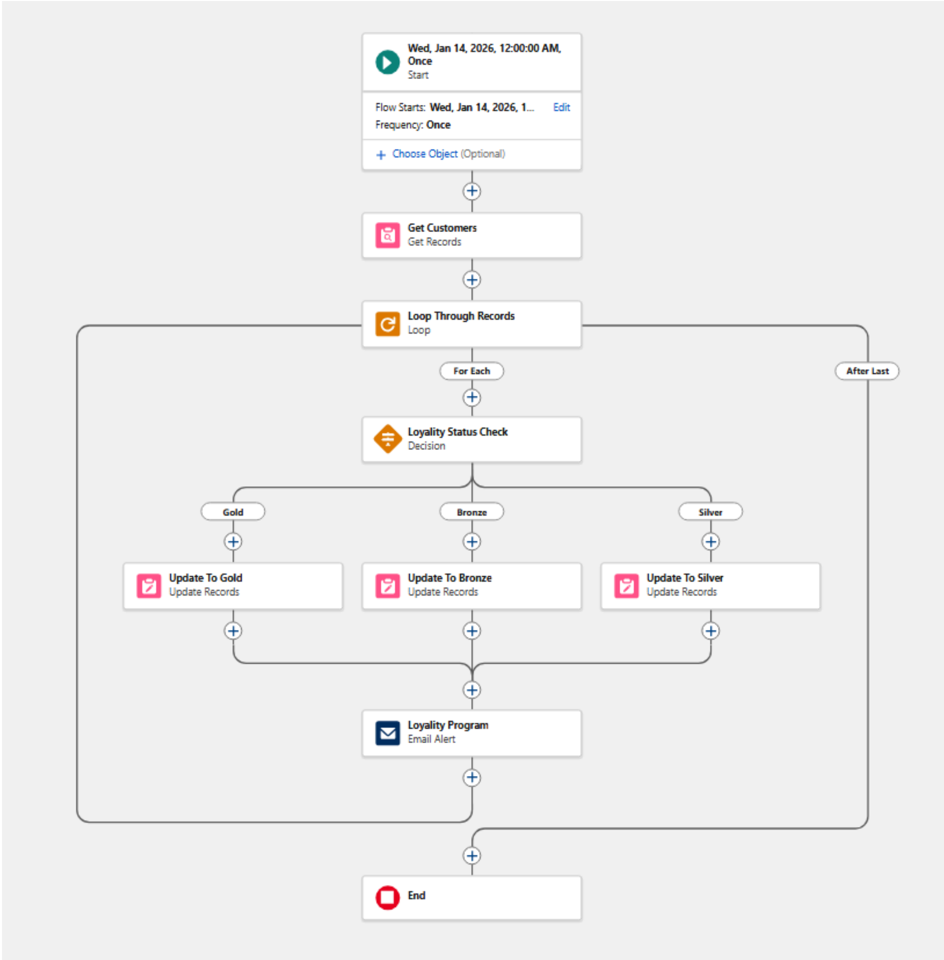


3) Loyalty Status Update Flow

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:**

  1. **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.

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

OrderTotalTrigger.apxt

Code Coverage: None ▾ API Version: 65

```
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;
            }
        }
    }
}
```

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.

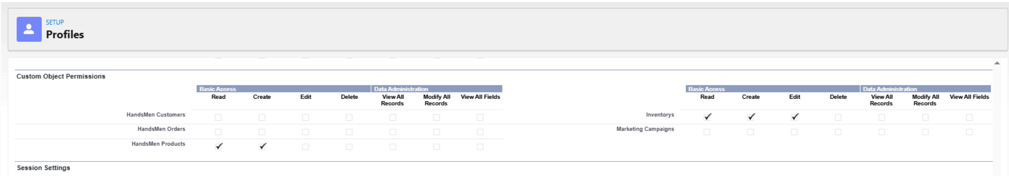
```
trigger StockDeductionTrigger on HandsMen_Order__c (after insert, after update) {
    Set<Id> productIds = new Set<Id>();

    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;

    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;
    }
}
```

