HandsMen Threads Salesforce Project Report

Intern Details:

Name: Pooja Soni

Location: Bhopal, Madhya Pradesh

College: Lakshmi Narain College of Technology

Platform: Salesforce Virtual Internship by SmartBridge

Project Title:

HandsMen Threads Salesforce Project

Use Case Overview:

HandsMen Threads, a forward-thinking organization in the fashion industry, aimed to streamline data operations and improve customer relationships through Salesforce. The objective was to develop a robust system that enables seamless business process automation, ensures data quality, and enhances overall operational efficiency.

Core Functionalities Implemented:

Custom Data Model:

1. Created custom objects:

- HandsMen_Customer__c
- HandsMen_Order__c
- HandsMen_Product__c
- Inventory__c
- Marketing_Campaign__c

2. Defined relationships using Lookup and Master-Detail fields.

Data Quality & UI Validation

- Used validation rules and required fields to maintain integrity from the UI.
- Implemented formula fields like `FullName__c` and `Stock_Status__c`.

Lightning App Builder

- Created a custom Lightning App with tabs for each object.
- Customized record pages for improved user experience.

Screen Flows

- Built guided flows for data entry, especially for creating new customers and orders.
- Example: Flow for capturing new order details linked to customer records.

Record-Triggered Flows

- Sent Order Confirmation emails automatically when new orders are placed.
- Triggered Loyalty Status updates based on `Total_Purchases__c`.

Scheduled Flow: Loyalty Status Update

- Runs daily at midnight.
- Updates loyalty status to:

  Gold if Total_Purchases__c > 1000

  Silver if 500–1000

  Bronze if <500

Email Templates & Alerts

- Created:

  Order Confirmation Email

  Low Stock Alert

  Loyalty Program Email

- Used HTML templates with Classic Letterheads.
- Set up email alerts for order confirmation and inventory low-stock notifications.

Inventory Monitoring

- Formula field Stock_Status__c on Inventory__c to show stock availability.
- Triggered email alert to Inventory Manager if Stock_Quantity__c < 5.

Asynchronous Apex

- Developed a Batch Apex job (`InventoryBatchJob`) scheduled using

  CopyEdit

  Apex

  System.schedule('Daily Inventory Sync', '0 0 0 * * ?', new InventoryBatchJob());

- Handles bulk updates of inventory and financials.

GitHub Deployment

- Pushed complete project metadata and screenshots to GitHub:

  GitHub Repository

- Includes project source files, README, and demo video link.

Skills & Tools Used:

Salesforce Platform

Screen & Record-Triggered Flows

Apex & Batch Apex

Data Modeling

Validation Rules & Formula Fields

Lightning App Builder

Git & GitHub

Email Templates & Alerts

Learning Outcomes:

Through this project, I learned to:

- Design and model real-world business scenarios using Salesforce.
- Automate repetitive tasks through flows and Apex.
- Use declarative tools to build scalable applications.
- Push and manage Salesforce metadata using GitHub.
- Communicate with customers via automated emails.

Attachments:

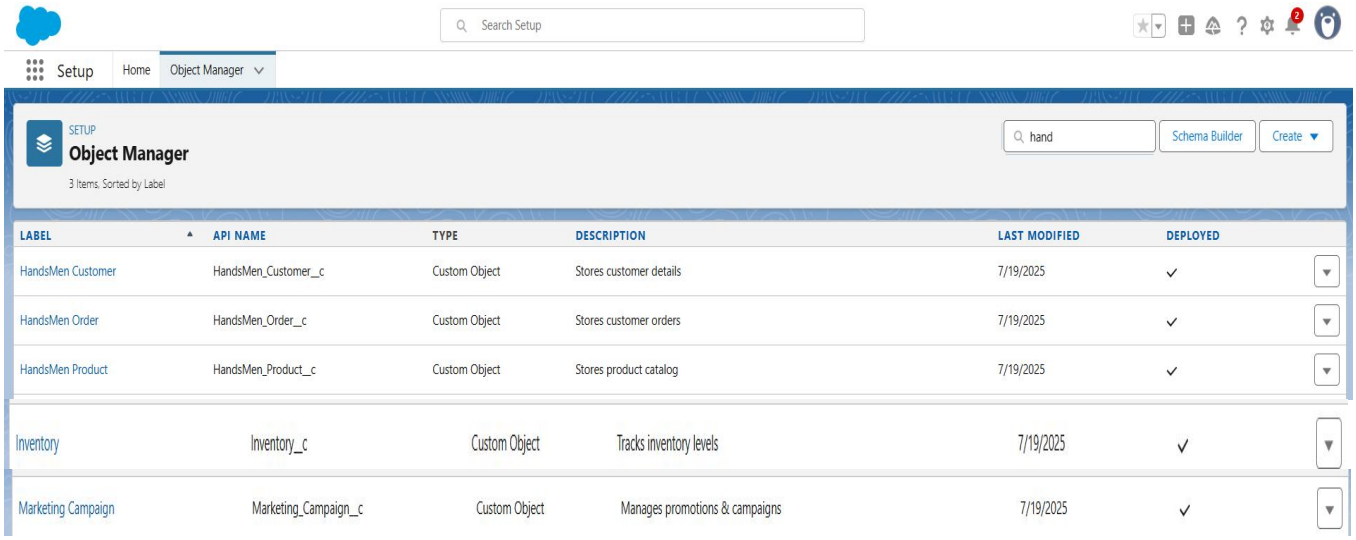
Screenshots: Showcasing objects, flows, templates, records.

