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

Abstract

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

The project focuses on developing a comprehensive data model to centralize and streamline business information while enforcing data integrity directly from the user interface. The solution involves designing five key custom objects: Customer, Order, Product, Inventory, and Marketing Campaign. Business processes were automated using Record-Triggered Flows, Scheduled Flows, Email Alerts, and Apex to handle order conformation, loyalty status update, and low stock alerts.

To ensure clean and reliable data, validation rules were established, and role-based security model was implemented for the Sales, Inventory, and marketing teams.

This end-to-end CRM implementation improves customer experience through personalized communication, ensures operational efficiency with automation, and lays a scalable foundation for future growth using Salesforce Platform.

Objective

The objective of this Salesforce implementation project for HandsMen Threads is to design and deploy a scalable, centralized Customer Relationship Management (CRM) system that enhances data accuracy, streamlines business operations, and improves customer engagement. This includes:

- **Building a robust and efficient data model** using five key custom objects: Customer, Order, Product, Inventory, and Marketing Campaign.
- **Enforcing data integrity** through validation rules and user interface controls.

- **Automating core business processes** such as order confirmation, loyalty status updates, and low stock alerts using Record-Triggered Flows, Scheduled Flows, Email Alerts, and Apex.
- **Implementing a secure, role-based access model** tailored for Sales, Inventory, and Marketing teams.
- **Enhancing customer experience** through personalized communication and ensuring seamless operational workflows that support future business expansion on the Salesforce platform.

Technology Used

Salesforce: -

Salesforce is a cloud-based software company that provides customer relationship management (CRM) services. It helps businesses connect with their customers and improve various aspects of their operations, such as sales, service, marketing, and more. Salesforce offers a suite of products and services, often referred to as Customer 360, designed to unify teams and provide a single view of customer information.

Custom Objects: -

Custom objects in Salesforce are user-defined database tables that extend the standard Salesforce data model, allowing organizations to store and manage information specific to their unique business needs.

Tabs: -

Tabs are used to display object data in Salesforce UI.

Custom App: -

An app is a collection of tabs grouped together for a specific business purpose.

Profiles: -

A profile defines a user's access and permissions within the system.

Validation Rules: -

Validation rules are a powerful tool for maintaining data quality by ensuring that the data entered into Salesforce meets specific criteria before a record can be saved.

Roles: -

Roles primarily control data visibility and access at the record level.

Permission Sets: -

Permission sets are collections of settings and permissions that grant users access to specific features and data.

Email Templates: -

Predefined formats for sending emails to customers or users.

Email Alerts: -

Email alerts are actions in Flows or Workflow rules that sends emails using predefined templates.

Flows: -

Flows are a powerful, point-and-click tool for automating business processes within Salesforce.