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

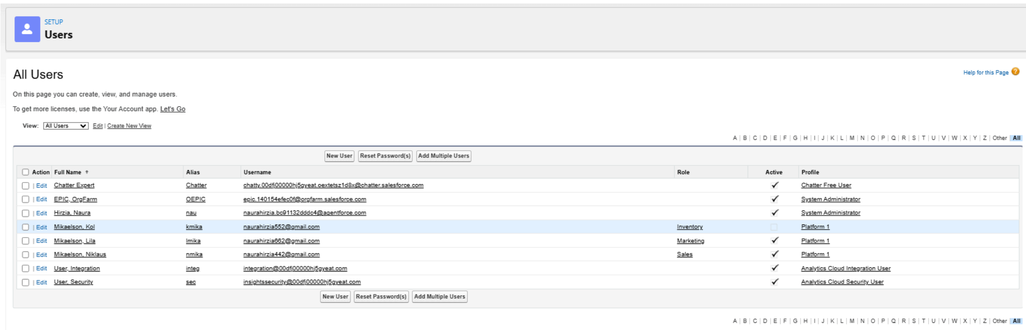


vi. User Accounts

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.



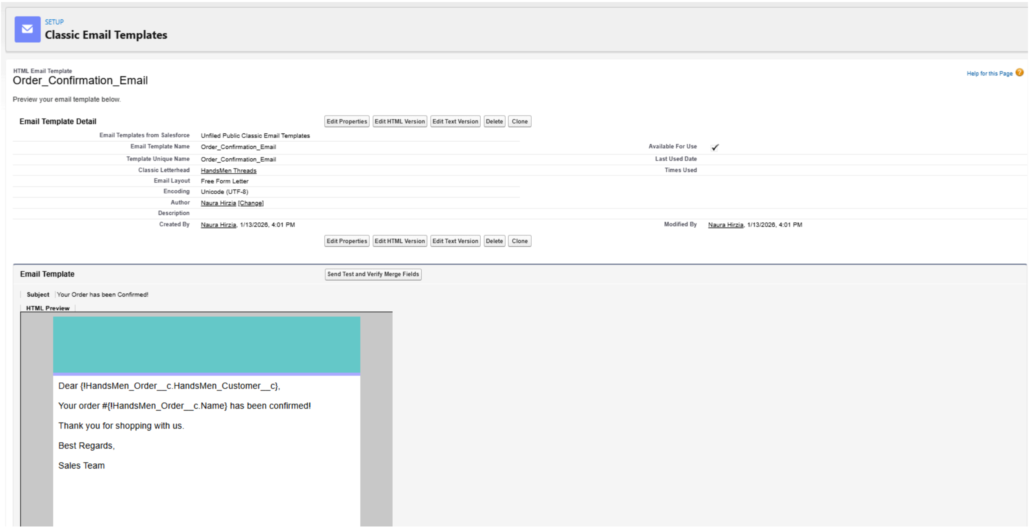
vii. Email Templates & Alerts

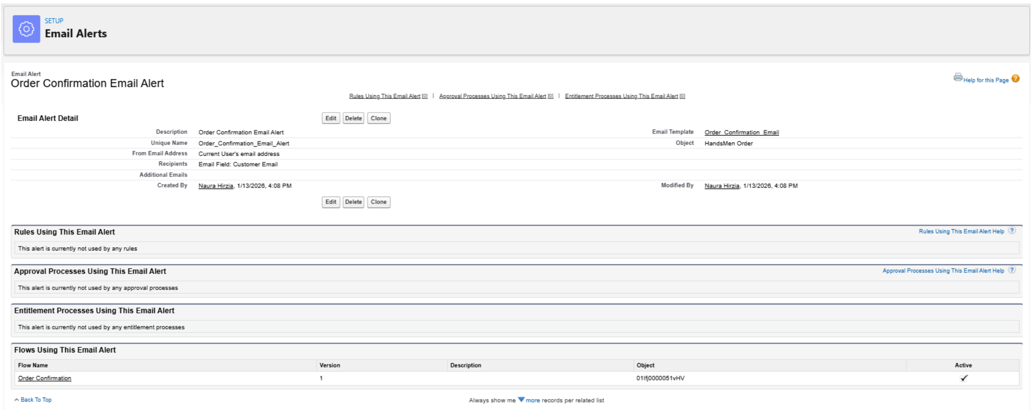
- **Order Confirmation Email:** Sent automatically when an order status changes to Confirmed.

