Project Title: HandsMen Threads - Salesforce CRM Implementation

# 1. Abstract

HandsMen Threads, a dynamic men’s fashion brand, is embarking on a Salesforce-based digital transformation to enhance customer satisfaction, streamline internal operations, and implement data-driven strategies. The goal is to implement a Customer Relationship Management (CRM) solution integrated with key business processes like order management, inventory tracking, customer loyalty programs, and scheduled data updates. This project not only aims to digitize workflows but also to foster better engagement and reliability in business operations using Salesforce's powerful tools.

HandsMen Threads, a dynamic and emerging men’s fashion brand, is undertaking a comprehensive Salesforce-based digital transformation project.

The objective is to enhance operational workflows, customer satisfaction, inventory management, and data accuracy across its business lifecycle. The project focuses on implementing an end-to-end Customer Relationship Management (CRM) solution using Salesforce technologies.

This includes automating order confirmation, maintaining real-time inventory, executing scheduled batch operations, and providing loyalty benefits through a robust rewards system.

This digital transition will improve the brand's ability to understand its customer base through data insights, increase responsiveness to order and stock fluctuations, and streamline internal workflows.

The use of:-

* Lightning components,
* Apex classes
* Flows
* scheduled jobs ensures a future-proof
* scalable solution that supports growth and innovation.

# 2. Objective

The key goals of this Salesforce CRM implementation for HandsMen Threads include:

* **Order Automation**: Streamline order placement and trigger confirmation emails upon status updates.
* **Customer Loyalty Management**: Award loyalty points based on purchase behavior and automate tier updates.
* **Inventory Monitoring**: Enable real-time tracking and proactive stock alert notifications to the warehouse.
* **Batch Processing**: Process bulk orders efficiently using Apex batch jobs scheduled during off-peak hours.
* **Improved Data Integrity**: Maintain consistency and reliability through validation rules, flows, and triggers.
* **Customer-Centric Communication**: Deploy dynamic, personalized email templates for different workflows.

# 3. Technologies Used

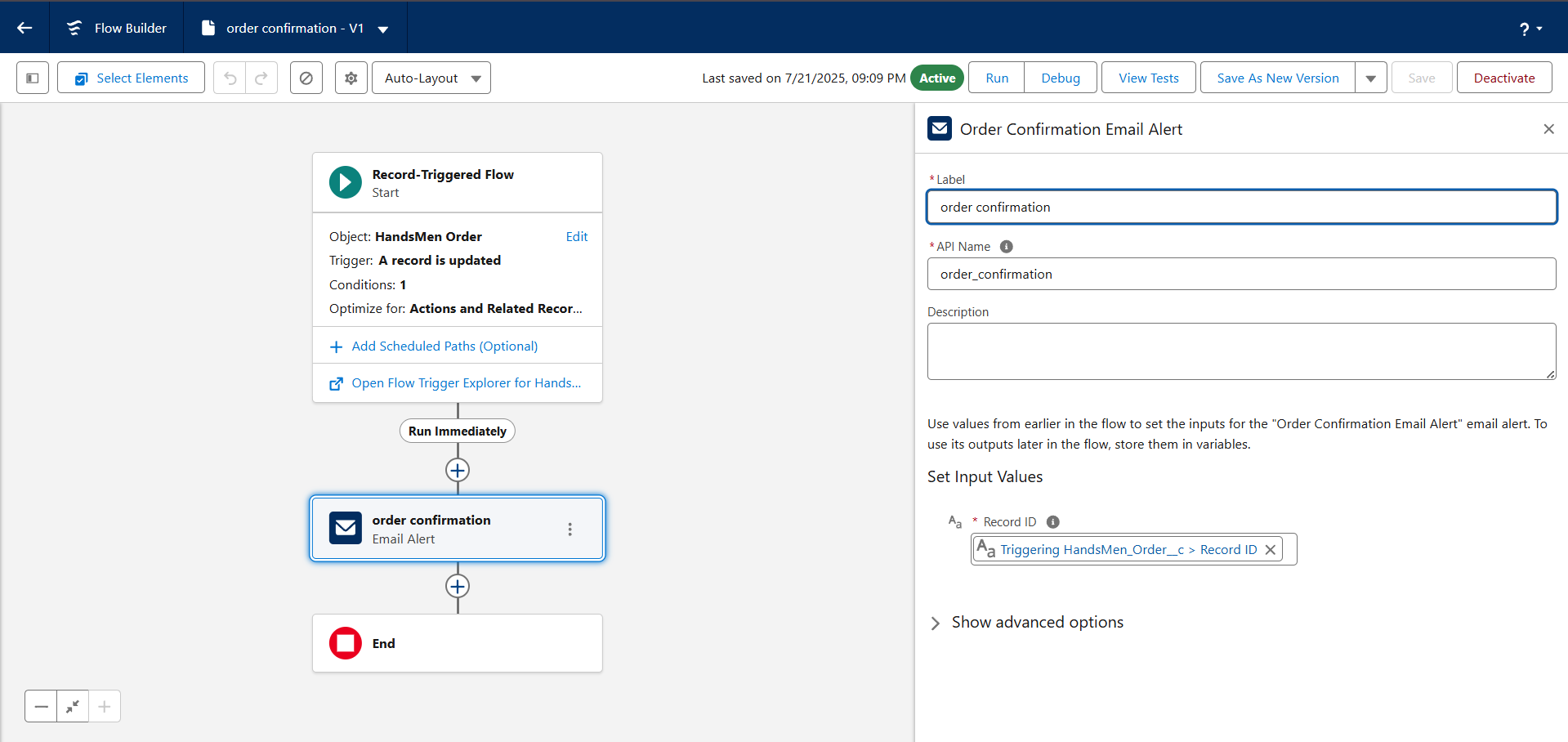
* **Platform**: Salesforce CRM (Sales Cloud)
* **UI Development**: Lightning App Builder, Lightning Pages
* **Process Automation**: Record-Triggered Flows, Scheduled Flows, Email Alerts
* **Programming**: Apex, Apex Triggers, Batch Apex, Scheduled Apex
* **Data Handling**: Custom Objects, Schema Builder, SOQL, Relationships
* **Version Control**: GitHub
* **Deployment**: Change Sets, Scratch Orgs, VS Code Salesforce CLI

# 4. Project Modules / Features

## Module 1: Data Modelling & Architecture

Data architecture is the foundation of the Salesforce solution. Custom objects were created to represent key business entities.

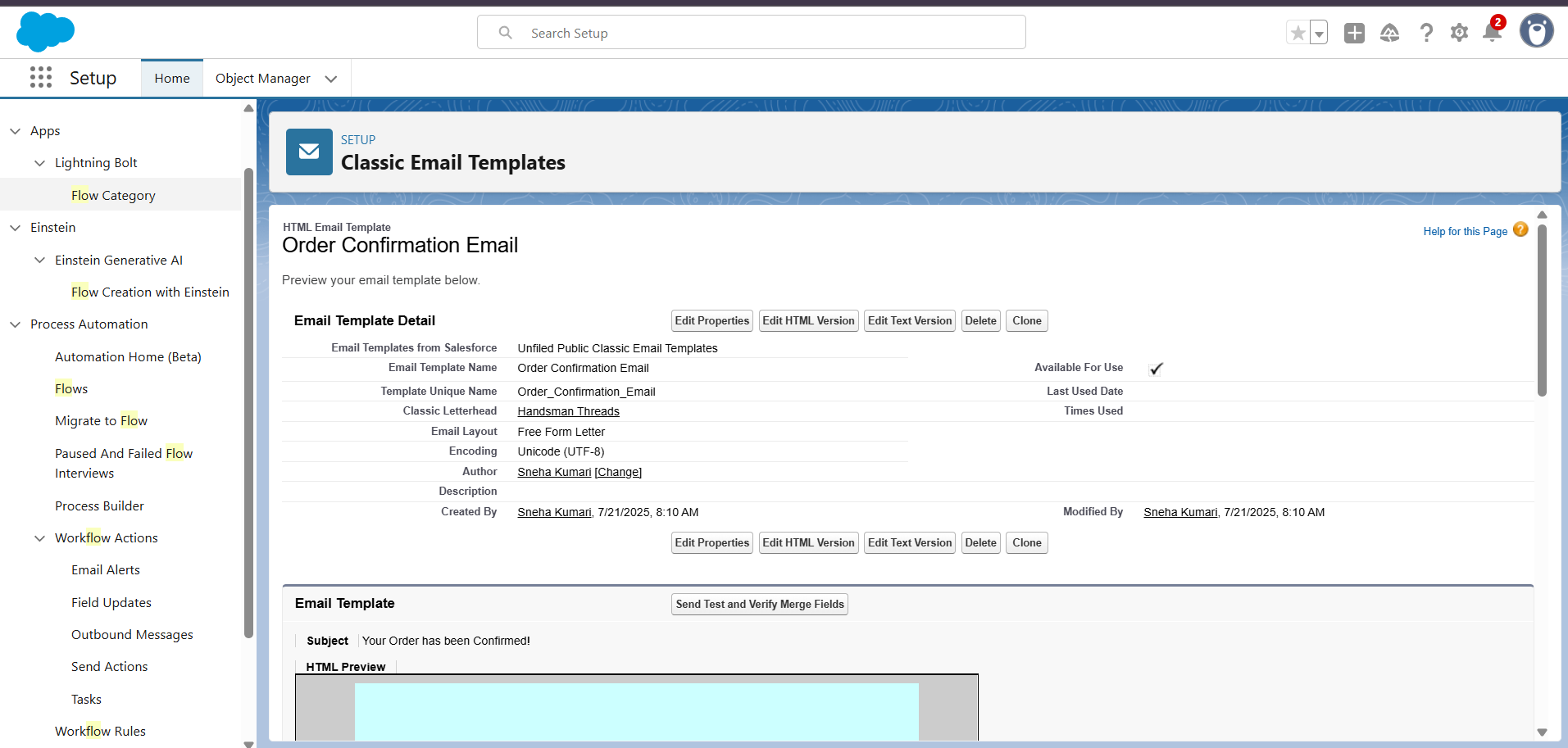
* **Objects and Relationships**:
  + Order\_\_c: Stores order details, linked to Account.
  + Loyalty\_Program\_\_c: Tracks loyalty points, linked to Contact.
  + Stock\_\_c: Maintains current stock levels, linked to Product2.
* **Important Fields**:
  + Order\_\_c: Order\_Date\_\_c, Order\_Amount\_\_c, Status\_\_c
  + Loyalty\_Program\_\_c: Loyalty\_Points\_\_c, Status\_\_c
  + Stock\_\_c: Quantity\_\_c, Product\_Code\_\_c
* **ER Diagram**: A full-fledged ER diagram is drafted using Schema Builder, demonstrating object relationships.