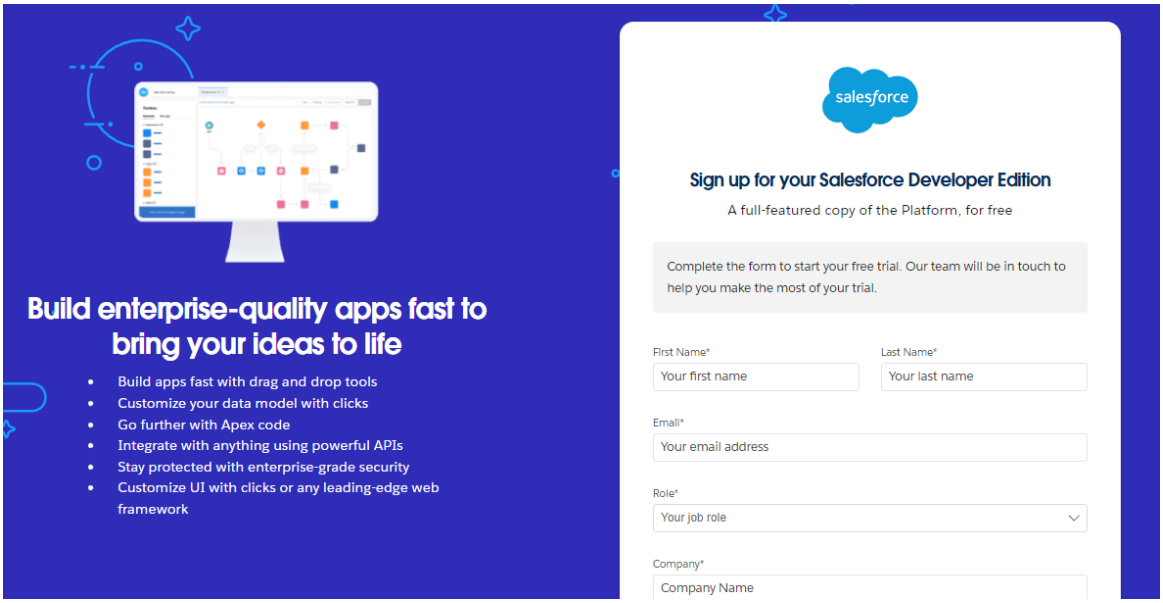
Apex: -

Apex is a proprietary, strongly-typed, object-oriented programming language used to extend the functionality of the Salesforce platform.

Detailed Execution of Project

1. Developer Org setup

- A developer org in Salesforce is created by https://developer.salesforce.com/signup link.



- Fill the details.
- Verify the account using the link given in the mail send by Salesforce.

2. Custom Object Creation

Five custom object was created to store business data:

- **HandsMen Customer** – Stores customer details like email, phone, loyalty status.
- **HandsMen Order –** Stores order details placed by customer.
- **HandsMen Product –** Stores product details like SKU, price, and stock.
- **Inventory –** Tracks stock quantity and warehouse location.
- **Marketing Campaign –** Stores promotional campaign and scheduling.

3. Custom Tab Creation

- Created tabs for the five custom objects

4. Creating the Lightning App

- A custom lightning app HandsMen Threads was created.
- Assigned to System Administrator profile.
- Included tabs: HandsMen Customer, HandsMen Order, Inventory, HandsMen Product, Reports, Dashboard, Account, Contact, Marketing Campaign.

5. Creating Custom Fields for Custom Objects

- The following custom fields are created for **HandsMen Customer** object:

Fields Name/Label	Data Type
Email	Email
Phone	Phone
Loyalty Status	Picklist: Bronze, Gold, Silver
Total Purchases	Number
FirstName	Text
LastName	Text

- The following custom fields are created for **HandsMen Order** object:

Fields Name/Label	Data Type
Status	Picklist: Pending, Confirmed, Rejection
Quantity	Number
Total Amount	Number

- The following custom fields are created for **HandsMen Product** object:

Fields Name/Label	Data Type
SKU	Text
Price	Currency
Stock Quantity	Number

- The following custom fields are created for **Inventory** object:

Fields Name/Label	Data Type
Warehouse	Text
Stock Quantity	Number

- The following custom fields are created for **Inventory** object:

Fields Name/Label	Data Type
Start Date	Date
End Date	Date

- The following **lookup relationships** are created:
  - Lookup Relationship between HandsMen Product and HandsMen Order
  - Lookup Relationship between HandsMen Order and HandsMen Customer
  - Lookup Relationship between Marketing Campaign and HandsMen Customer
  - Master-Detail Relationship between Inventory and HandsMen Product

- The following formula fields are created:

Object Name	Field Name/Label	Return Type	Formula
HandsMen Customer	FullName	text	FirstName__c + " " + LastName__c
Inventory	Stock Status	text	IF(Stock_Quantity__c > 10, "Available", "Low Stock")