- **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.



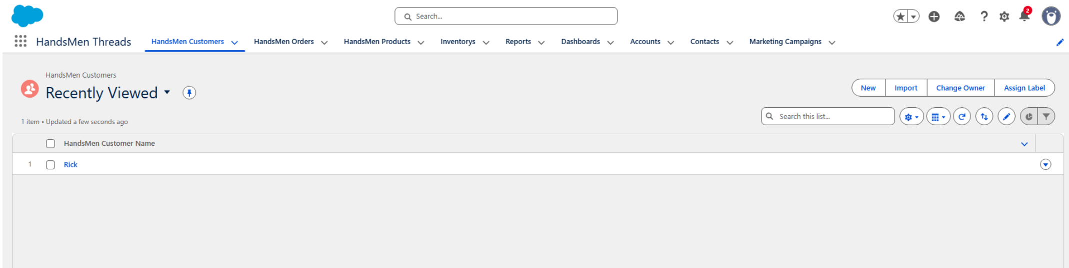


c. UI/UX Development & Customization

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)

Rika 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 x 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, Rika 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

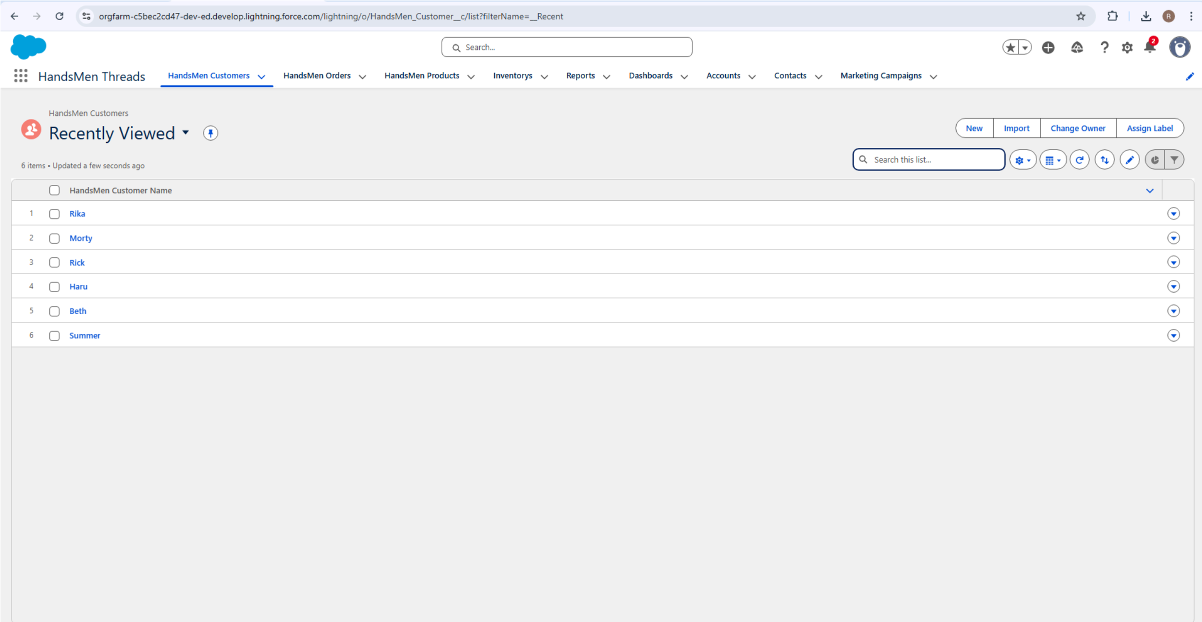


