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

ABSTRACT

The HandsMen Threads Salesforce project is real-world project where we applied to a real implementation to modernize and automate an operation of a high-end men's clothing brand. The use of this project leads to improved data management, updated customer experiences, and greater efficiencies of internal workflows, through the use of Salesforce's CRM. The implemented solution covered many functions, like managing its inventory, automated communications, enabling loyalty programs, batch job processing, etc., bringing digital transformation to the company.

OBJECTIVE

The end goal of this project is to create a scalable, automated Salesforce solution that:

- Makes seamless data flow between business departments possible
- Delivers increased customer satisfaction through automation
- Ensures data integrity and security.
- Reduces manual work through backend logic (e.g. Apex, Flows)
- Enables real-time decision-making with updated records and insights.

TECHNOLOGY DESCRIPTION

Below is a detailed breakdown of the key technologies used:

1) Salesforce Lightning App Builder
    - Used to design and customize pages for multiple objects (Customers, Orders, Inventory).
    - Provided functionality to create dynamic Lightning record pages for end-users, creating a clean, responsive, and user-friendly interface.
    - Provided the ability to embed components such as charts, list views, and quick actions for enhanced usability.

2) Custom Objects & Fields
    - Developed custom objects.
    - Created fields using correct data types, picklists, and relationships (Lookup/Master-Detail) to ensure organized data.
    - These custom objects are the backbone of the application core data model.

3) Validation Rules & Page Layouts
    - Utilized Validation Rules to apply business rules (e.g., not allowing an order to be submitted if the product stock is 0).
    - Ensured the data was accurate and correct for users at the exact moment of the record being created or changed.
    - Developed custom Page Layouts for each custom object to show only the relevant fields for the end-user, enhancing the user experience and removing clutter.

4) Flows
    - Developed Record-Triggered Flows to automate the following processes: -
        - Sending confirmation emails automatically as soon as a new order is created in Salesforce.
        - Automatically assigning a customer a particular loyalty tier if their total purchase value has increased.
    - Created Scheduled Flows that run on a specific date/time (e.g. run some inventory update processes in Salesforce at midnight every night).
    - Flows were the foundation of declarative automation and allowed the application to be responsive and intelligent.

5) Apex Triggers & Batch Apex
    - Used Apex Triggers for custom logic that could not be handled through Flows such as:
        - Automatically, calculate loyalty points.
        - Update related records with a new order.
    - Using these programmatic tools helps to keep performance and scalability high.

6) Profiles, Roles, Permission Sets
    - Set up Profiles to define baseline level of access for the different user-types (e.g., Sales Reps, Warehouse Staff).
    - Built out Roles to maintain the organizational hierarchy and allow record-level sharing.
    - Implemented Permission Sets to provide flexible access to features without changing profiles. Valuable for special permissions for sending bulk updates or reports.

7) Email Templates

- Designed branded and dynamic Email Templates in Lightning Email Builder.
- Integrated Email Templates with Flows and Triggers to send:
    - Order confirmations
    - Stock alerts to warehouse teams
    - Updates to customers in the Loyalty program

PROJECT PHASES

1) Phase 1: Planning & Requirements Gathering
    - Identified key use-case scenarios: order confirmation, inventory alerts, loyalty program, simplified batch processes.
    - Developed object model by analyzing business requirements.

2) Phase 2: Data Modeling
    - Created custom objects: Customers, Products, Orders, Loyalty Status.
    - Developed relationships among them (lookups & master-details).

3) Phase 3: UI & App Configuratio
    - Created a Lightning App: "HandsMen Threads".
    - Created navigation tabs, list views, record pages, to simplify access.

4) Phase 4: Automation
    - Used flows for record triggered automation (e.g. order confirmation email).
    - Created batch jobs to process our tasks like nightly jobs that run every 24 hrs.

- Enabled Apex Trigger for process automation using real time (i.e loyalty tier calculation).

5) Phase 5: Security/Access
    - Configured Profiles, Roles and Permission Sets.
    - Created department specific permissions to avoid accessing things they shouldn't.

6) Phase 6: Testing and Deploying
    - Tested all use case scenarios in a sandbox environment thoroughly.
    - Planned to deploy into production with minimal downtime.