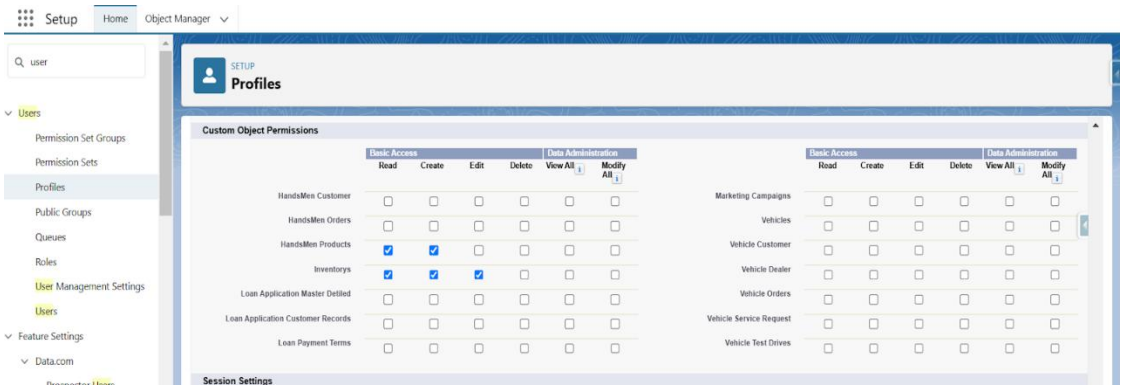
6. Creating Validation Rules

To ensure accurate data entry and to enforce business logic following validation rules were created:

Object Name	Rule Name	Error Condition Formula	Error Message	Description
HandsMen Order	Total Amount	Total_Amount__c <= 0	Please Enter Correct Amount	Prevents invalid amount insertion
Inventory	Stock Quantity	Stock_Quantity__c < = 0	the inventory count is never less than zero	Prevents invalid stock insertion
HandsMen Customer	Email	NOT CONTAINS(Email, "@gmail.com")	Please fill Correct Gmail	Prevents invalid email insertion

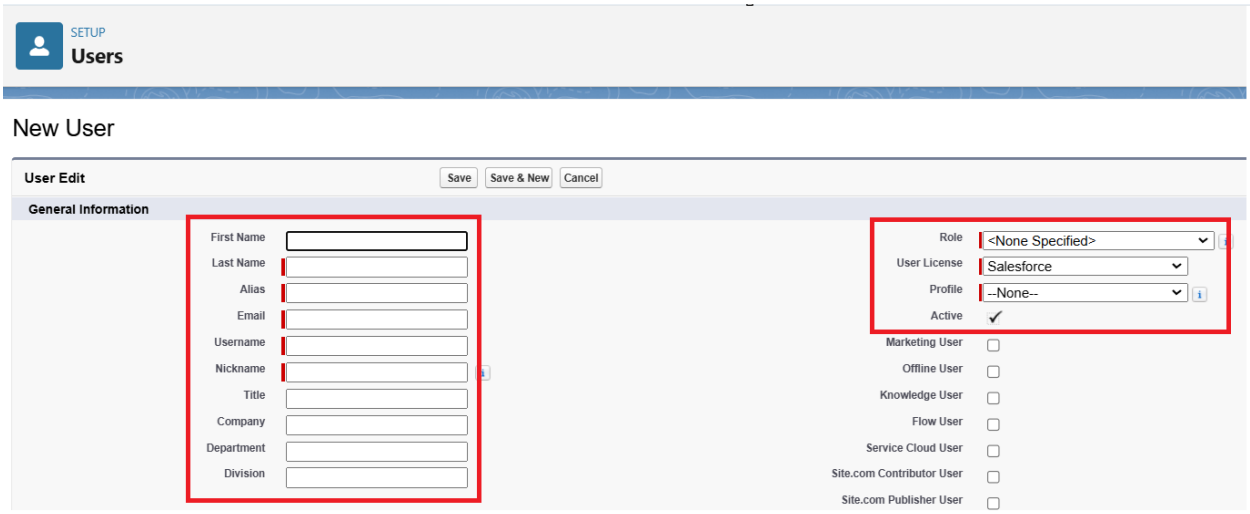
7. Creating Profiles, Roles, and Users

- To securely manage data profile named with “Profile 1” was created and then granted access permissions for HandsMen products and Inventory objects.