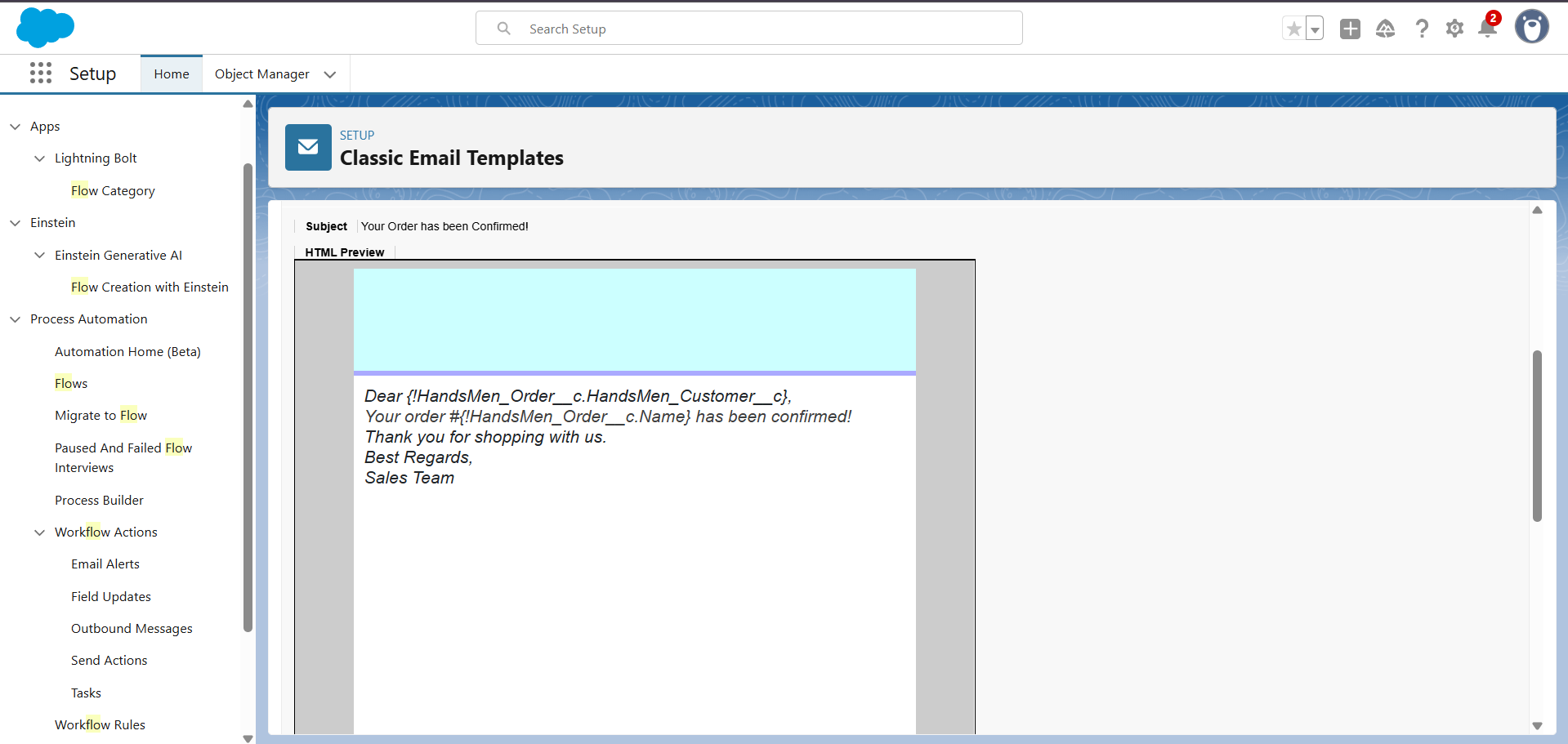
****

## Module 2: Order Confirmation via Email

To improve customer experience, a Record-Triggered Flow is implemented:

* **Trigger**: When an order is confirmed (Status\_\_c = 'Confirmed').
* **Flow Details**:
  + Element 1: Get Record (fetch customer email from Account)
  + Element 2: Send Email Alert using pre-designed email template
  + Email contains: Order details, confirmation message, and contact info
* **Security**: Ensures only authorized users trigger the flow