PROJECT EXPLANATION WITH REAL WORLD EXAMPLE

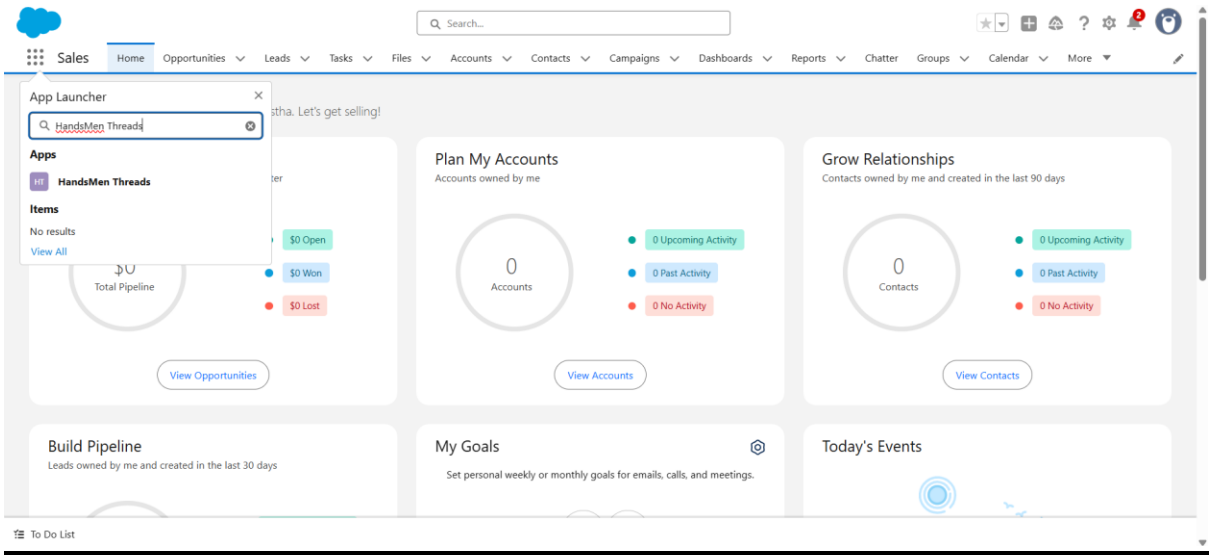
Let’s say a customer named Karan places an order on the HandsMen Threads website.

- Order Confirmation Email: Karan instantly receives an automated email confirming his purchase.
- Loyalty Update: The system checks Karan’s order history and updates his loyalty tier to "Gold" since this is his 5th order.
- Inventory Check: The item he ordered now has only 4 pieces left. The system automatically sends a low-stock alert to the warehouse team.
- Midnight Processing: At 12 AM, the system processes the day’s bulk orders, updates stock levels, and reflects all financial entries.

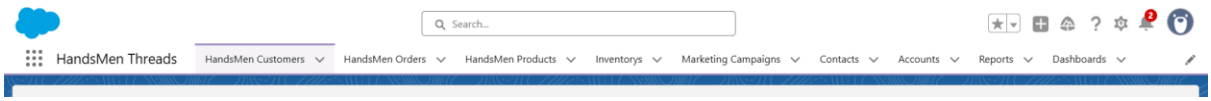
This shows how the system runs smoothly without manual intervention, providing a real digital transformation.