- Now three roles under CEO were created. They are: Sales, Inventory, and Marketing.
- Finally three users for respective roles were created. They are as follows:

User Name	Role Assigned
Niklaus Mikaelson	Sales
Kol Mikaelson	Inventory
Dany Mikaelson	Marketing



8. Creating Permission Sets

To enhance data security following permission sets were created:

Permission Set Name	Role	Access Level	User Assigned
Sales Permission Set	Sales	Full Access to Customers, Orders	Niklaus Mikaelson
Inventory Permission Set	Inventory	Read & Edit on Inventory, Products	Kol Mikaelson
Marketing Permission Set	Marketing	Read on Customers, Edit on Marketing Campaigns	Dany Mikaelson

9. Email Template and Email Alert

Created following email templates:

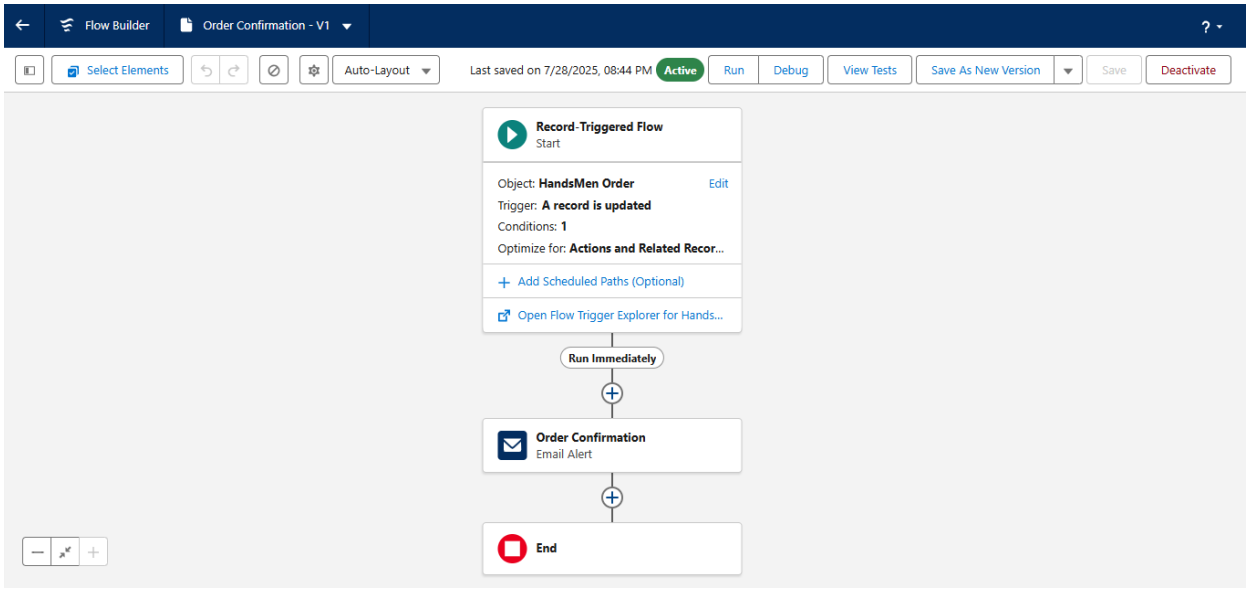
Name	Type	Trigger
Order Confirmation	HTML	Sent when Order is placed
Low Stock Alert	Text	Sent when Inventory__c.Stock_Quantity__c < 5
Loyalty Program Email	HTML	Sent when customer qualifies for loyalty rewards

Corresponding email alerts were also created.

