HandsMen Threads – Salesforce CRM Project Documentation

Project Overview

HandsMen Threads is a custom Salesforce CRM platform tailored for tailoring, fashion, and boutique businesses. It provides end-to-end digitalization of service, customer, and inventory management through Salesforce automation tools. With a user-centric approach, it improves operations from order booking to stock alerts, loyalty tracking, and campaign engagement. The system minimizes manual tasks and empowers teams with real-time updates, effective communication, and insightful dashboards.

Key Business Needs Solved:

- Eliminates manual entries for orders and inventory.
- Automates email notifications for better customer communication.
- Ensures real-time stock monitoring with low-stock alerts.
- Enhances customer retention with loyalty programs and marketing insights.

Objectives

This project focuses on streamlining business operations and enhancing customer satisfaction through intelligent Salesforce automation.

Core Objectives of the HandsMen Threads Application:

- Workflow Automation: Automate order processing, stock management, and customer updates.
- Data Integrity: Enforce accurate and clean data entry from the UI via validations.
- Alerts & Notifications: Real-time email alerts to customers and staff for stock/order updates.
- Loyalty Program: Track and update customer loyalty levels automatically.
- Insights: Enable leadership to make informed decisions using reports and dashboards.

Phase 1: Requirement Analysis & Planning

Understanding Business Requirements

Here, we gathered and analyzed the actual pain points from a tailoring/fashion business:

- Customers want automated order status updates via email.
- The inventory manager needs real-time low-stock alerts to avoid order delays.
- Loyalty should be calculated based on customer order frequency or amount.

- Admins want daily automated updates for stock and bulk orders.

Project Scope & Objectives

This phase defines what's included in the project and sets clear boundaries:

- Build custom objects that mirror real-world entities (Customer, Product, Inventory).
- Add logic to automate stock changes, order status, and email triggers.
- Schedule jobs for daily stock restocking and perform backend automation.

Data Model Design

A well-designed data model is the foundation of Salesforce:

🔹 **Custom Objects Used:**

- HandsMen_Customer__c – Stores customer details, contact info, loyalty status.
- HandsMen_Order__c – Contains order information like product, quantity, and status.
- HandsMen_Product__c – Represents products (fabrics, accessories).
- Inventory__c – Tracks stock level for each product.
- Marketing_Campaign__c – Manages campaigns like sales, promotions.

🔹 **Relationships Between Objects:**

- HandsMen_Order → Customer: Lookup to know who placed the order.
- Product → Order: Lookup to see which product is ordered.
- Inventory → Product: Master-detail so deletion cascades; Inventory can't exist without a Product.
- Campaign → Customer: Lookup to track which customer was targeted.

🔹 **Key Fields:**

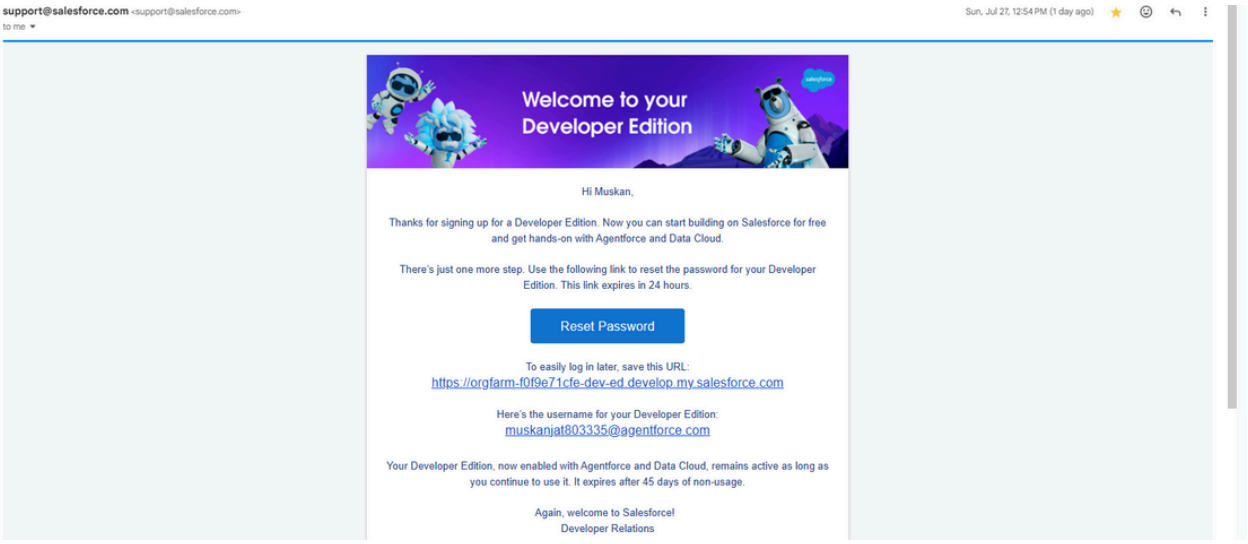
- Email, Phone, Loyalty_Status__c (Customer)
- Quantity__c, Status__c (Pending, Confirmed, Rejected), Total_Amount__c (Order)
- Stock_Quantity__c (Product)
- Lookup fields to relate records correctly.