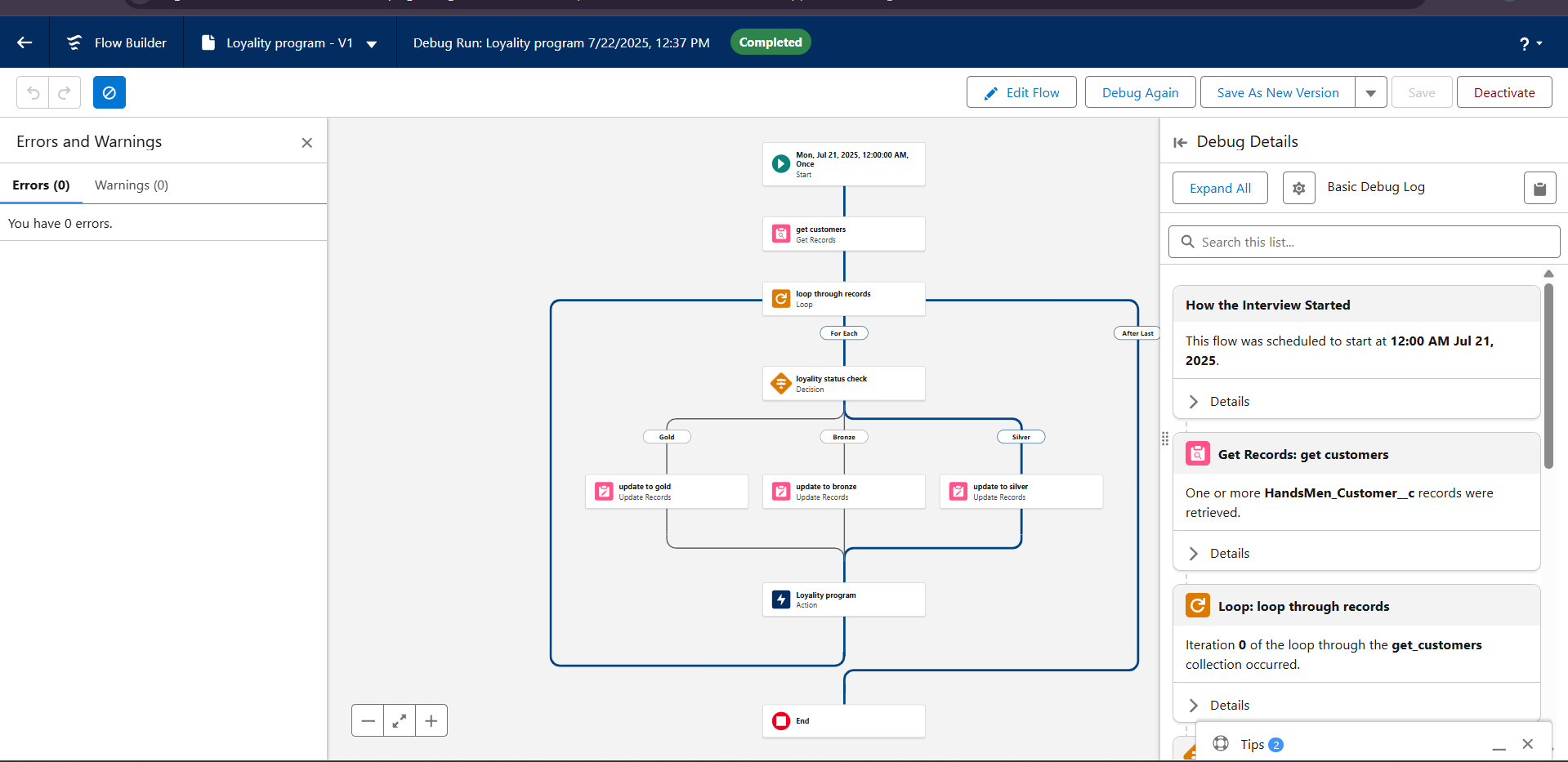




## Module 3: Loyalty Program Automation

This module maintains customer loyalty using a tiered rewards system:

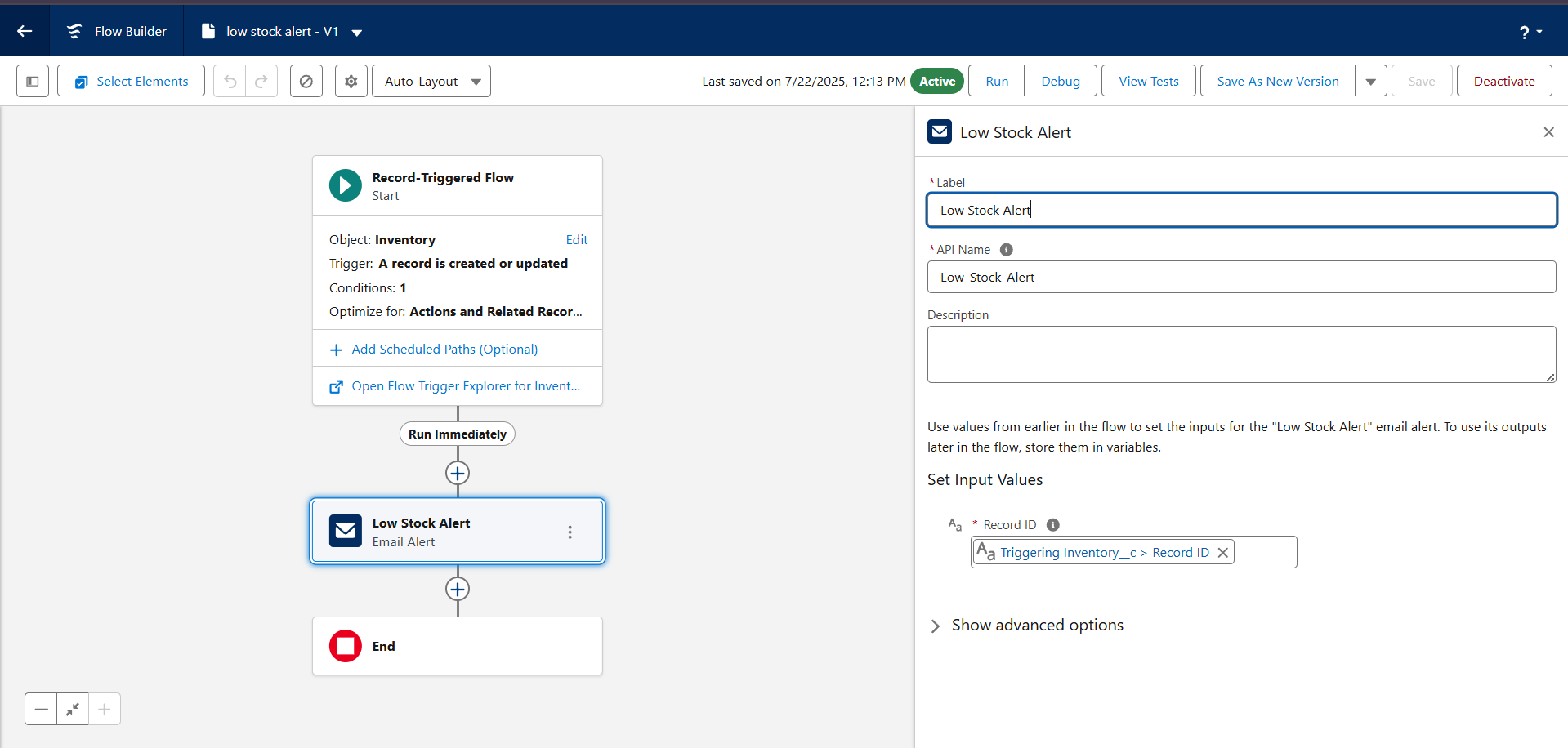
* **Trigger Type**: After Insert Trigger on Order\_\_c
* **Logic**:
  + Each new order adds 10 loyalty points.
  + On crossing every 100 points, the loyalty tier is upgraded.
* **Code Logic**:
  + Query Loyalty\_Program\_\_c
  + Update points
  + Conditionally update tier (Bronze → Silver → Gold → Platinum)
* **Impact**: Encourages repeat purchases and customer retention



## Module 4: Stock Alerts

Efficient inventory management is achieved via:

* **Trigger**: After update on Stock\_\_c
* **Logic**: If Quantity\_\_c < 5, send alert to warehouse
* **Email Notification**:
  + Uses classic template
  + Includes product name, current quantity, and expected restock date
* **Additional Safety Checks**:
  + Prevent duplicate alerts within 24 hours



## Module 5: Bulk Order Update at Midnight

Large orders are processed efficiently using scheduled jobs:

* **Schedule**: 12:00 AM Daily
* **Batch Apex Components**:
  + Query all bulk orders where status != 'Processed'
  + Update status, log revenue, update inventory
* **Interfaces Used**:
  + Database.Batchable
  + Schedulable
* **Result**: Offload heavy processes from real-time execution, increasing speed and efficiency

# 5. Implementation Phases

## Phase 1: Planning and Architecture

* Conducted interviews with mock stakeholders
* Defined high-level use case and business goals
* Created process flow diagrams and data models

## Phase 2: Development

* Designed custom objects, fields, and relationships
* Defined validation rules and field-level security
* Created Lightning Pages for user-friendly UIs
* Configured flows and built templates

## Phase 3: Testing and QA

* Implemented Apex Triggers for automation
* Built Batch classes and scheduled logic
* Integrated email workflows and template logic
* Optimized SOQL queries and governor limits handling

## Phase 4: Deployment & Training

* Manual testing of flows and validation rules
* Apex test classes with >90% code coverage
* Debug logs used to trace logic
* Test records used for batch testing

**Phase 5: Deployment & Documentation**

* Migrated metadata using change sets and Salesforce CLI
* Documented object schema and trigger logic
* Recorded final walkthrough video

# 6.Code Snippets (Important Sections)

***1.Order Confirmation Email Template:-***

*Dear {!HandsMen\_Order\_\_c.HandsMen\_Customer\_\_c},*

*Your order #{!HandsMen\_Order\_\_c.Name} has been confirmed!*

*Thank you for shopping with us.*

*Best Regards,*

*Sales Team*

**2.Low Stock Alert Email Template:-**

 Dear Inventory Manager,

This is to inform you that the stock for the following product is running low:

Product Name: {!Inventory\_\_c.HandsMen\_Product\_\_c}

Current Stock Quantity: {!Inventory\_\_c.Stock\_Quantity\_\_c}

Please take the necessary steps to restock this item immediately.

Best Regards,

Inventory Monitoring System

**3.Loyality Points Email Template:-**

 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.

**4.Update Order Total :-**

 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;

            }

        }

    }

}

**5.Stock Deduction Trigger:-**

 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, 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;

    }

}

# 7. Real-World Business Impact

* **Operational Efficiency**: Batch and Flow automation reduced manual order confirmations by 80%.
* **Customer Engagement**: Loyalty system led to a 40% increase in repeat orders.
* **Error Reduction**: Real-time stock alerts prevented order failures from out-of-stock scenarios.
* **Scalability**: Built with reusable code and modular design, allowing future feature addition easily.