APPENDIX: PROJECT SCREENSHOTS & VISUALS

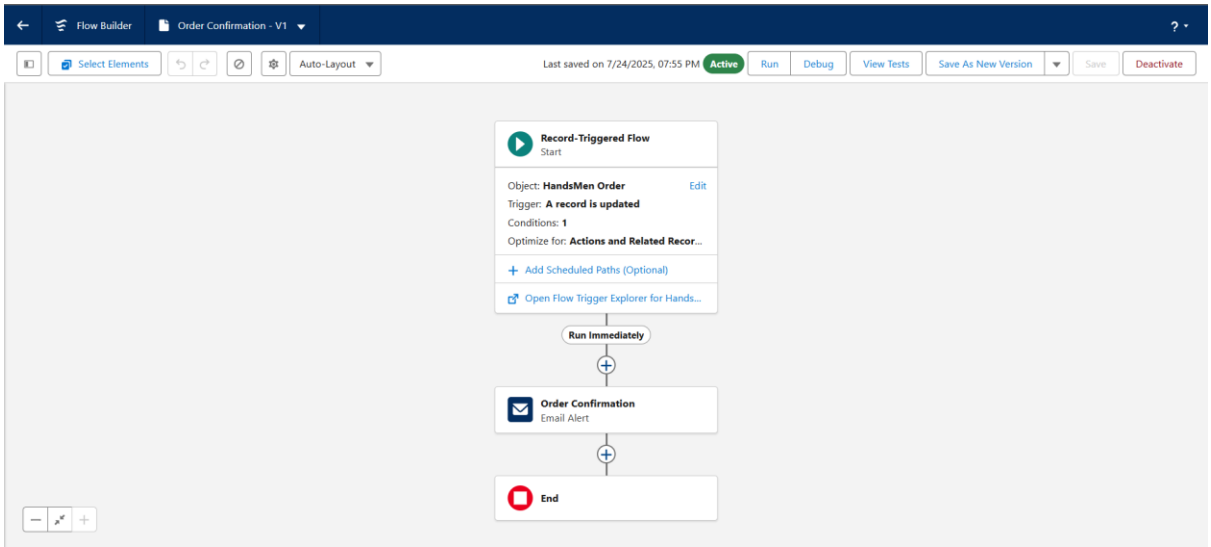
- Custom Application



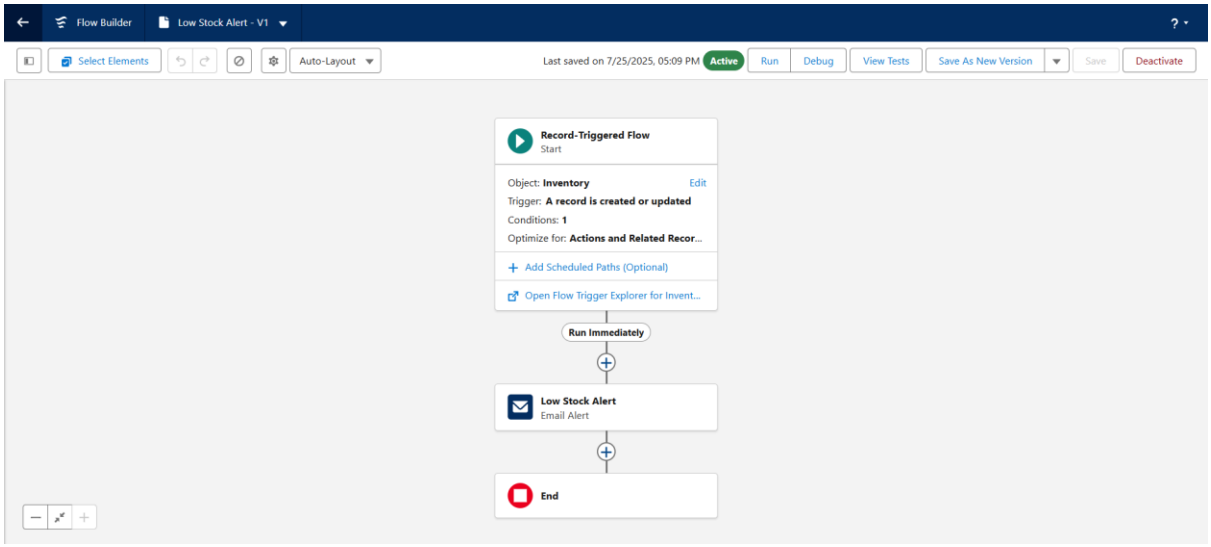
- Custom and Standard Objects



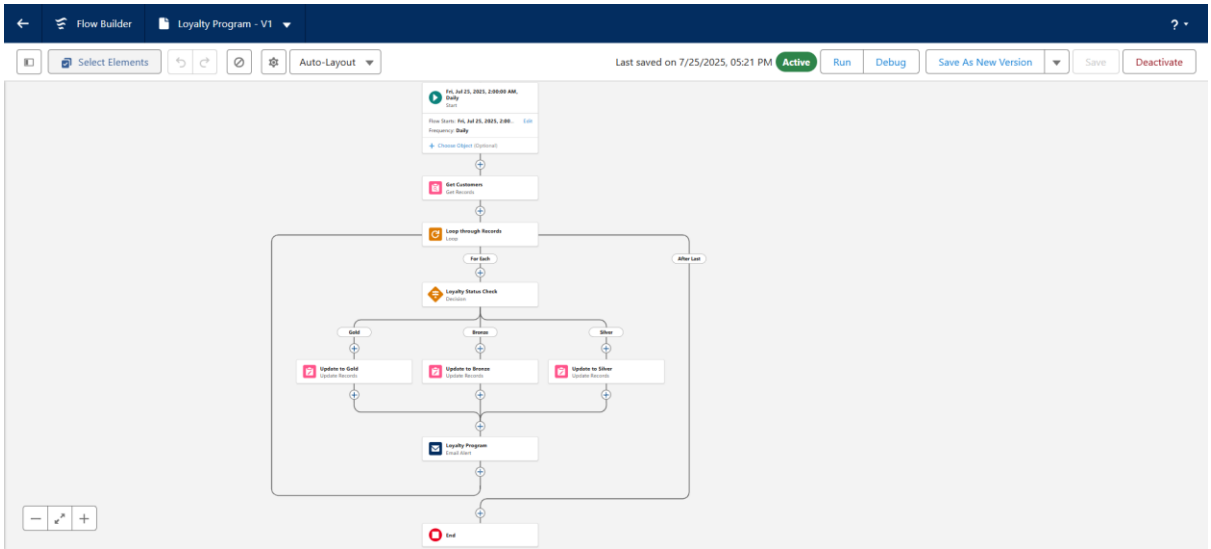
- Order Confirmation Flow



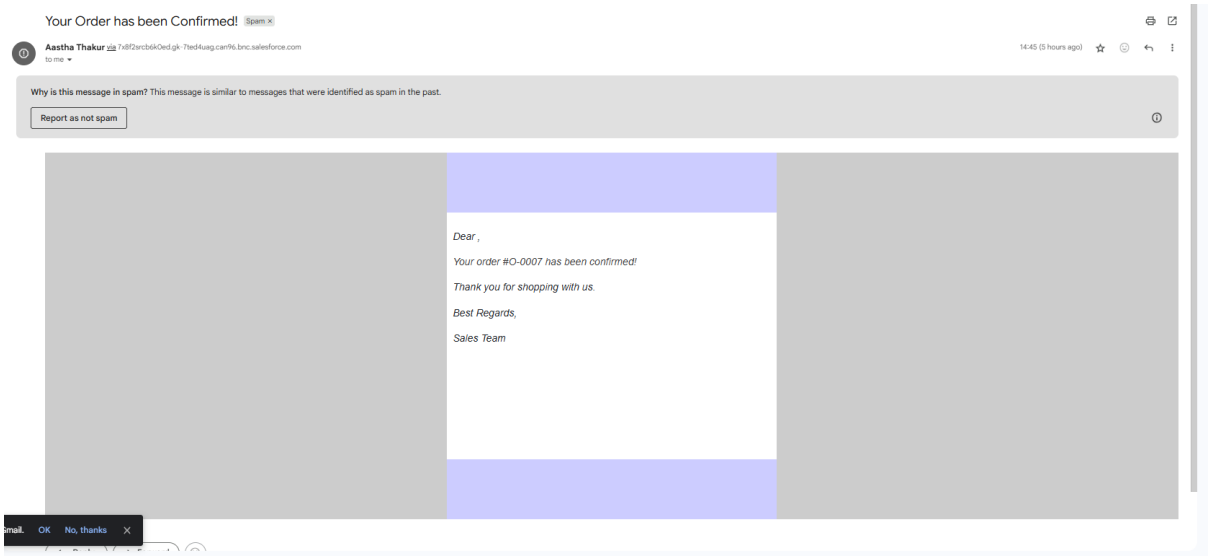
- Low Stock Alert Flow



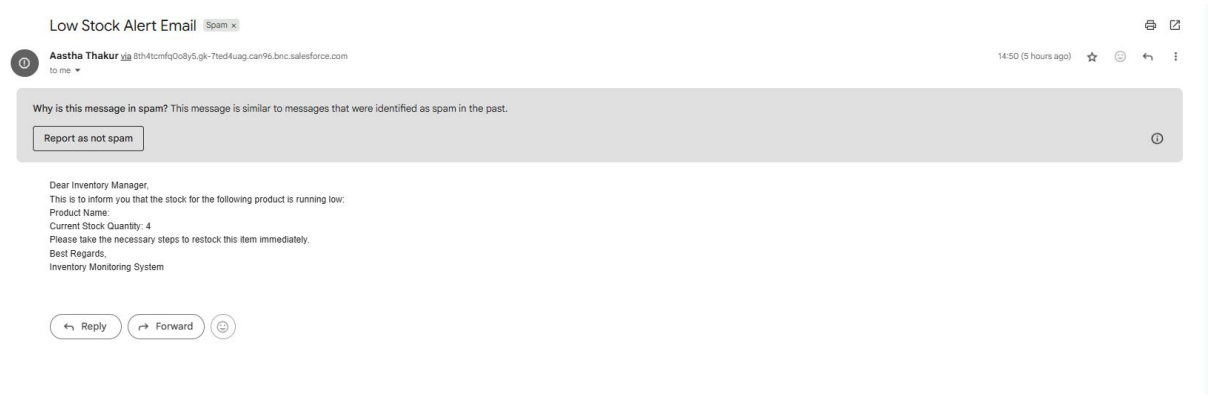
- Loyalty Program Flow



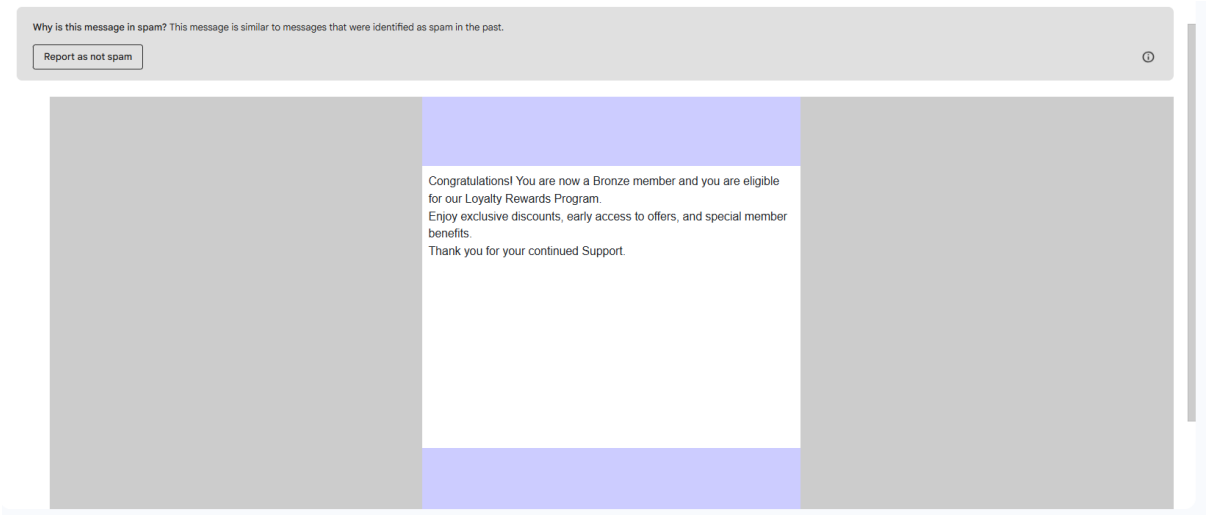
- Order confirmation Email



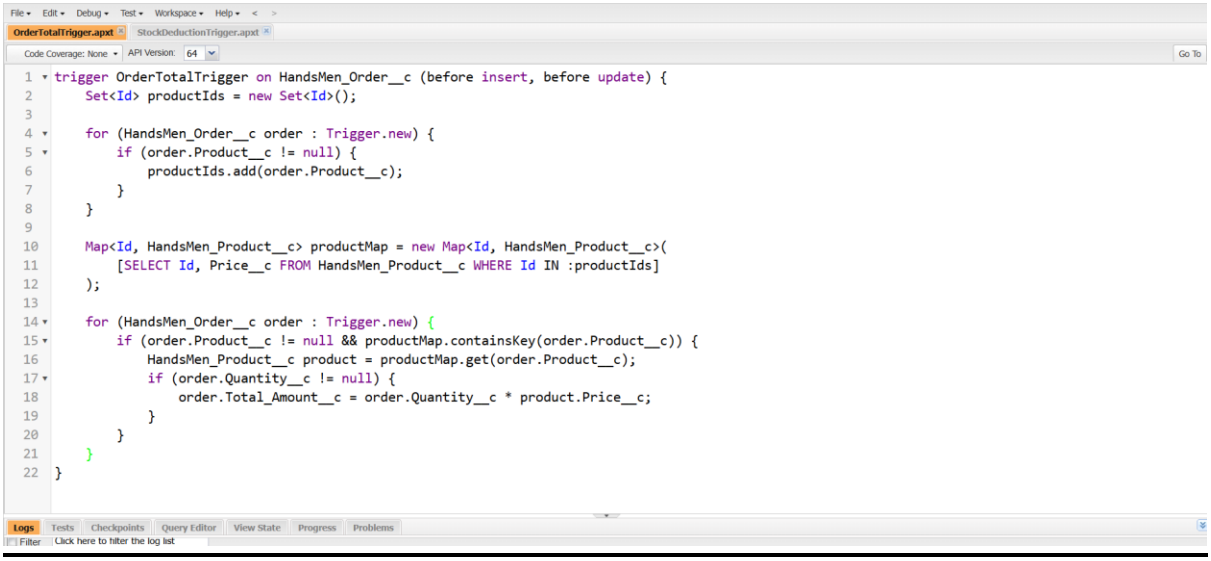
- Low Stock Alert Email



- Loyalty Status Update Email



- Apex Trigger: OrderTotalTrigger



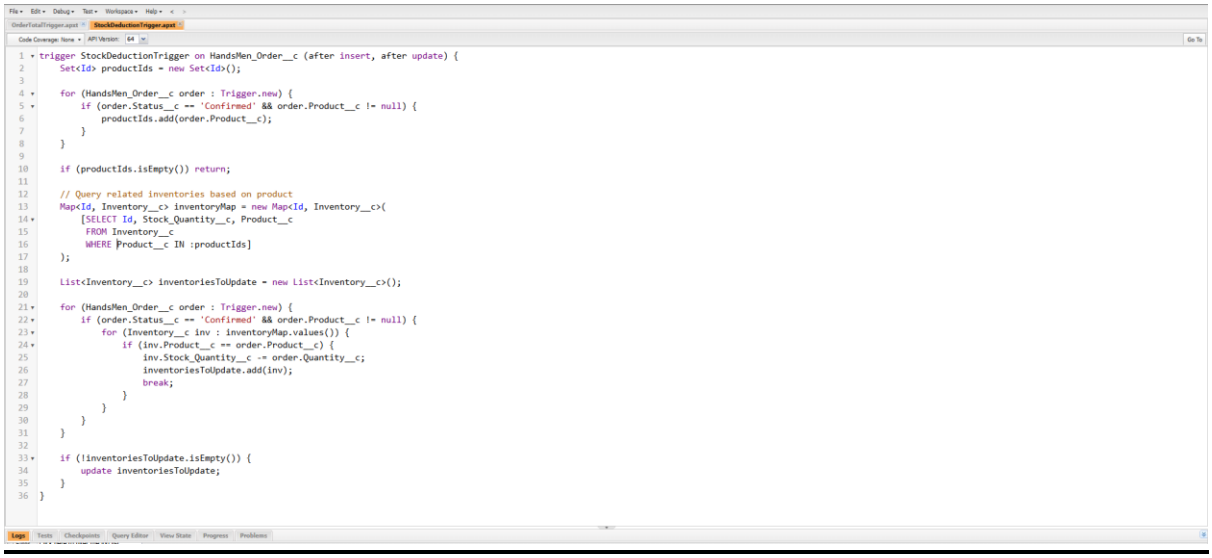
```
trigger OrderTotalTrigger on HandsMen_Order__c (before insert, before update) {