Phase 2: Salesforce Backend Development & Configurations

Environment Setup

- Developer Org: Created a dedicated org for building and testing.
- Developer Console: Used for Apex classes and triggers.
- Scheduled Jobs Panel: For setting and managing batch jobs (e.g., Inventory restock at 12 AM).

How set up Developer Org for Project



Automation Using Apex

Apex Classes & Triggers:

- OrderTriggerHandler: Contains business logic for checking order quantities. Ensures stock isn't overbooked or negative.
- OrderTrigger: Fires on before insert and before update of orders. It uses handler methods to apply validations.
- InventoryBatchJob: A Schedulable class that runs every midnight and checks products with quantity below 10 units, then creates tasks or alerts.

Apex Batch Classes and Triggers

InventoryBatchJob.apxc

Code Coverage: None API Version: 64

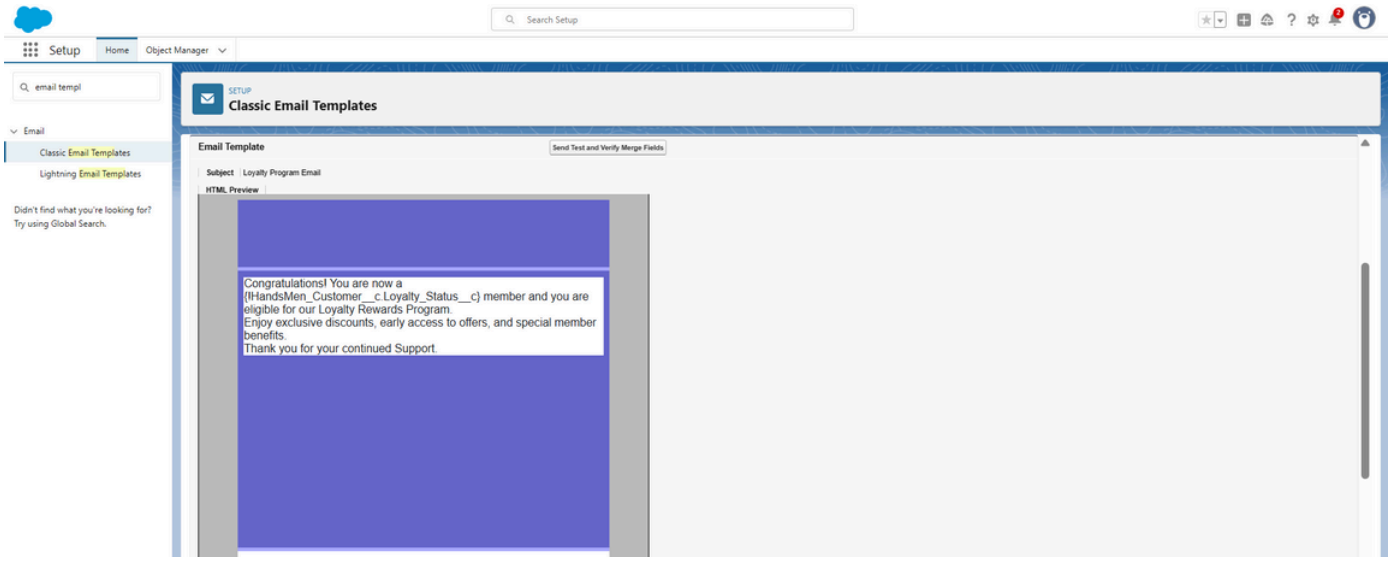
```
global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {

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);

}
```