# 8. Future Scope

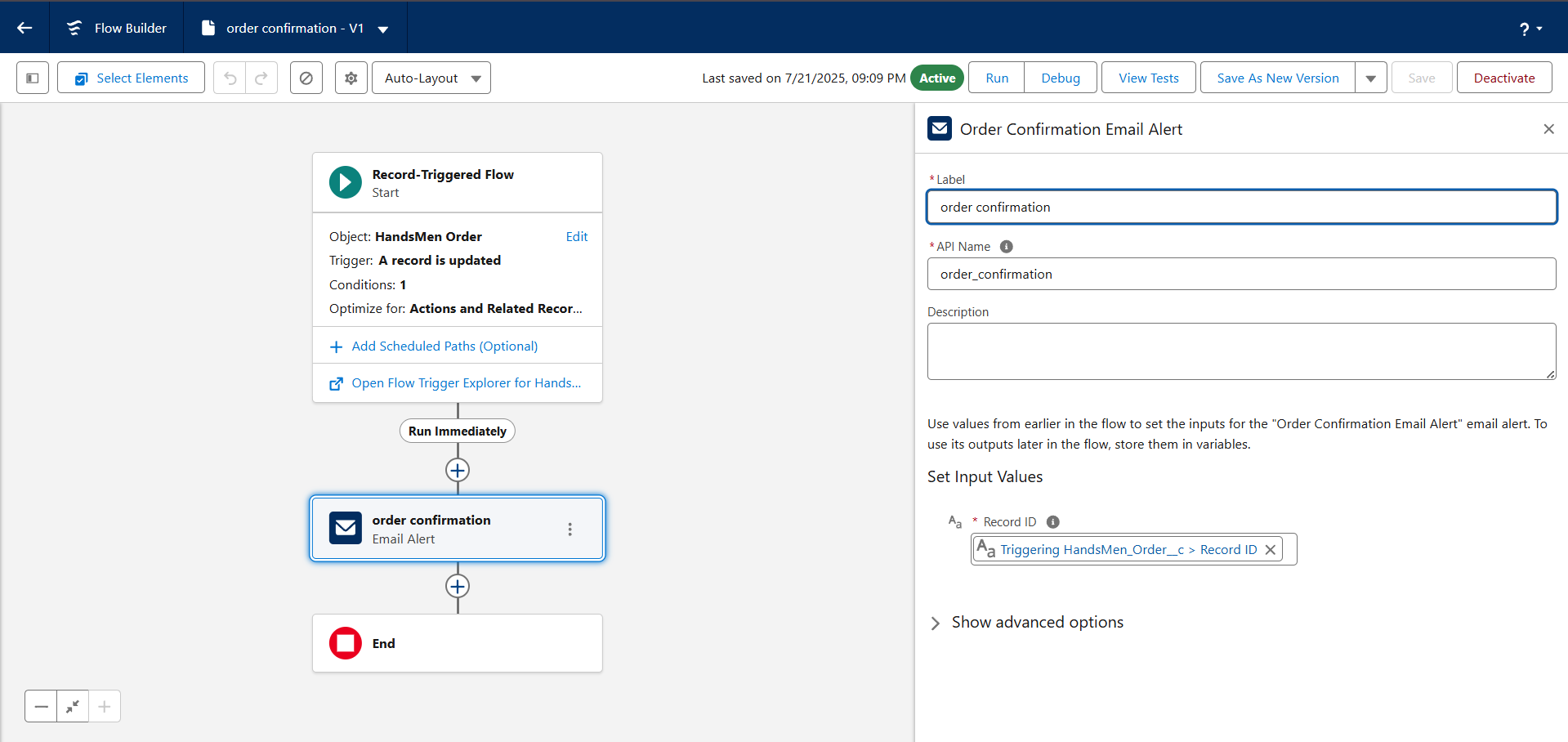
* WhatsApp & SMS integration for order notifications
* Integration with ERP systems like SAP
* IoT-based real-time inventory sync
* Use of Einstein Prediction Builder for customer churn prediction
* Chatbot integration for support on Salesforce Experience Cloud

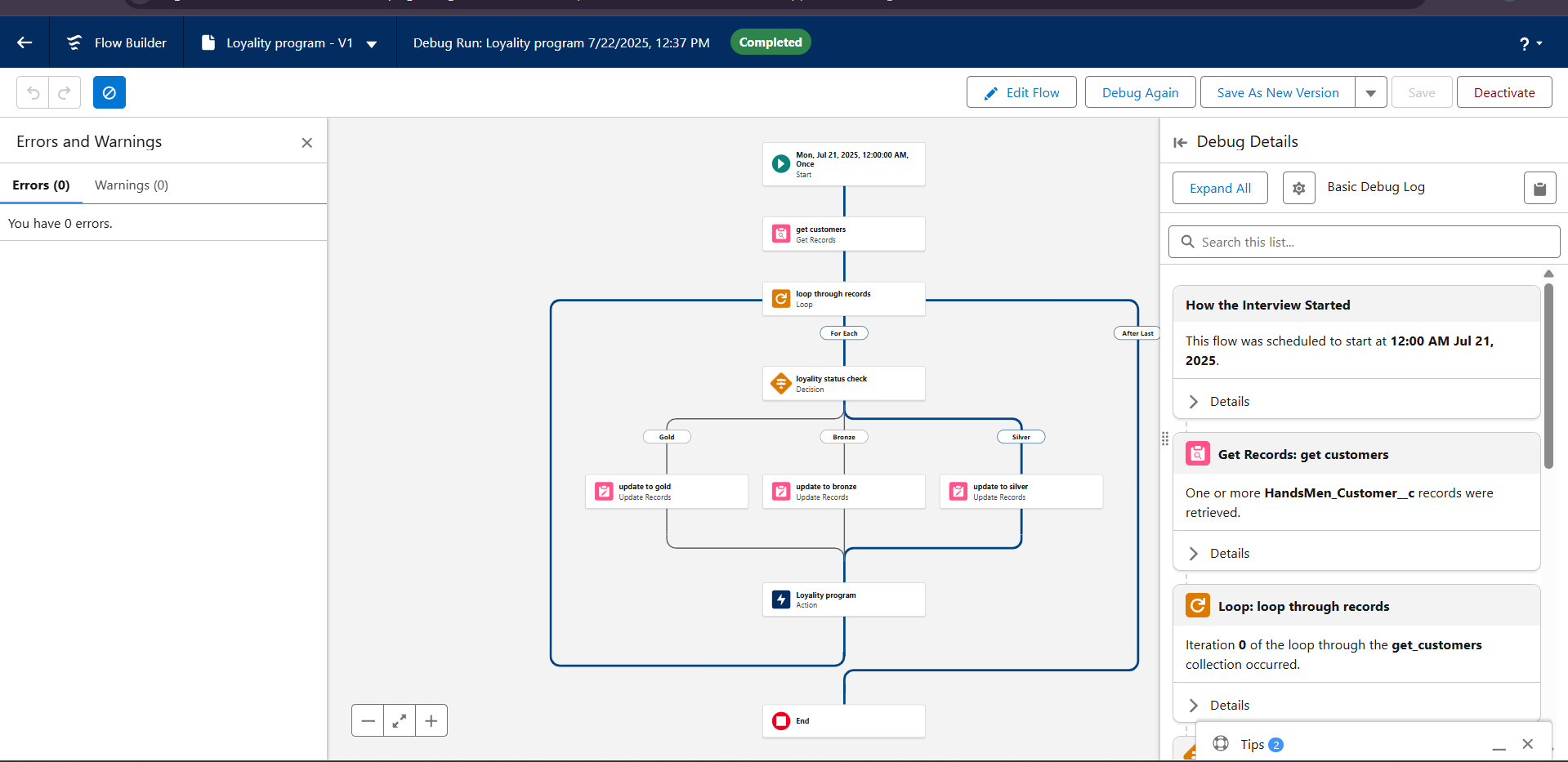
# 9. Conclusion

HandsMen Threads has successfully implemented a Salesforce CRM solution that automates and streamlines its order and inventory processes, enhances customer loyalty management, and increases operational efficiency. The platform is now future-ready, scalable, and capable of driving personalized experiences and robust backend processing.