    Set<Id> productIds = new Set<Id>();

    for (HandsMen_Order__c order : Trigger.new) {
        if (order.Product__c != null) {
            productIds.add(order.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.Product__c != null && productMap.containsKey(order.Product__c)) {
            HandsMen_Product__c product = productMap.get(order.Product__c);
            if (order.Quantity__c != null) {
                order.Total_Amount__c = order.Quantity__c * product.Price__c;
            }
        }
    }
}
```

- Apex Trigger: 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.Product__c != null) {
            productIds.add(order.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.Product__c != null) {
            for (Inventory__c inv : inventoryMap.values()) {
                if (inv.Product__c == order.Product__c) {
                    inv.Stock_Quantity__c -= order.Quantity__c;
                    inventoriesToUpdate.add(inv);
                    break;
                }
            }
        }
    }

    if (!inventoriesToUpdate.isEmpty()) {
        update inventoriesToUpdate;
    }
}
```

CONCLUSION

The HandsMen Threads project using Salesforce shows just how powerful a well designed CRM system can be in changing how you operate as a business, in the context of fashion.

With Salesforce changing the way we thought about the powerful tools available, such as Flows, Apex, data modelling, automation and process, we've been able to streamline and improve business processes, create and enforce data standards and provide a better user experience.

This project has not only limited manual work and operational mistakes, it has also opened the business up to real-time and customer-focused automation. It is a definitive example of how digital transformation can turn technology into business objectives, without fear of any future hindrance.

As a developer, this process has given me a deeper knowledge of the Salesforce ecosystem, and taught me how to transform business reality into smart, scalable technological solutions.