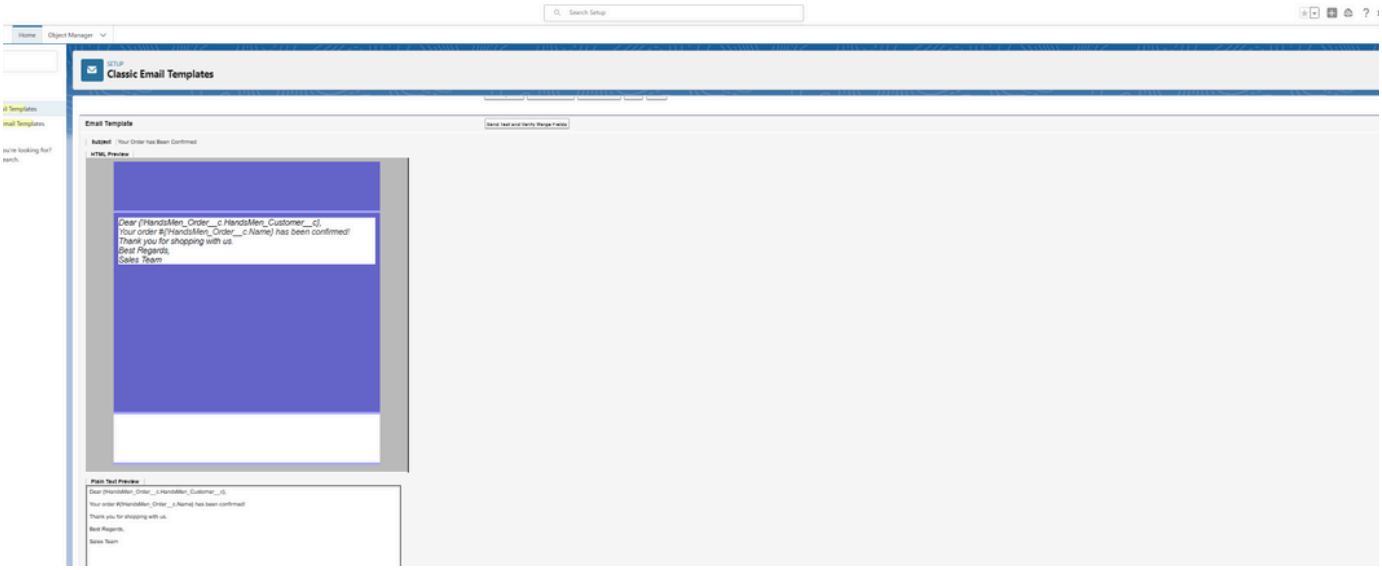
```
for (HandsMen_Order__c order : Trigger.new) {
    if (order.Status__c == 'Confirmed' && order.HandsMen_Product__c != null) {
        productIds.add(order.HandsMen_Product__c);
    }
}

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;
```

Email Alerts & Templates

- Custom HTML/Plain Text email templates used for:
  - Order Confirmation
  - Stock Running Low
- Email Alert Actions tied with Flows and Workflow Rules to send messages automatically to the user or admin.

Email Alert Actions



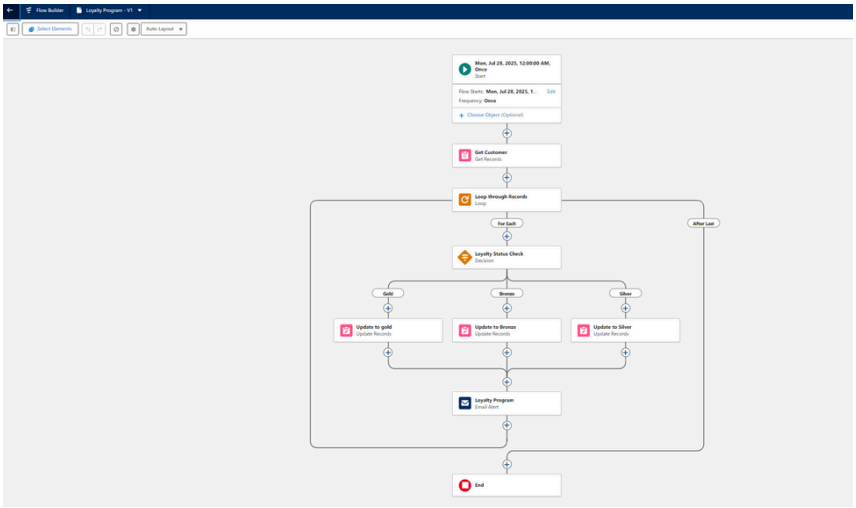


Flows Used

Types of Flows:

- **Record-Triggered Flows:**
  - On new Orders → Sends confirmation email.
  - On Inventory updates → Sends low-stock alert.
  - On Customer updates → Updates Loyalty Status based on order count/amount.
- Scheduled Flow:
  - Runs daily to update bulk records, stock level adjustments, or re-calculate loyalty status.