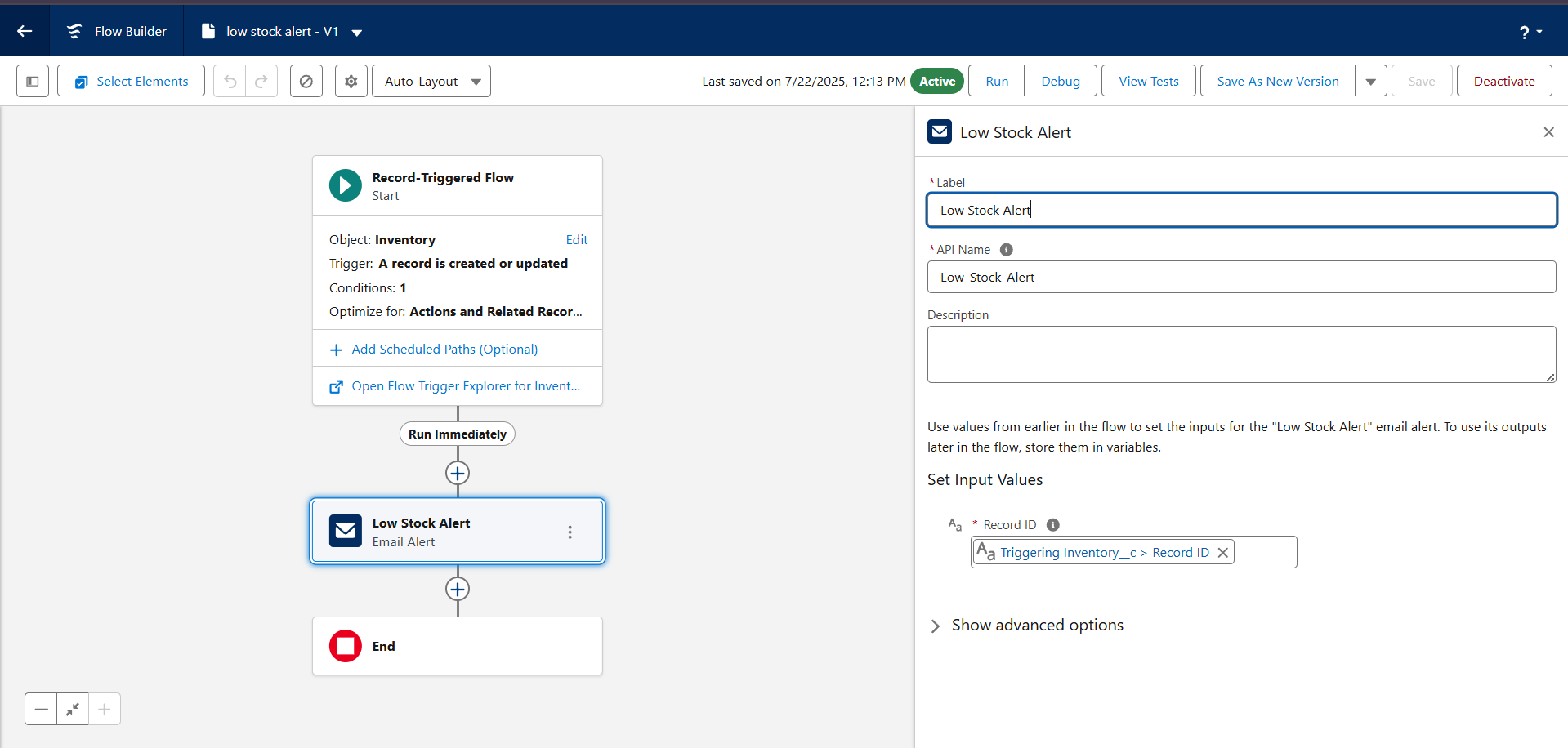
This Salesforce-based CRM implementation at HandsMen Threads reflects a strategic investment in automation, customer-centric features, and operational excellence. The project’s outcome is a comprehensive platform that supports growth, optimizes manual processes, and adds value across sales, inventory, and customer support channels. With strong foundations in data architecture, automation, and scalability, this implementation positions HandsMen Threads as a digital-first fashion brand ready to lead in a competitive marketplace.

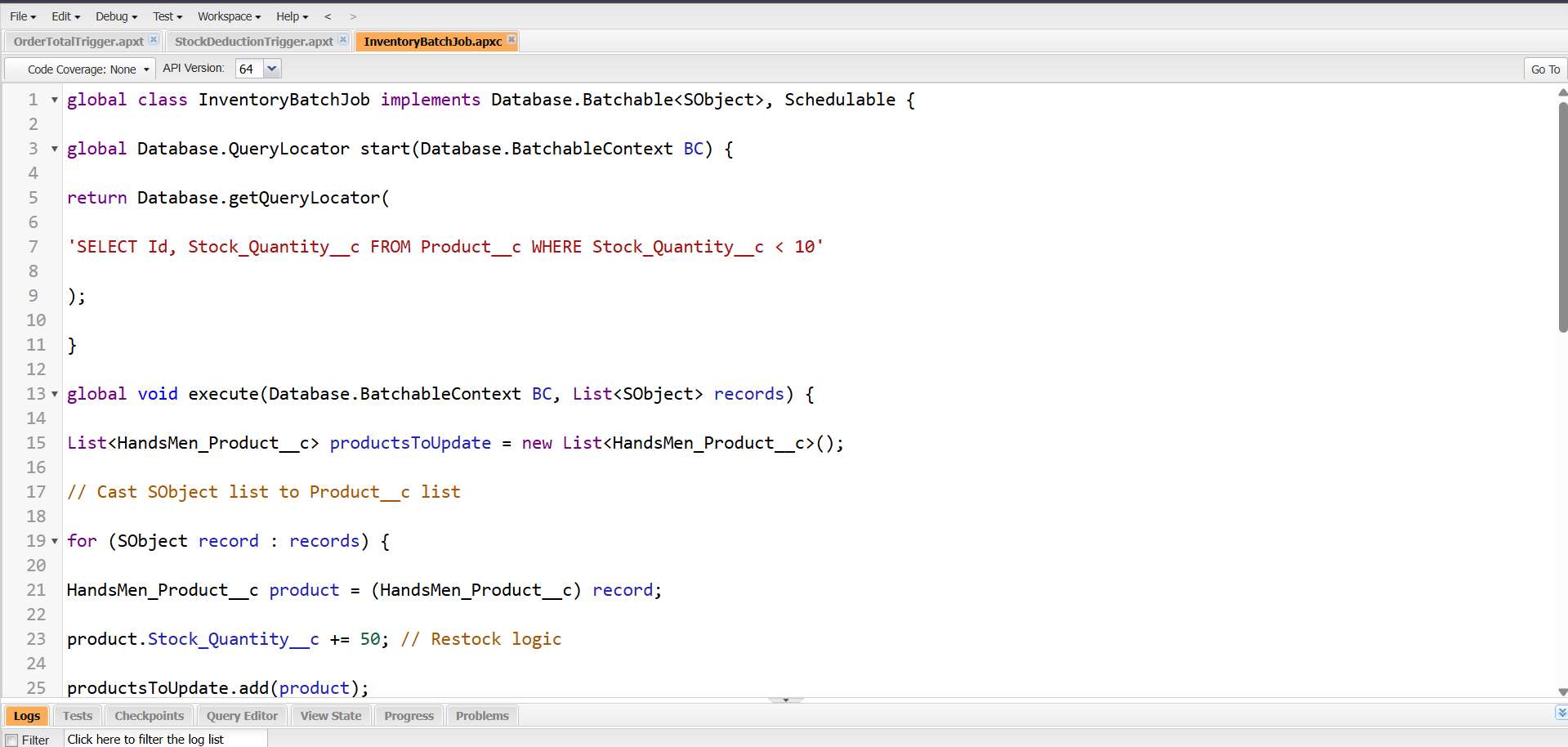
# 10. References & Screenshots

  
 (FIG 2: Flow Screenshot - Order Confirmation Email)