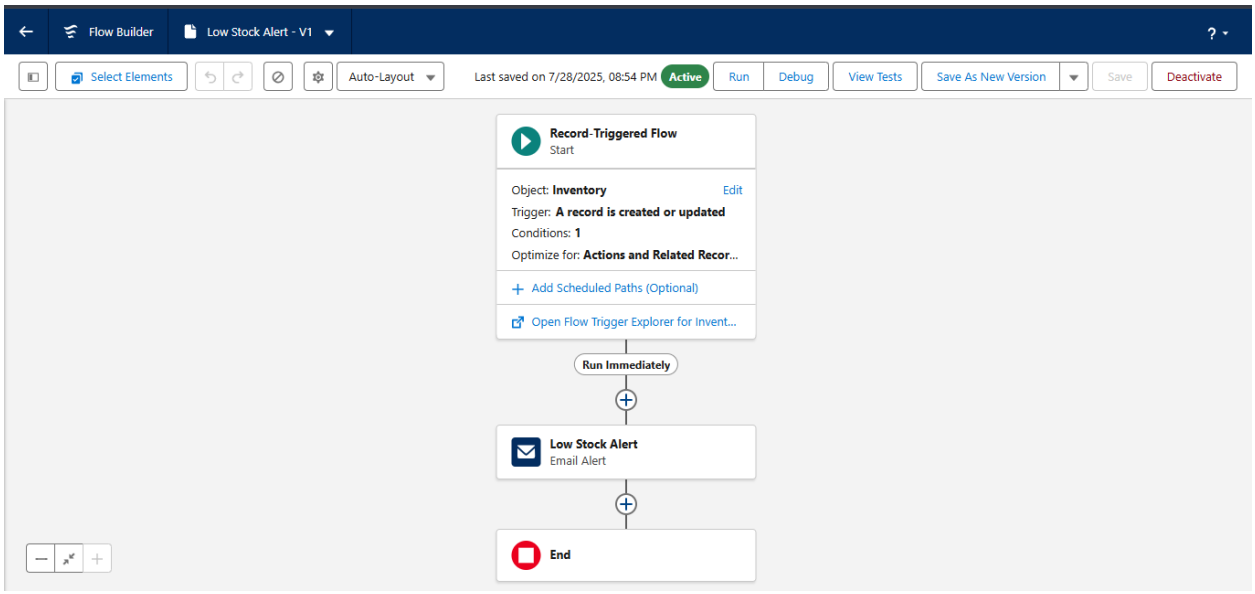
10. Implementation of Flows

To automate business logic the following flows are created:

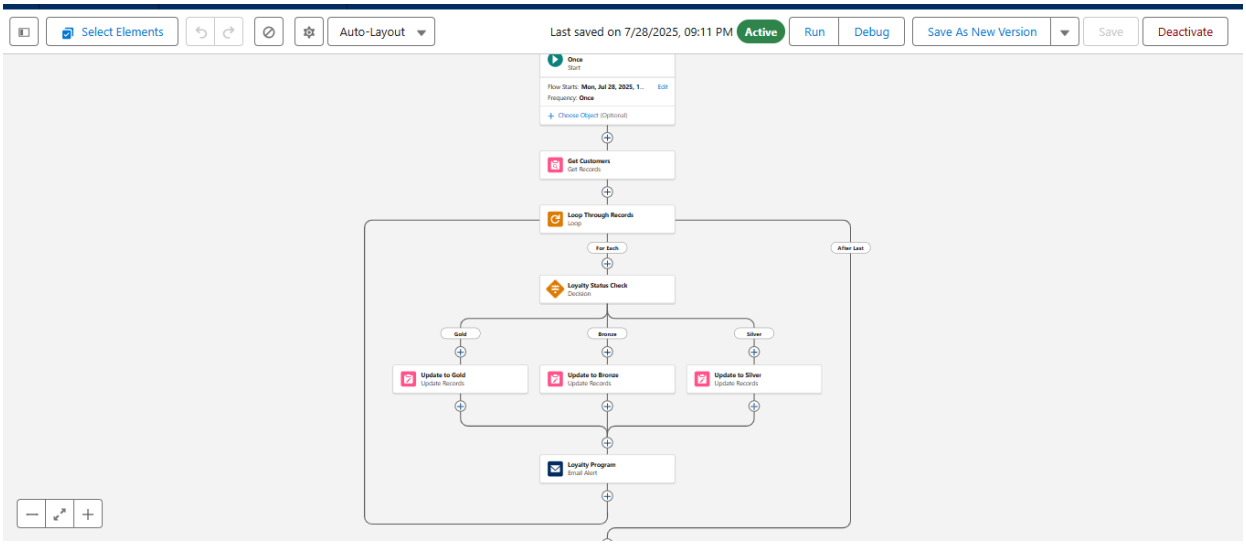
- **Order Conformation**: triggered when order status = ‘confirmed’.