Representation of Autolauched Flow



Permission Set

- **Permission_Platform_1:**
  - Created to manage access control (which objects/fields users can see/edit).

- Assigned to users like Inventory Manager, Customer Support, etc., instead of altering profiles.

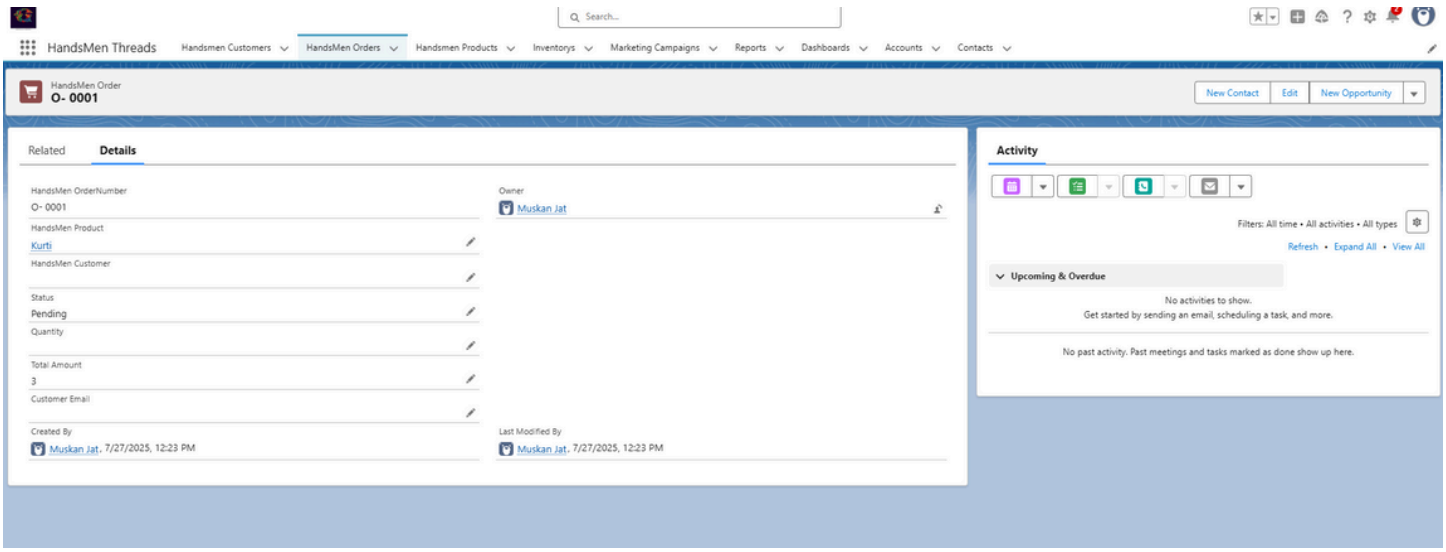
Phase 3: UI/UX Development & Customization