Demo Video: Uploaded on SkillWallet platform.

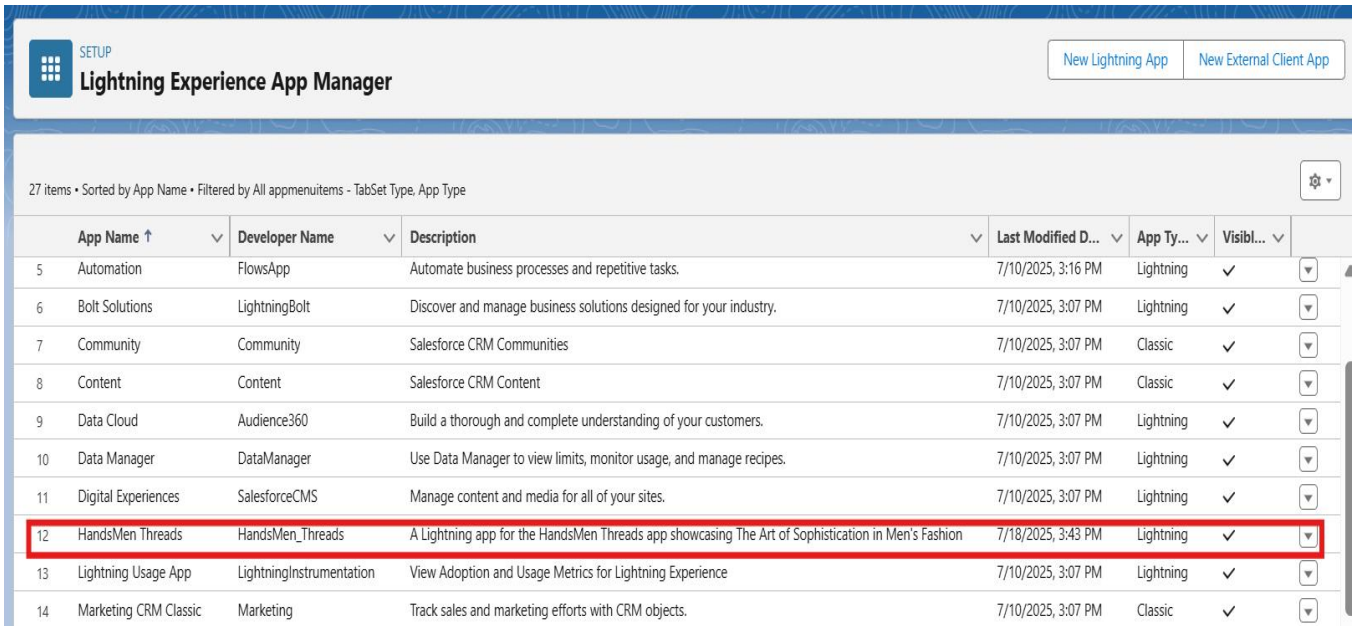
GitHub: Contains the metadata, flows, code, and assets.

ScreenShots:

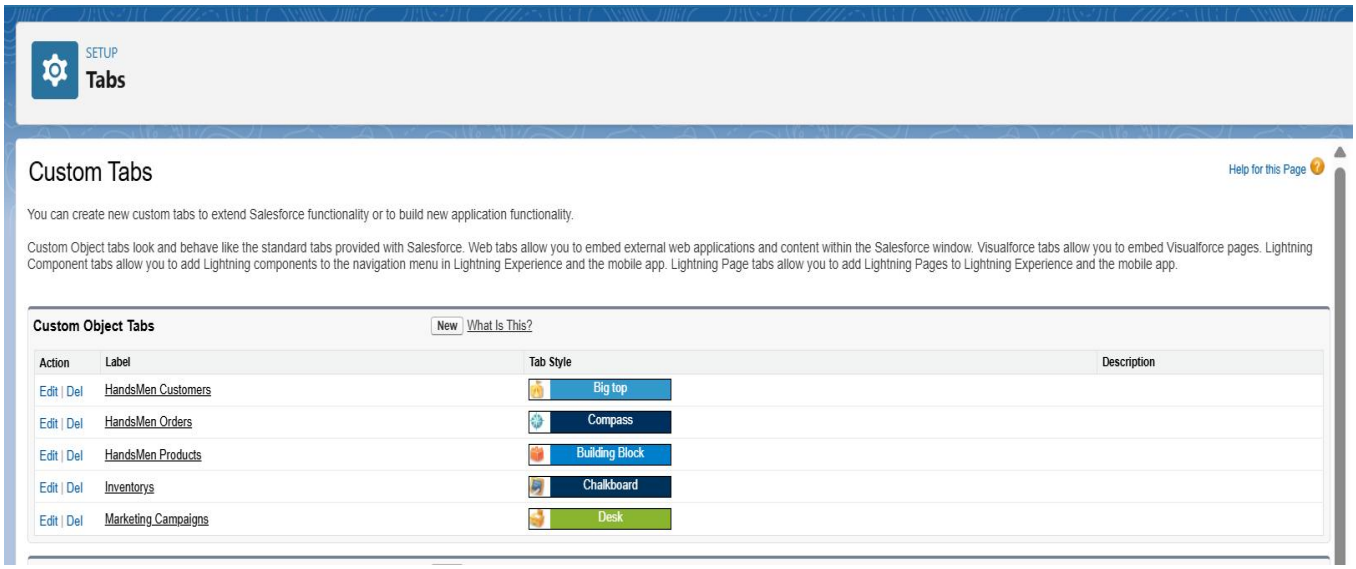
Custom Objects:



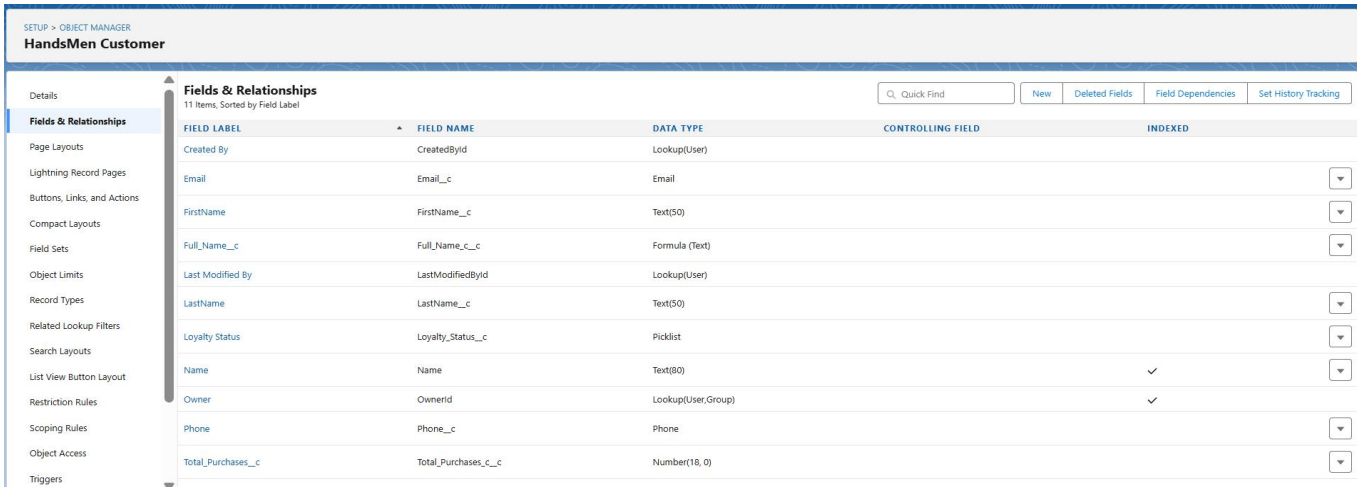
HandsMen Threads App:



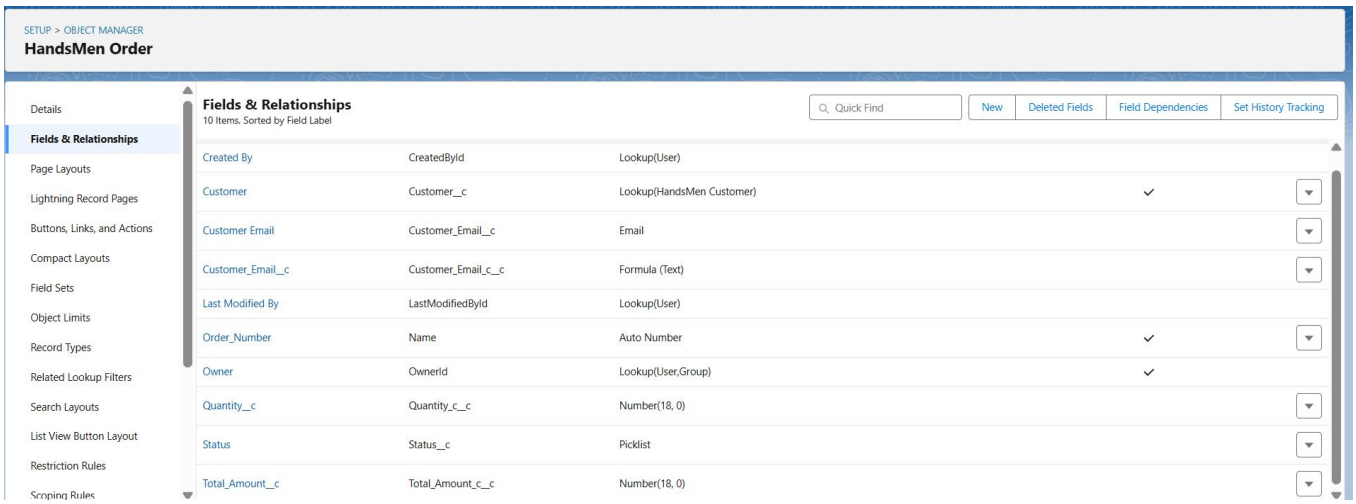
Custom Tabs:



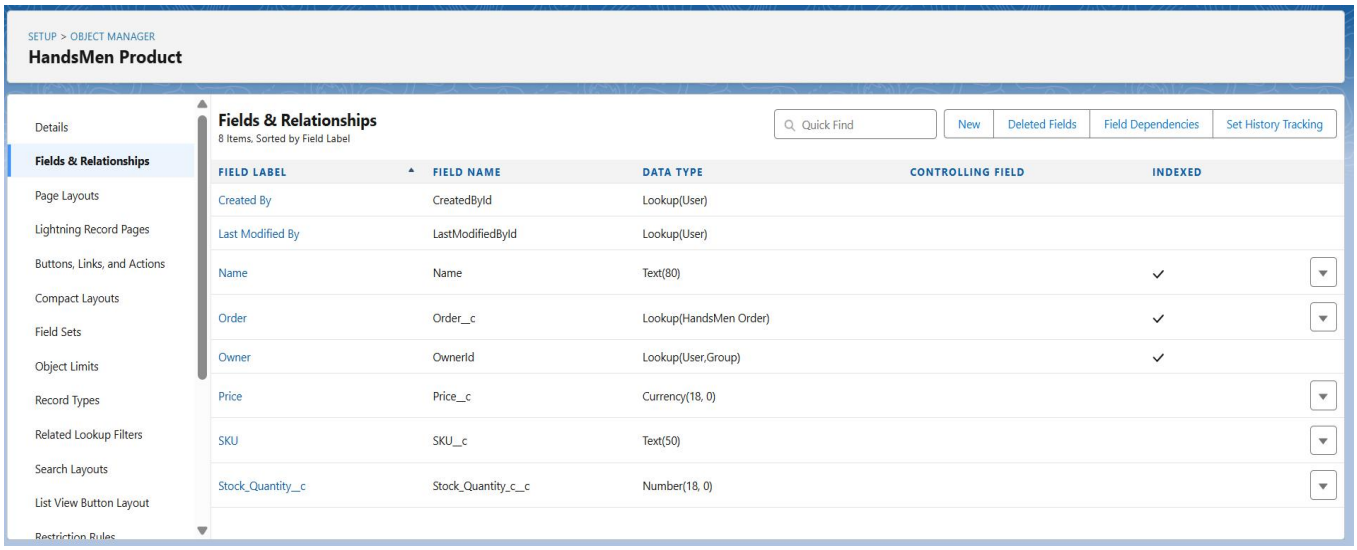
HandsMen Customer Fields:



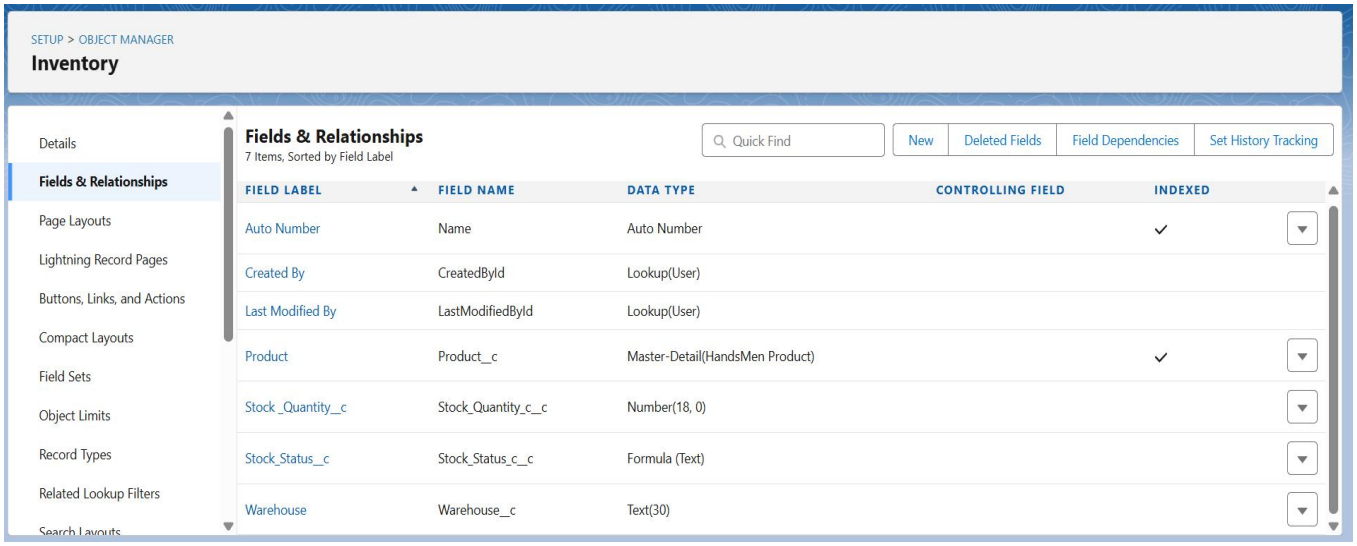
HandsMen Order Fields:



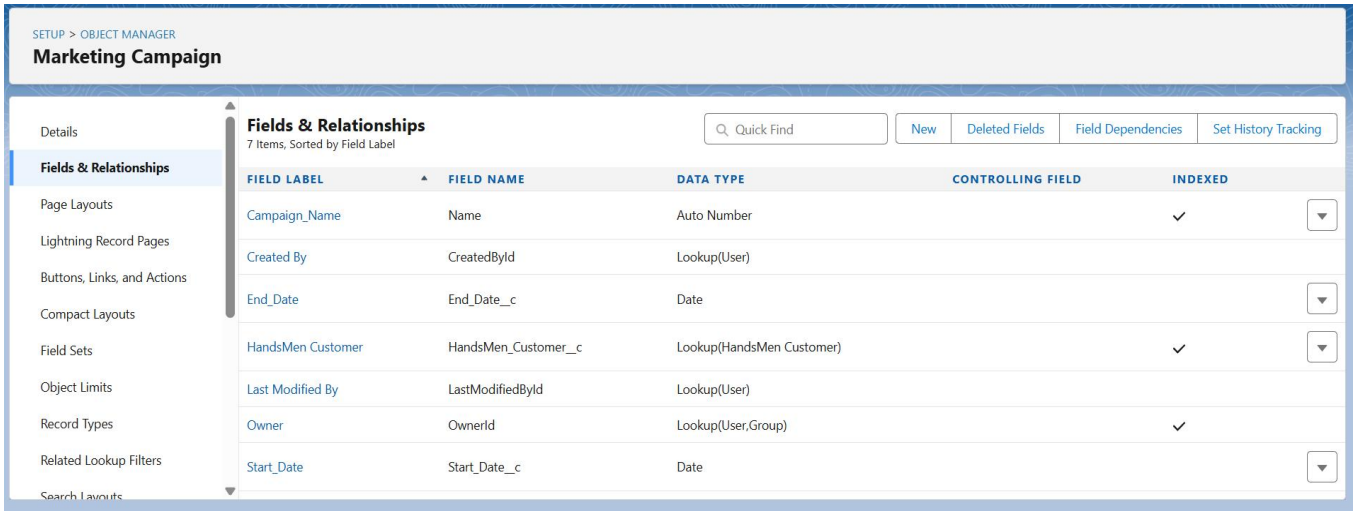
HandsMen Product Fields:



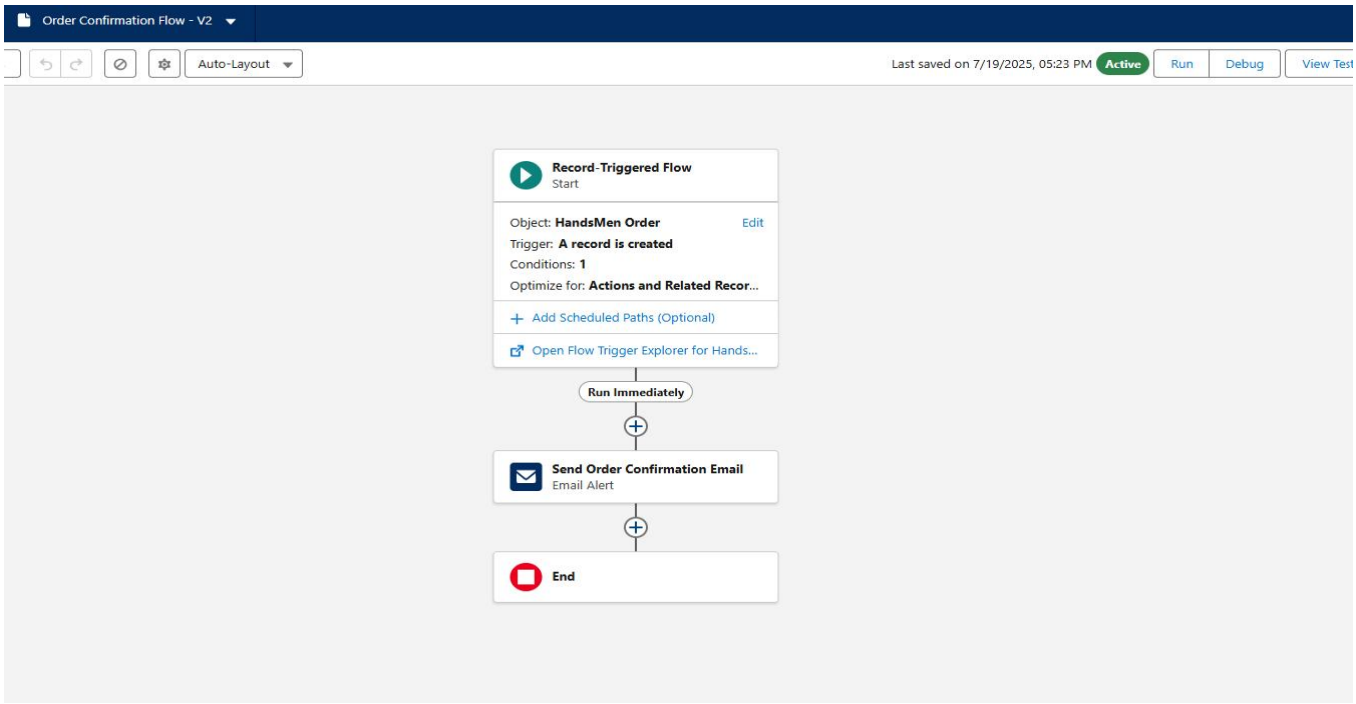
Inventory Fields:



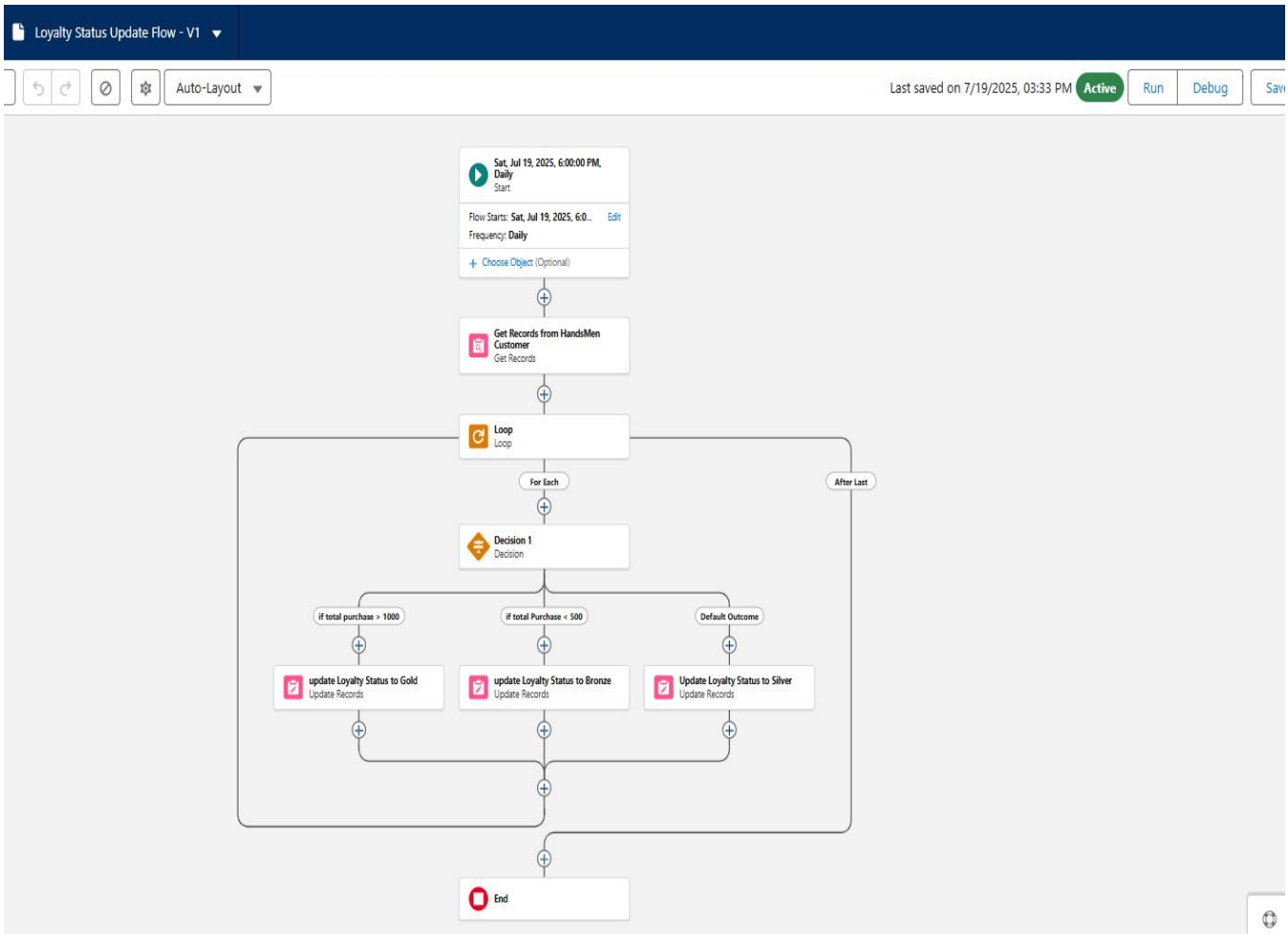
Marketing Campaign Fields:



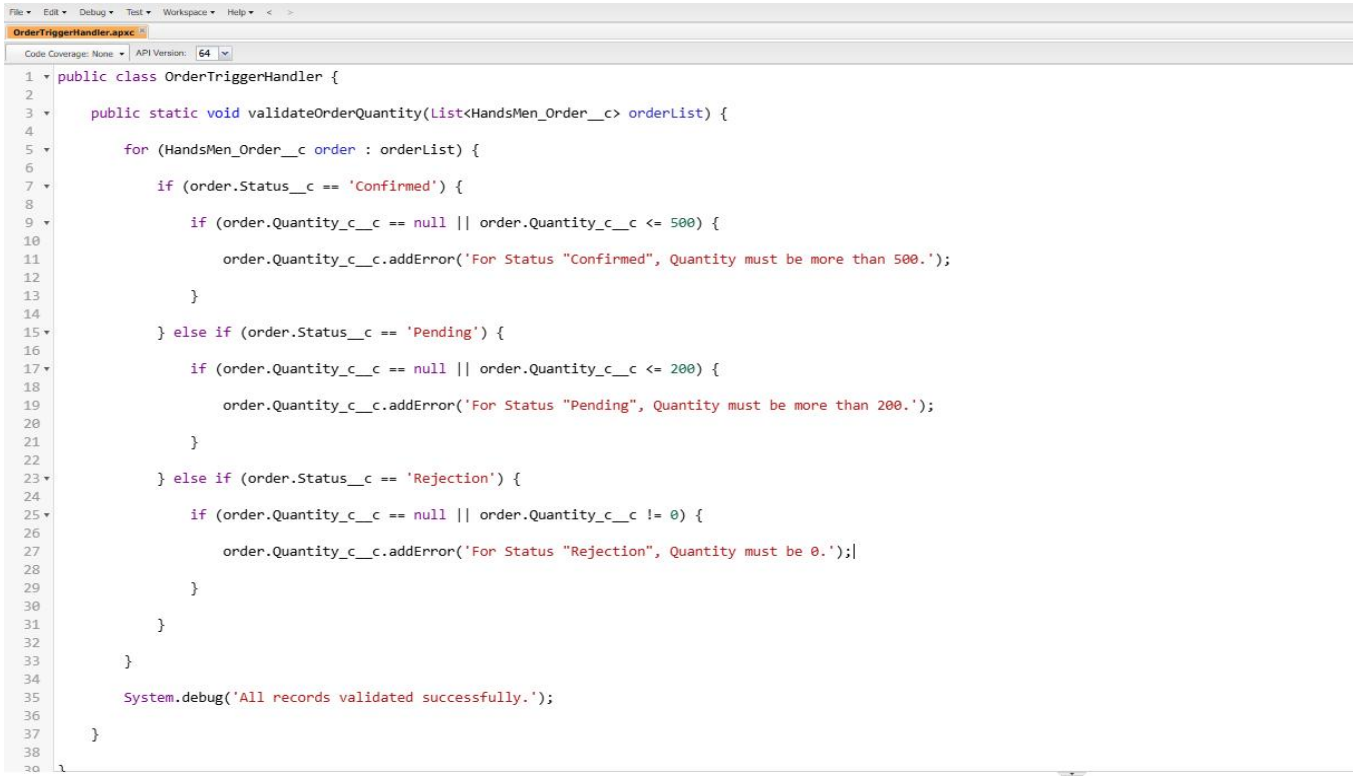
Order Confirmation Flow:



Loyalty Status Update Flow:

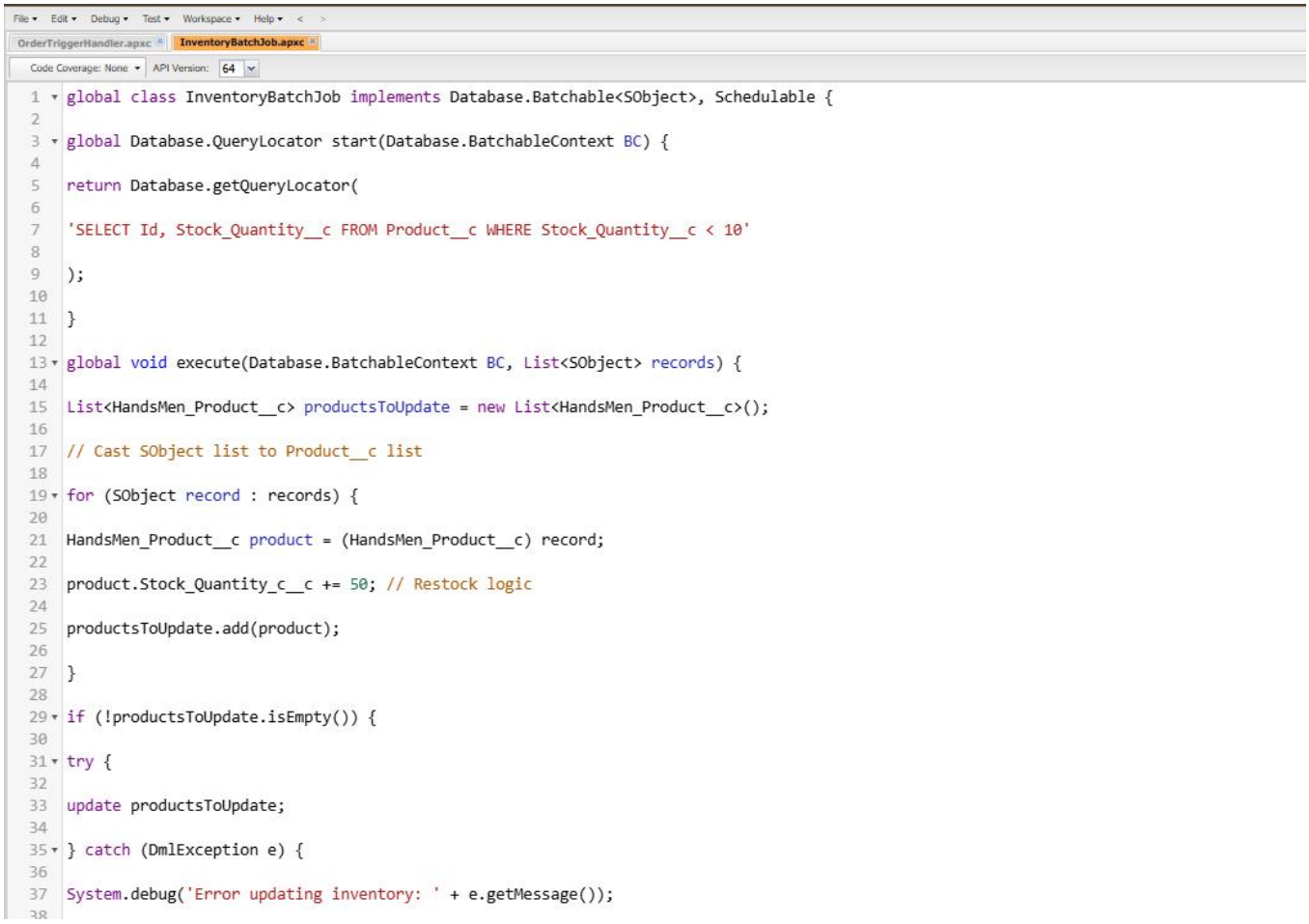


Order Trigger Handler Apex Class:



```
public class OrderTriggerHandler {

    public static void validateOrderQuantity(List<HandsMen_Order__c> orderList) {

        for (HandsMen_Order__c order : orderList) {

            if (order.Status__c == 'Confirmed') {

                if (order.Quantity_c__c == null || order.Quantity_c__c <= 500) {

                    order.Quantity_c__c.addError('For Status "Confirmed", Quantity must be more than 500.');

                }

            } else if (order.Status__c == 'Pending') {

                if (order.Quantity_c__c == null || order.Quantity_c__c <= 200) {

                    order.Quantity_c__c.addError('For Status "Pending", Quantity must be more than 200.');

                }

            } else if (order.Status__c == 'Rejection') {

                if (order.Quantity_c__c == null || order.Quantity_c__c != 0) {

                    order.Quantity_c__c.addError('For Status "Rejection", Quantity must be 0.');

                }

            }

        }

        System.debug('All records validated successfully.');

    }

}
```

Inventory Batch Job Apex Class:

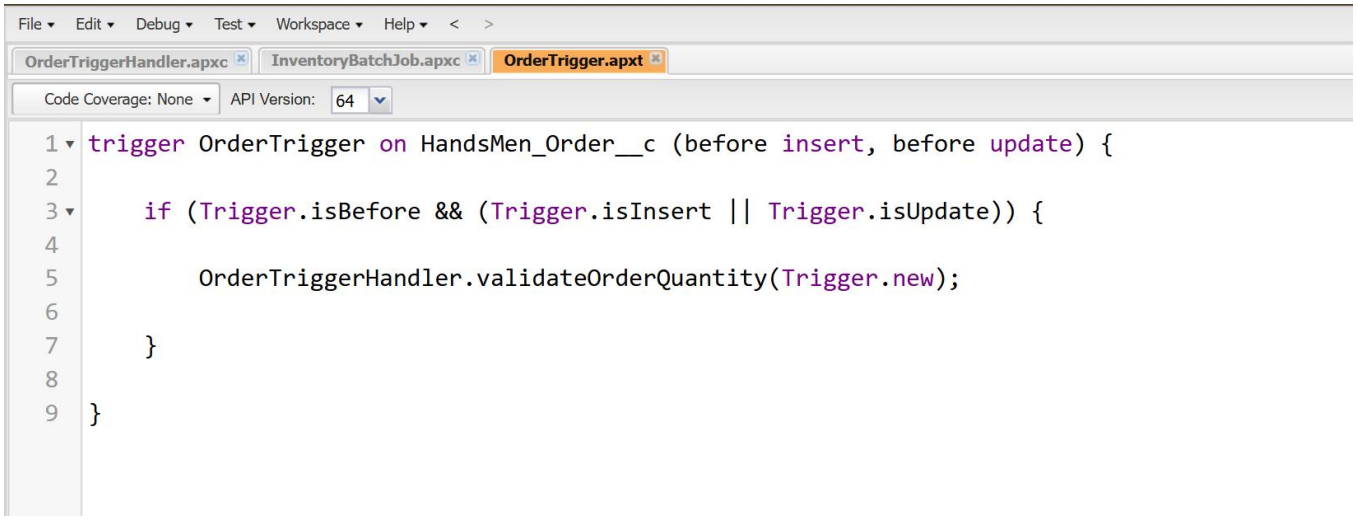


```
global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {

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__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);
}
}
```

Logs | Tests | Checkpoints | Query Editor | View State | Progress | Problems

Order Trigger:



Conclusion:

This project gave me valuable insight into end-to-end Salesforce development—from UI-driven automation to backend data processing using Apex. It reflects how Salesforce can be tailored for industry-specific use cases like fashion retail and inventory management.

I'm grateful for the opportunity provided by SmartBridge and Salesforce and proud to have implemented a complete working solution.