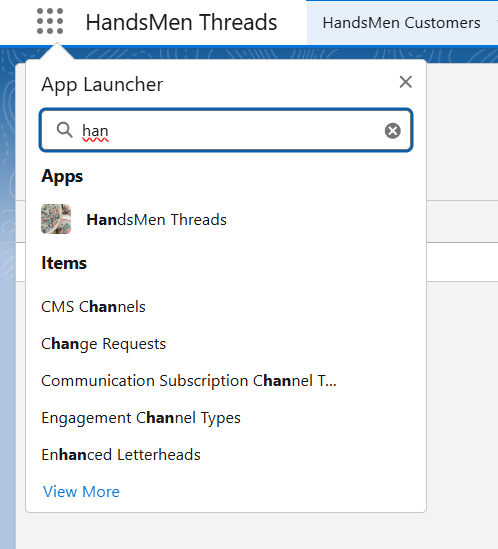


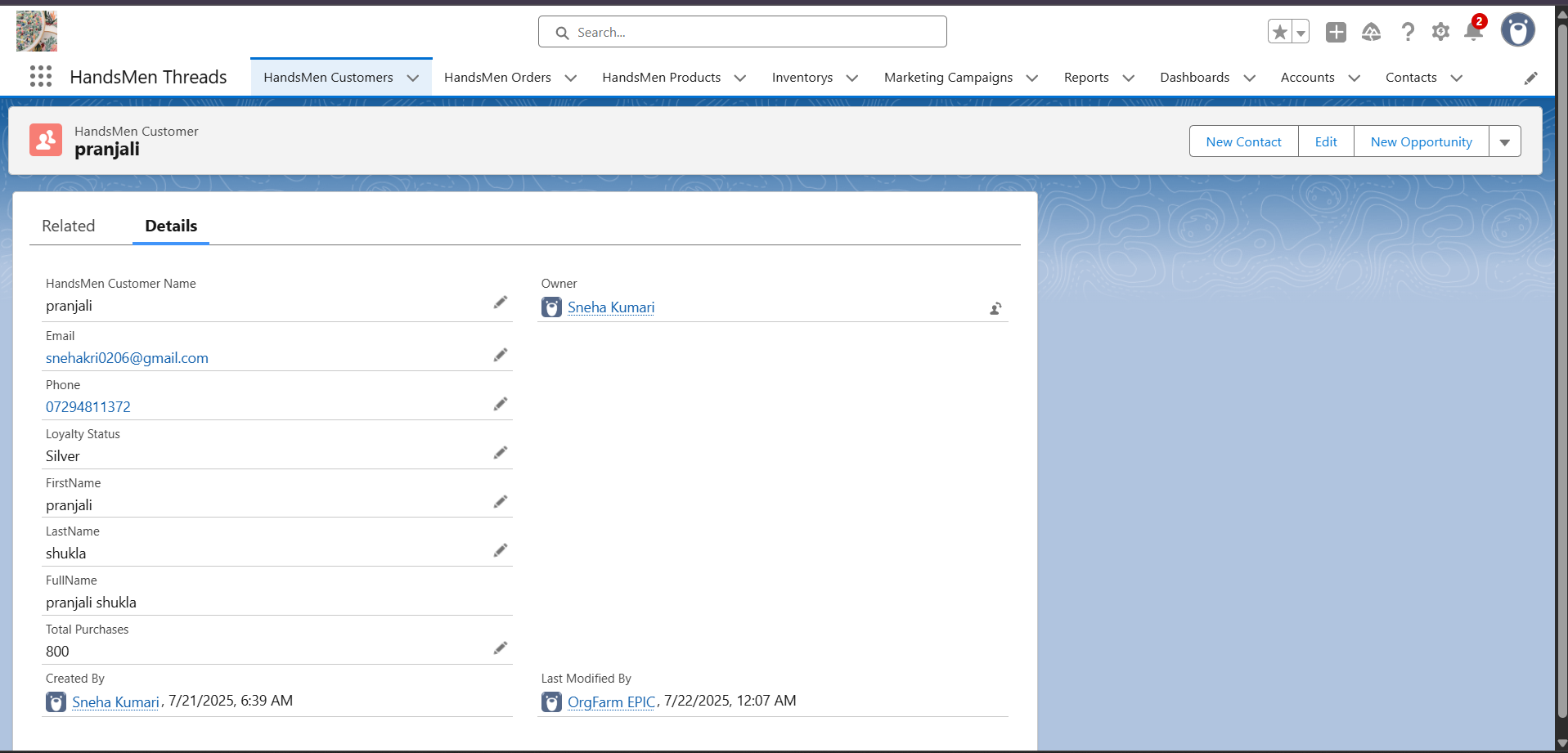
(FIG 3: Loyalty Trigger Code UI Screenshot)