Fig: Custom App for HandsMen Threads

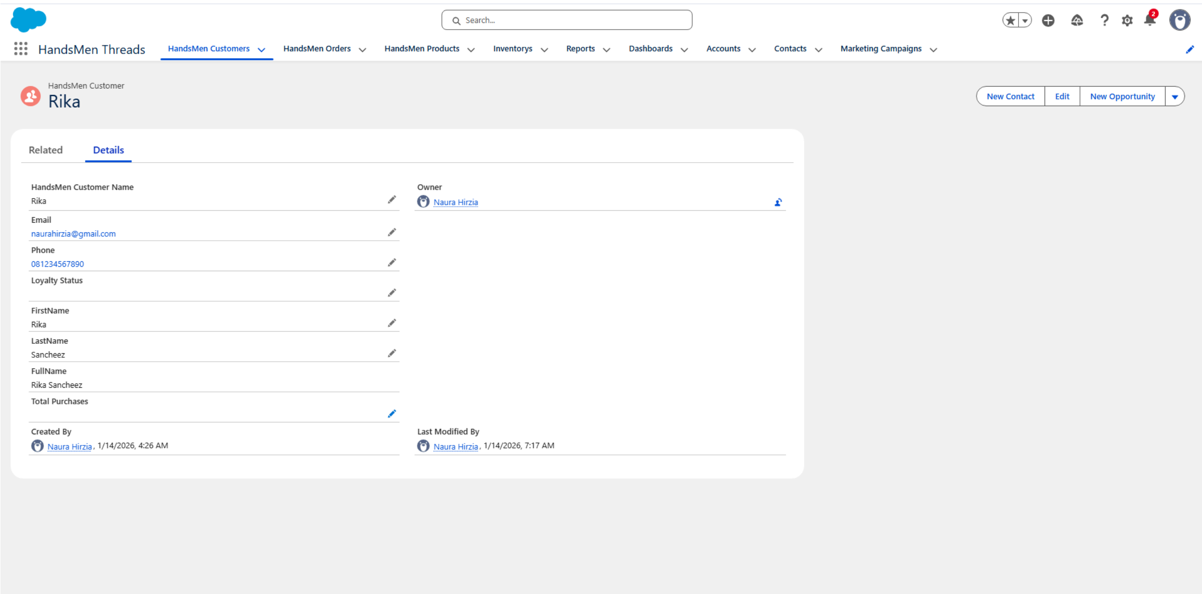


Fig: Customer Creation in HandsMen Threads

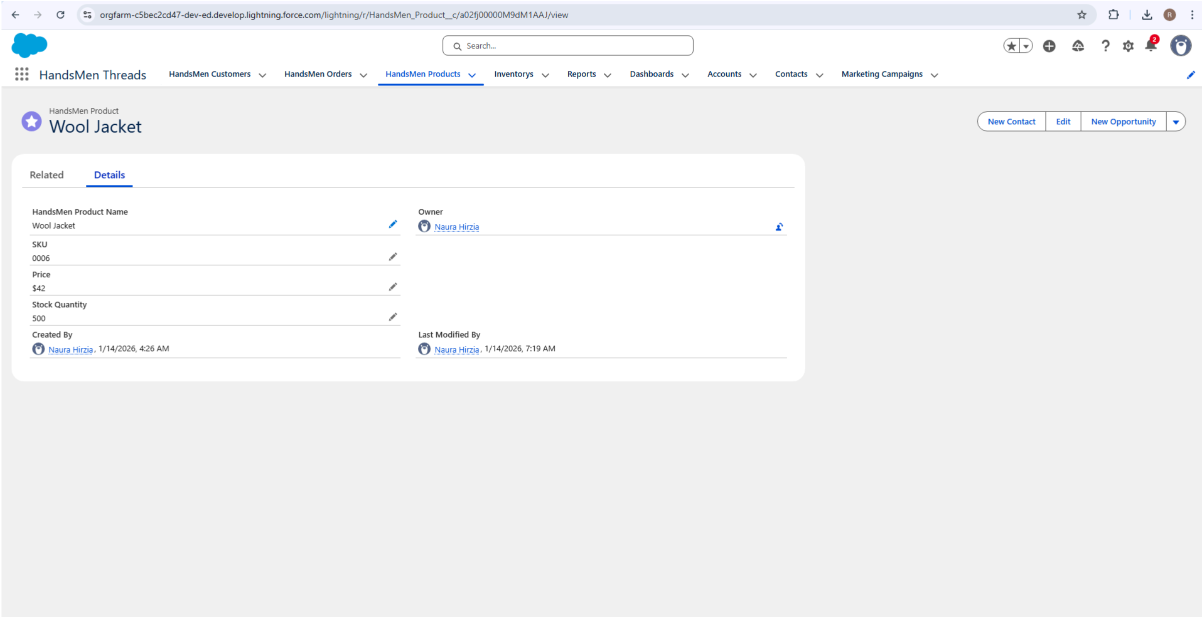


Fig: Products in HandsMen Threads