- **Low Stock Alert**: triggered when stock quantity<5.



- **Loyalty Program**: Daily updates loyalty status.



11. Create apex triggers and classes

To enhance the automation, we create the following apex triggers:

- **StockDeductionTrigger** on HandsMen_Order__c, when an order confirms stock quantity for the order reduces form the Inventory.

Code:

```
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, Product__c
         FROM Inventory__c
         WHERE 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.Product__c == order.HandsMen_Product__c) {
                    inv.Stock_Quantity__c -= order.Quantity__c;
                    inventoriesToUpdate.add(inv);
                    break;
                }
            }
        }
    }

    if (!inventoriesToUpdate.isEmpty()) {
```

```
        update inventoriesToUpdate;
    }
}
```

- **OrderTotalTrigger** on HandsMen_Order__c , updates the Total Amount automatically by the formula Total Amount=order_quantity*product_price

Code:

```
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;
            }
        }
    }
}
```

To enhance the automation, we create the following apex class:

- **InventoryBatchJob,** Helps prevent stockouts, ensuring operational continuity.

Code:

```
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);

}

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);
}
}
```

Project Explanation with a Real-World Example

1.**Customer Registration**

- A customer, John Smith, visits the store.
- In salesforce: A record is created in the HandsMen Customer object with his name, phone, email, etc.
- Validation Rule: Ensures email is valid.

2.Product Setup

- The admin adds products like T-Shirts, Jeans, etc. into Product object
- Each Product has a price and other details.
- Inventory is also created to manage stock for those products.

3.Order Place

- John decides to purchase 2 jeans(costs 150 each). An order is placed.
- In Salesforce: A new order record is created.
- Apex Trigger: Automatically calculates Total Amount=150*2=300

4.**Inventory Update:**

As soon as order is confirmed

- Apex Trigger on Inventory reduces stock by 2.
- Validation rule: ensures stock never goes to 0

5.Loyalty Program:

- John has total of Rs.300 purchase.
- John becomes Bronze member

6.Email Notification:

- As soon as order is confirmed John receives an email notification.
- As soon as John becomes Bronze member he receives an email notification.
- Flow + Email Alert is Triggered

Conclusion

The Salesforce implementation for HandsMen Threads marks a significant step forward in streamlining operations and enhancing customer engagement within the organization. By developing a robust and scalable data model with custom objects like Customer, Order, Product, Inventory, and Marketing Campaign, the project ensures centralized data management and seamless information flow. The integration of automation through Flows, Apex, and Email Alerts has optimized critical business processes such as order confirmations, loyalty status updates, and low-stock alerts, thereby improving efficiency and responsiveness.

With enforced data integrity via validation rules and secure, role-based access for Sales, Inventory, and Marketing teams, the system supports clean, reliable data and protects sensitive business information. Furthermore, the inclusion of batch and scheduled processes, like the InventoryBatchJob, lays the groundwork for proactive inventory management.

Overall, this end-to-end Salesforce CRM solution not only strengthens HandsMen Threads' operational capabilities but also creates a personalized and engaging experience for customers, positioning the company for scalable growth and long-term success in the competitive fashion industry.