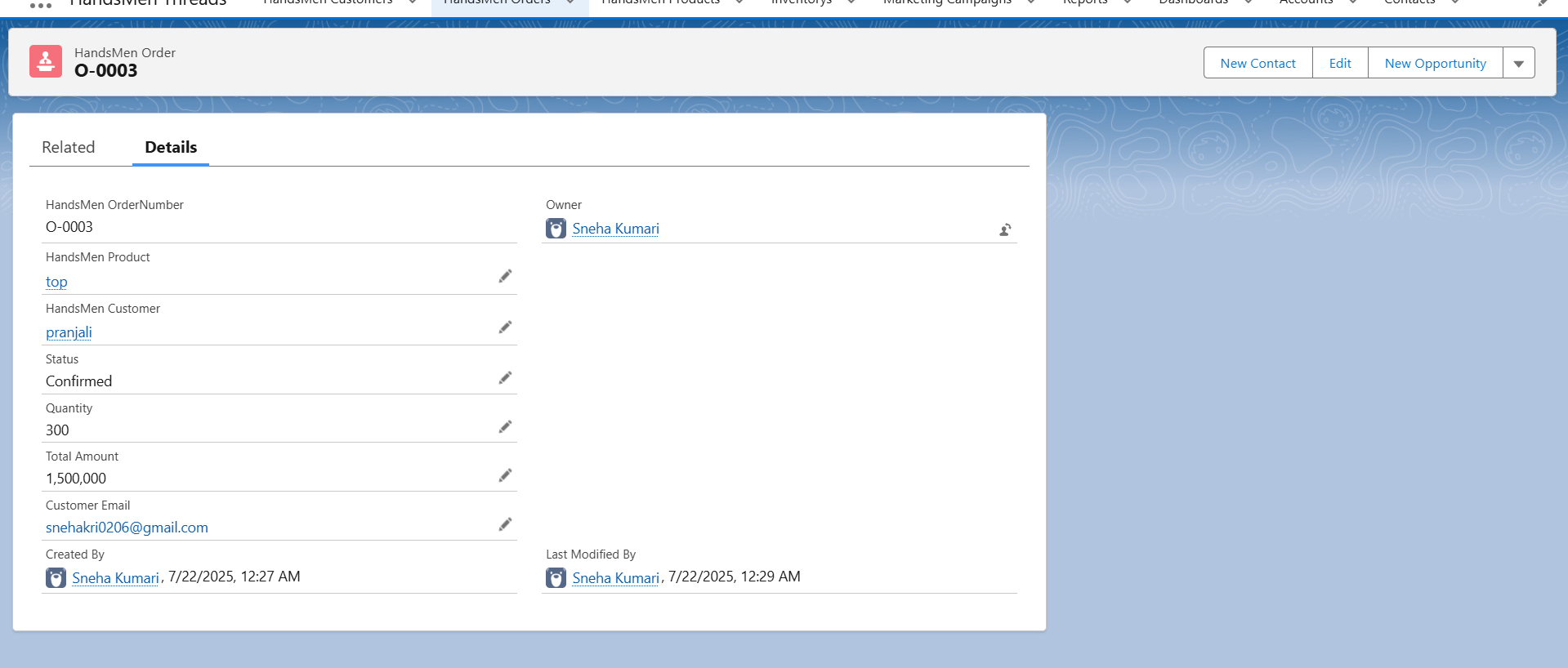
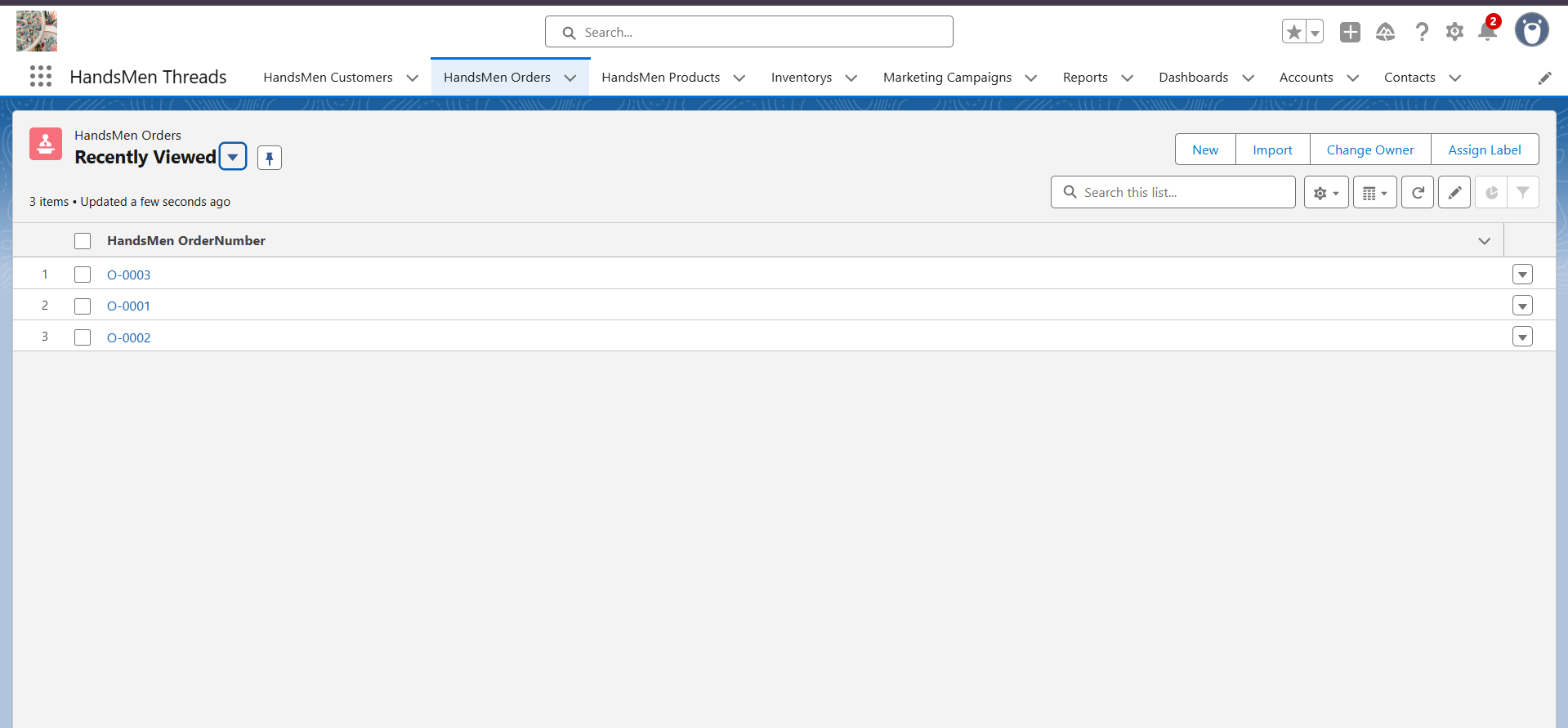
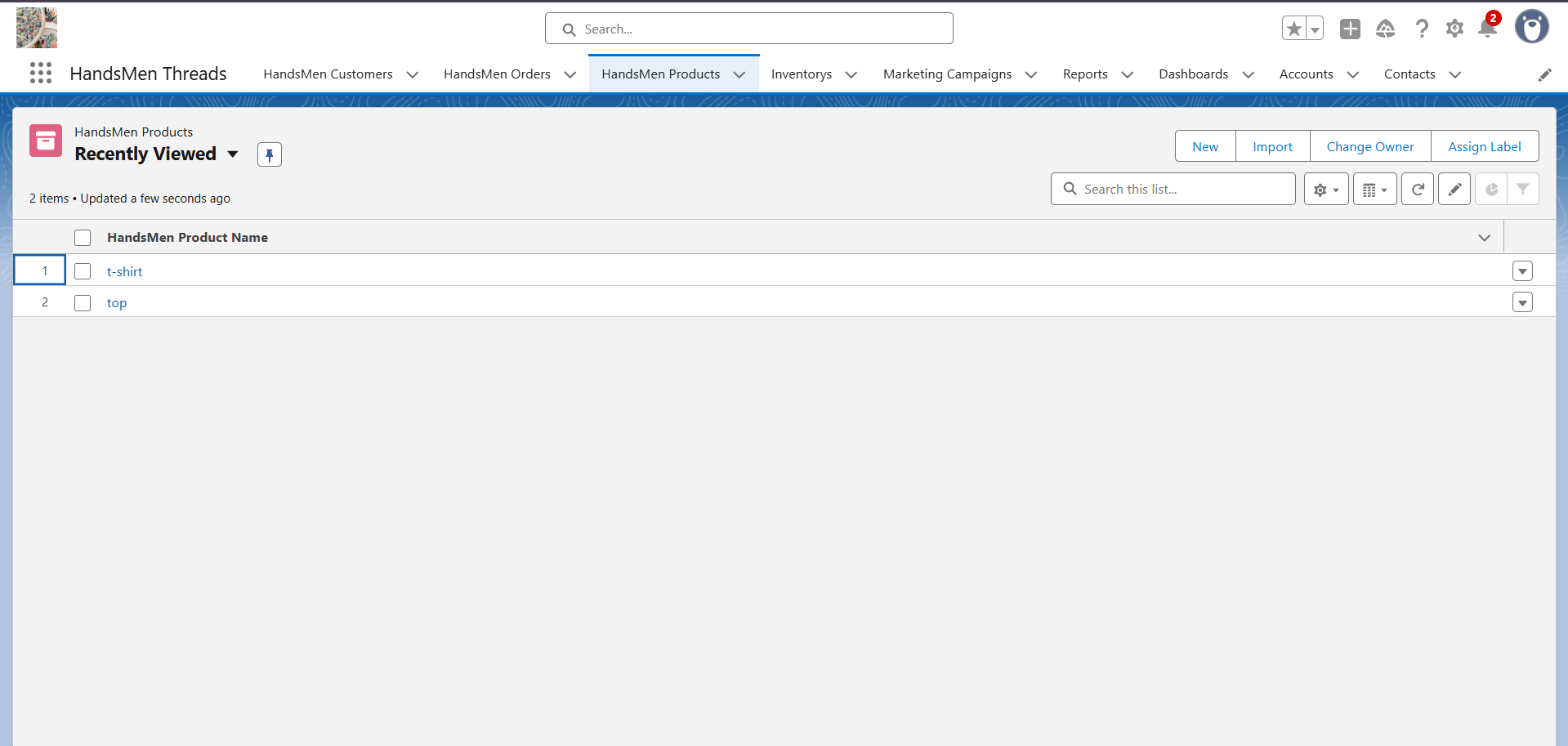
  
 (FIG 4: Stock Alert Email Flow)