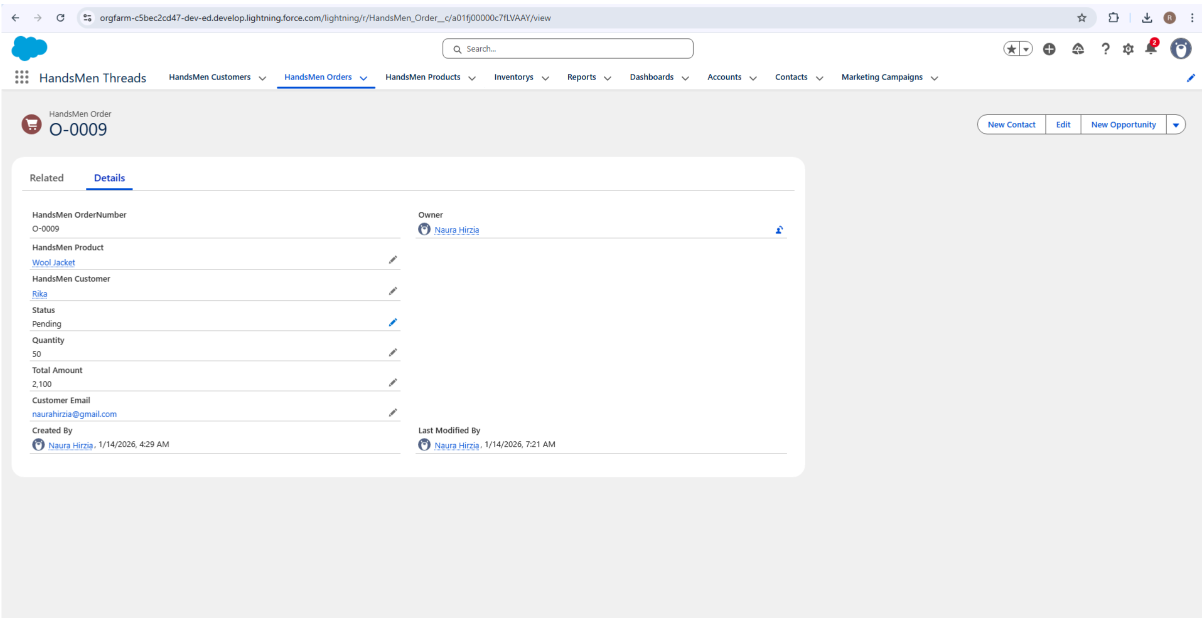


Fig: Order Confirmation

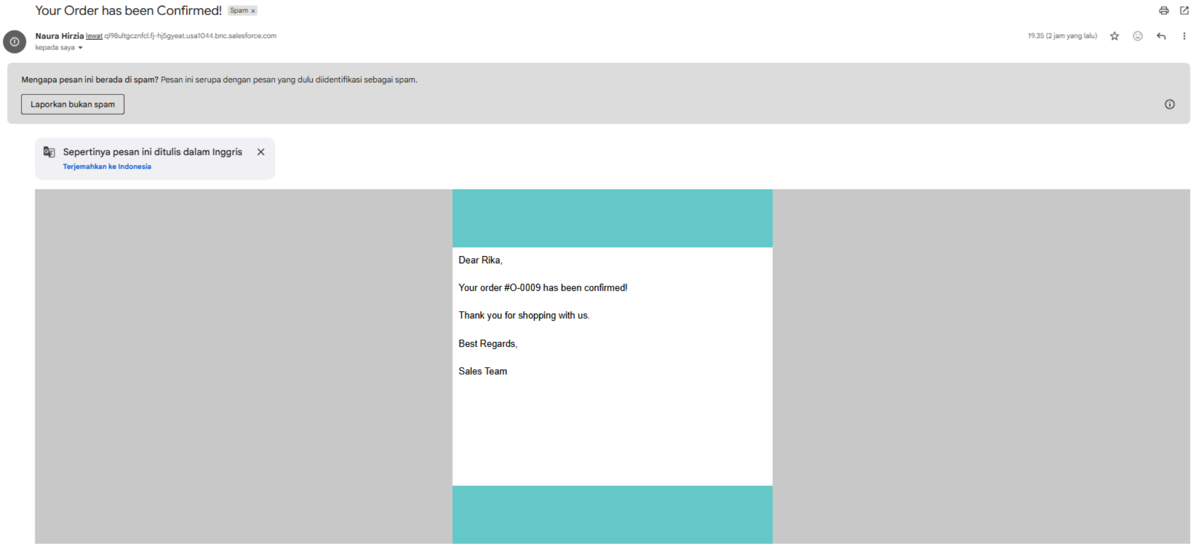


Fig: Order Confirmation Email

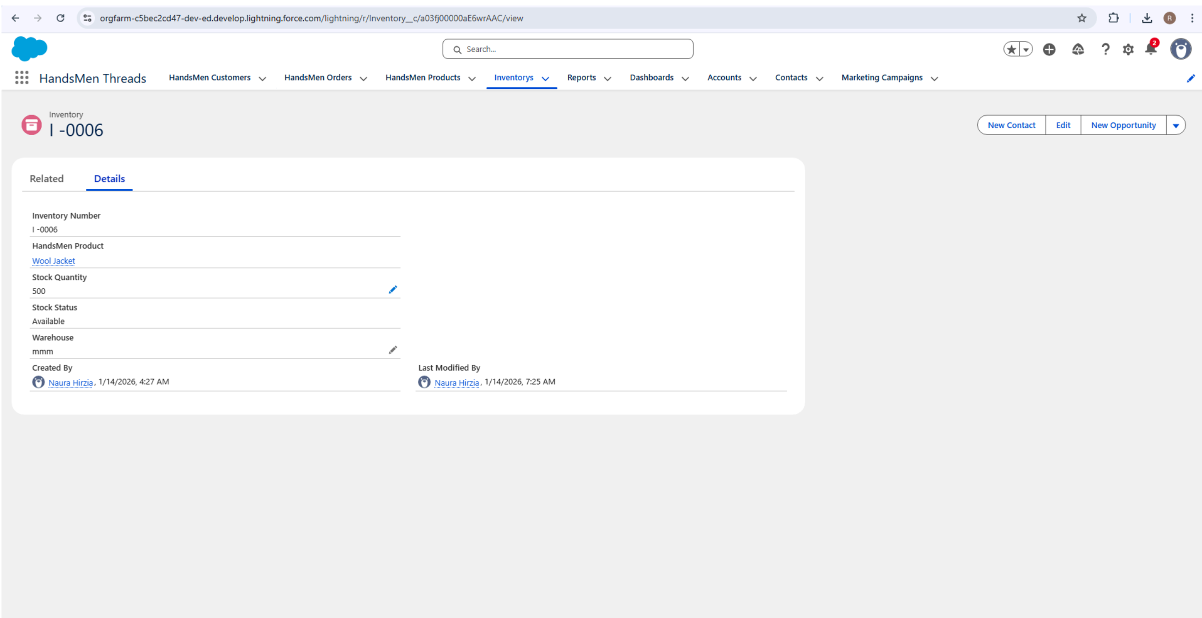


Fig: Inventory Creation