Lightning App Configuration

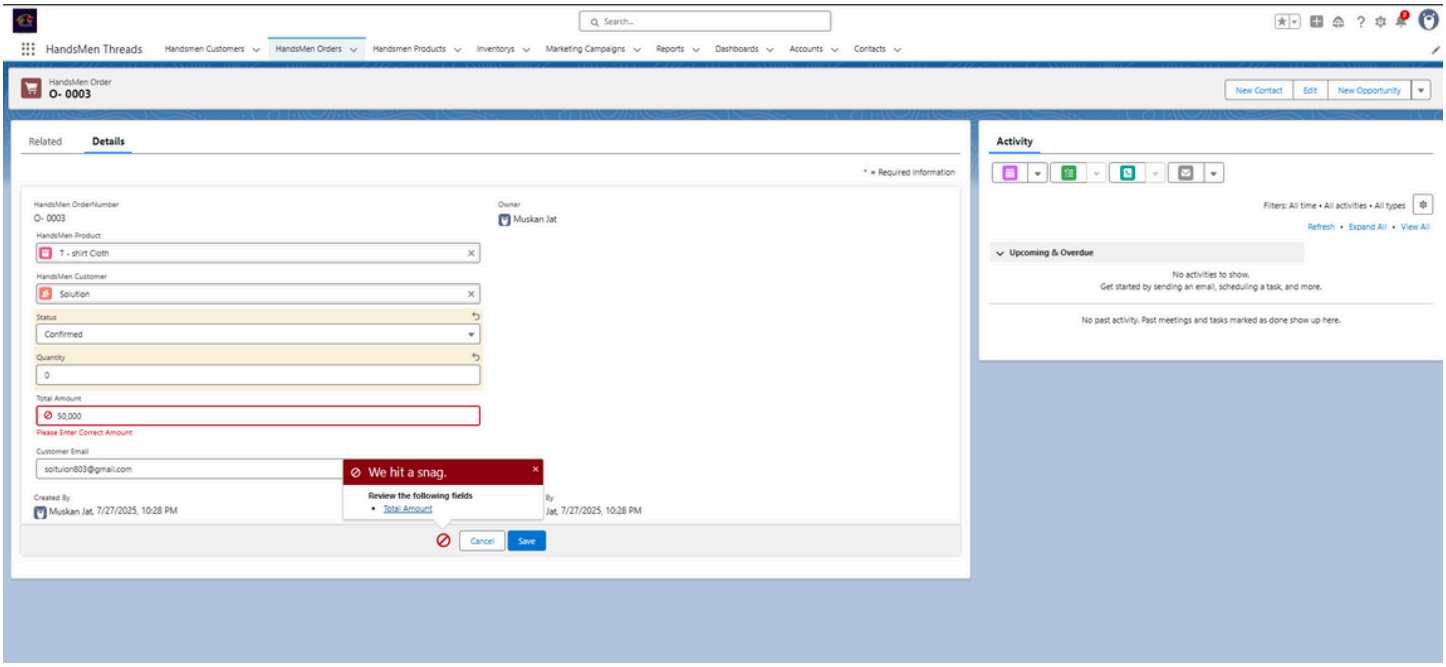
- Created HandsMen Threads Application with a proper logo and description
- Application Description: A CRM platform to manage tailoring orders, stock, and customer relationships.

Navigation & Layouts

- Tabs Created For:
  - HandsMen Order
  - HandsMen Customer
  - Inventory
  - Marketing Campaign
  -
- **HandsMen Threads Application User Interface**



Validation Rules Testing



Page Customization

- Dynamic Forms & Record Pages:
  - Display fields conditionally based on role/record type.
  - Use Lightning App Builder to arrange components and visibility rules.

User Profiles

- Admin: Full access, can view and manage everything.
- Inventory Manager: Can access Inventory, Products, and Orders.
- Customer Support: Can view customers, create orders, view campaign interactions.

Phase 4: Testing & Security

Data Quality and Security

- Field History Tracking: Enabled on Inventory and Order objects to track changes.
- Validation Rules: Example:
  - Customer Email format validation.
  - Order Quantity must be ≥1.
- Profiles & Roles: Defined hierarchy access (Manager > Support Agent > Viewer).
- Testing: Manual and automated testing with screenshots for QA documentation.

Phase 5: Deployment, Maintenance & Troubleshooting

Deployment

- Used Change Sets to move metadata from sandbox/dev org to production.

Monitoring & Maintenance

- Debug Logs: Captured for errors and performance.
- Scheduled Jobs: Monitored using Salesforce UI.
- Flow Errors: Monitored via email alerts and Flow Error Logs.

Troubleshooting

- Error messages from validation rules and flow error emails help debug issues quickly.
- Apex Exception Handling: try-catch used in Apex logic; logs stored using custom logging if needed.

Conclusion

HandsMen Threads is a robust and scalable CRM system tailored for tailoring businesses. The project successfully leverages Salesforce's features — custom objects, Apex automation, Flows, and Dashboards — to meet operational needs and enhance customer engagement. Its modular design ensures scalability, and it can be adopted by any small-to-medium tailoring unit seeking automation and insight.

Future Enhancements

Here are potential future upgrades:

1. WhatsApp & SMS Integration: Notify customers via WhatsApp or SMS using Twilio.
2. Chatbot Support: LWC-powered chatbot for real-time customer queries.
3. AI Order Suggestions: Recommend products based on purchase history using Agentforce .
4. Responsive Mobile Layout: Mobile-first redesign for field agents.
5. Multi-language Support: Add translations using Custom Labels and Translation Workbench.