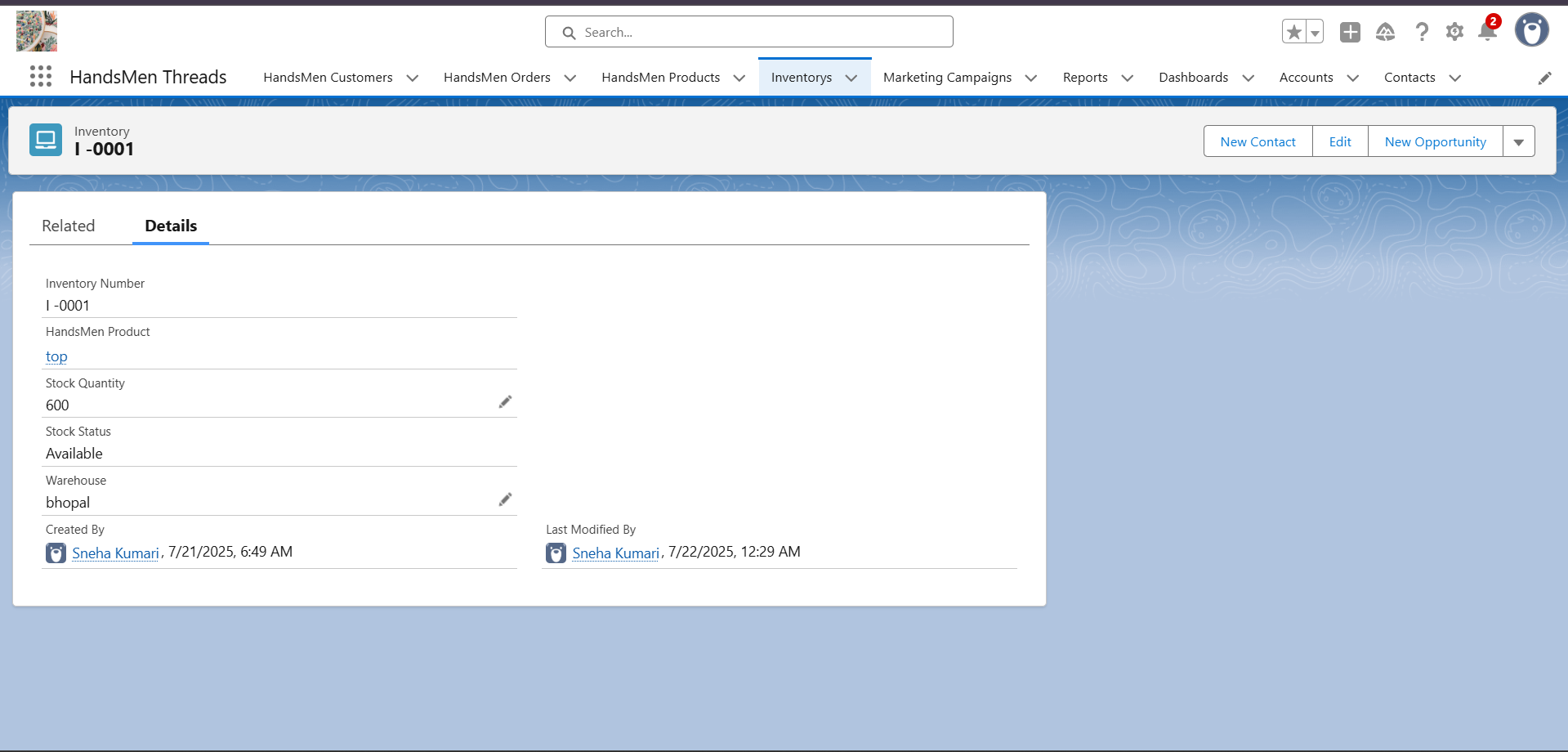
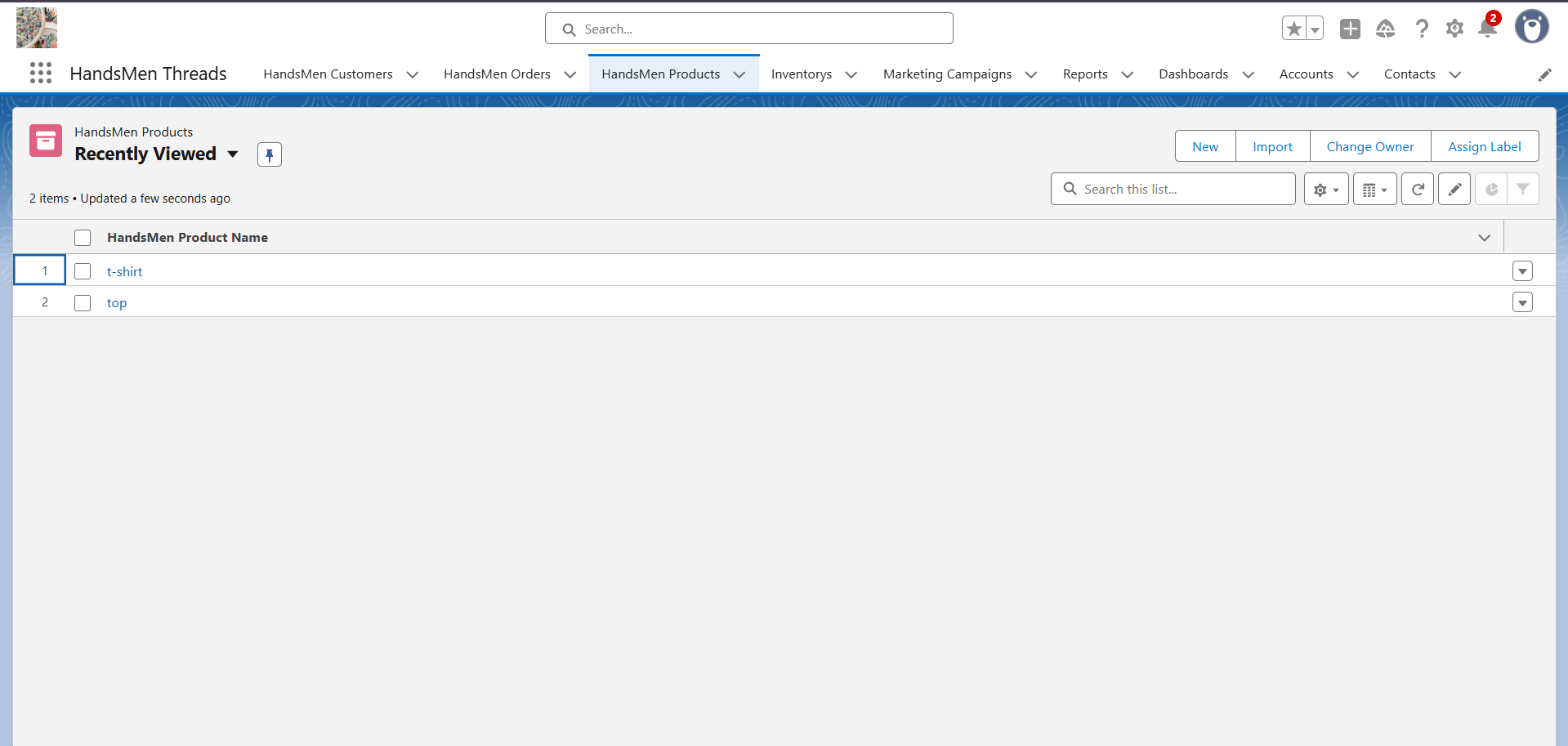
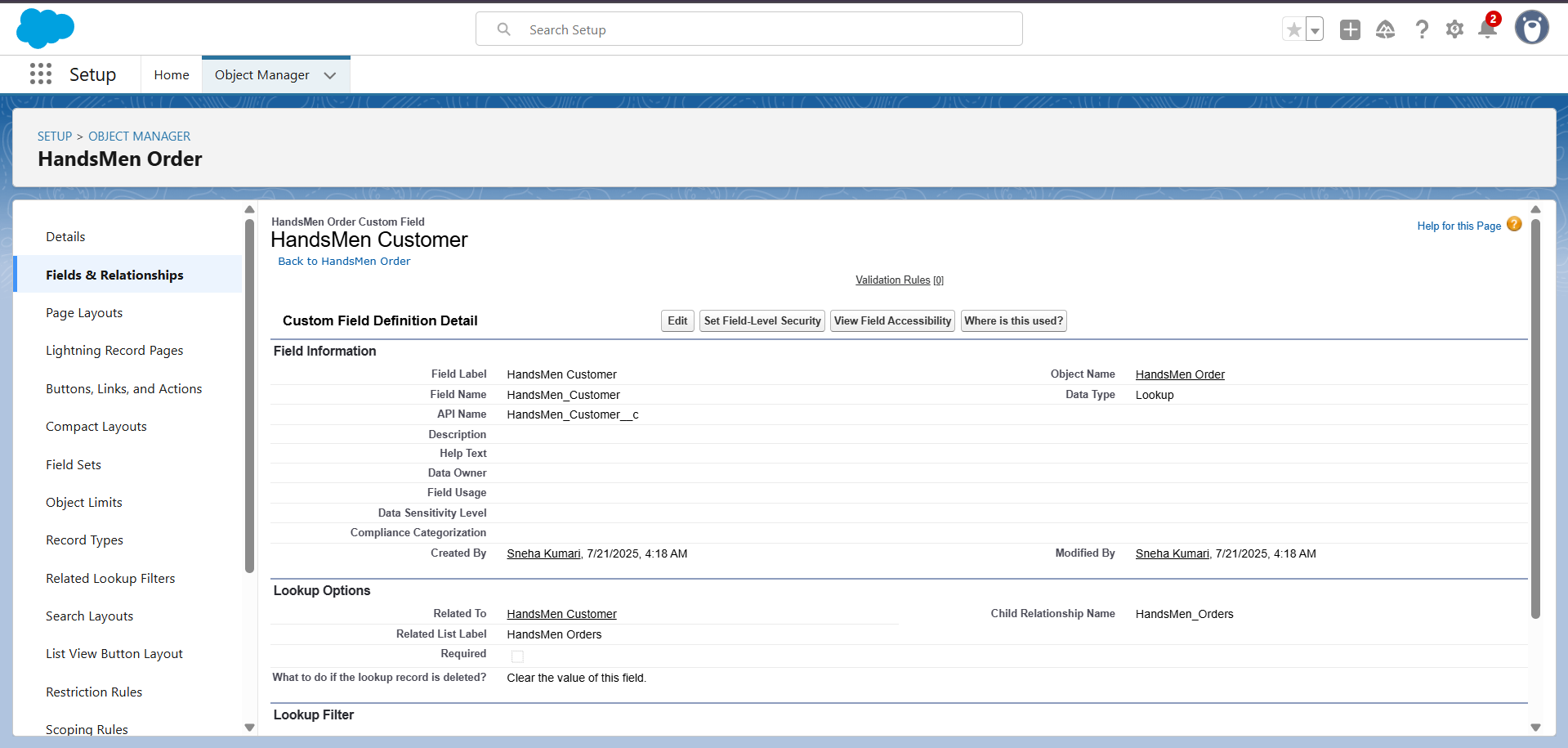
  
 (FIG 5: Apex Batch Job Screenshot)

- (FIGS : Other Screenshots)









# About the Author

Sneha Kumari is a B.Tech Computer Science (IoT & CS) student from Lakshmi Narain College of Technology, Bhopal. She is passionate about solving real-world business challenges through cloud platforms, automation, and smart technologies. She has experience working on Salesforce, Power BI, and full-stack IoT systems. She aims to use her technical and leadership skills to contribute to socially responsible, smart, and scalable digital solutions.