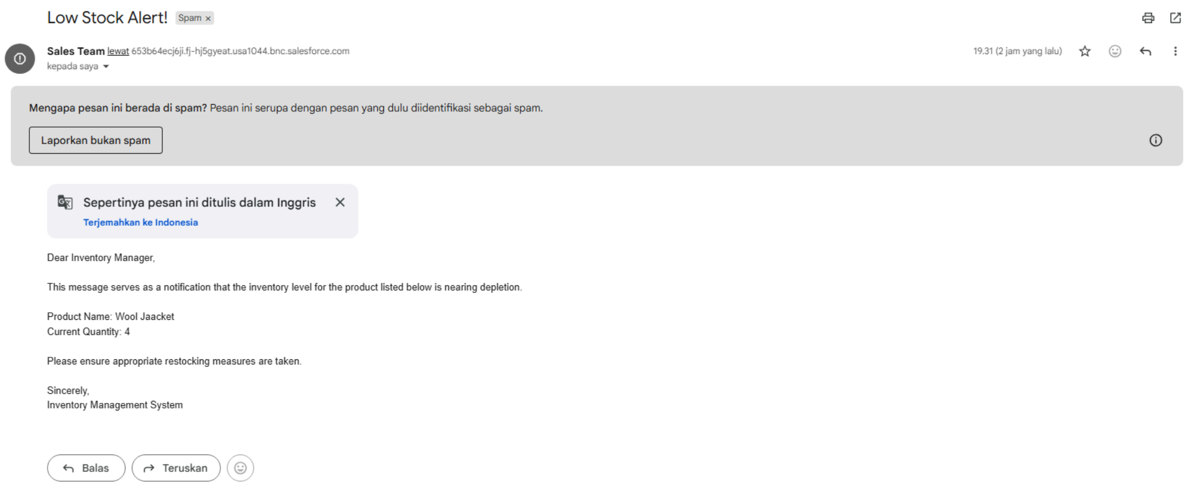


Fig: Low Stock Alert Email

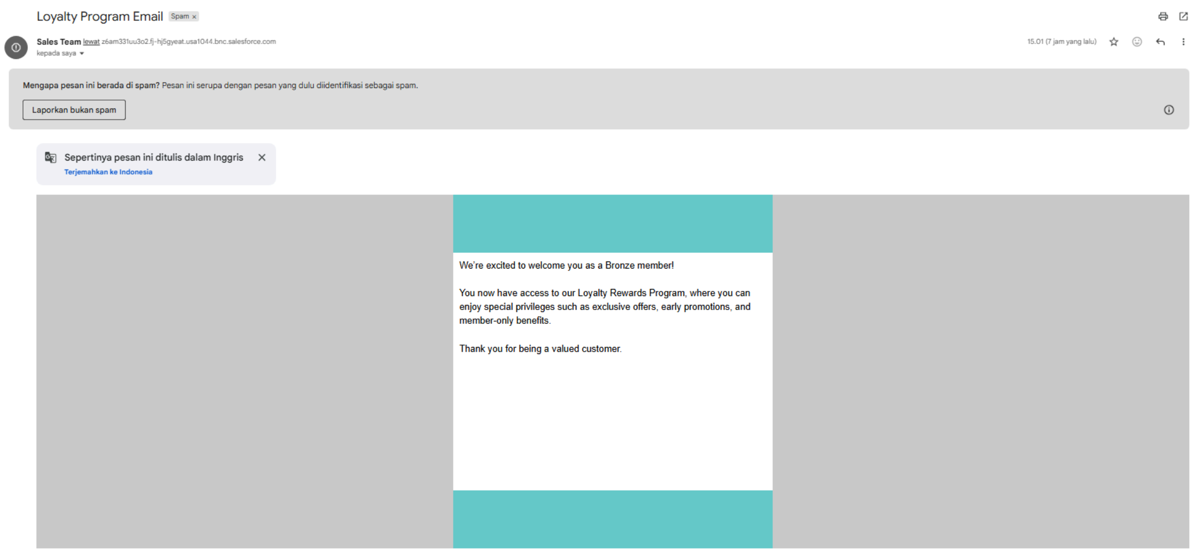


Fig: Loyalty Program Email

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.