HandsMen Threads: Salesforce CRM Implementation Documentation

Project Overview

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

Objective: Implement a comprehensive Salesforce CRM solution to revolutionize data management, enhance customer relations, and streamline business operations for a premium men's fashion brand.

Duration: 4 Phases (Architecture → Development → Testing → Deployment)

1. Business Requirements

Core Functionalities

- **Customer Management:** Track customer information, purchase history, and loyalty status
- **Product Catalog:** Manage inventory, pricing, and stock levels
- **Order Processing:** Handle order lifecycle from creation to completion
- **Automated Communications:** Send order confirmations and stock alerts
- **Loyalty Program:** Dynamic customer status updates based on purchase behavior

Key Business Processes

1. **Automated Order Confirmations:** Email notifications post-order confirmation
2. **Dynamic Loyalty Program:** Status updates based on purchase history
3. **Proactive Stock Alerts:** Automatic notifications when inventory drops below 5 units
4. **Scheduled Bulk Updates:** Daily midnight processing for orders and inventory

2. Technical Architecture

System Requirements

- **Browsers:** Chrome, Firefox, Edge, Safari (latest versions)
- **Hardware:** Minimum 4GB RAM, Intel Core i3 processor
- **Network:** Stable broadband connection (30+ Mbps)
- **Storage:** 10GB free disk space

Data Model Design

Custom Objects Structure

Object	Purpose	Key Fields
HandsMen_Customer__c	Customer information	Name, Email, Phone, Loyalty_Status__c, Total_Purchases__c
HandsMen_Product__c	Product catalog	Name, SKU, Price, Stock_Quantity__c
HandsMen_Order__c	Order management	Order_Number, Status, Quantity__c, Total_Amount__c
Inventory__c	Inventory tracking	Auto Number, Warehouse, Stock_Quantity__c
Marketing_Campaign__c	Campaign management	Campaign_Name, Start_Date, End_Date

Relationships

- **Customer to Order:** One-to-Many (Lookup relationship)
- **Product to Order:** One-to-Many (Lookup relationship)
- **Product to Inventory:** One-to-One (Master-Detail relationship)

3. Implementation Details

Phase 1: Foundation Setup

A. Custom Objects Creation

```
Objects Created:
- HandsMen_Customer__c
- HandsMen_Product__c
- HandsMen_Order__c
- Inventory__c
- Marketing_Campaign__c
```

B. Field Configuration

- **Email Field:** Email type with validation
- **Phone Field:** Phone type
- **Loyalty Status:** Picklist (Bronze, Silver, Gold)
- **Formula Fields:** For calculated values
- **Lookup Relationships:** Between related objects

C. Data Quality Controls

Validation Rules:

1. **Email Validation (Customer):**
   - Rule: `NOT CONTAINS(Email, "@gmail.com")`
   - Message: "Please fill Correct Gmail"
2. **Stock Quantity Validation (Inventory):**

    - Rule: `Stock_Quantity__c <= 0`
    - Message: "The inventory count is never less than zero"

Phase 2: User Management & Security

A. Profile Setup

- **Profile Name:** Platform 1 (cloned from Standard User)
- **Object Permissions:** Read, Create, Edit, Delete access for custom objects

B. Role Hierarchy

```
CEO
├── Sales Manager
├── Inventory Manager
└── Marketing Manager
```

C. User Creation

- **Sales User:** Niklaus Mikaelson
- **Inventory User:** Kol Mikaelson
- **Marketing User:** [Additional user as needed]

D. Permission Sets

- **Permission_Platform_1:** Additional permissions for custom objects
- **Assignment:** Assigned to Platform 1 profile users

Phase 3: Process Automation

A. Email Templates

1. **Order Confirmation Email**

    html

    ```html
    <p>Dear {!Order__c.Customer__c},</p>
    <p>Your order #{!Order__c.Name} has been confirmed!</p>
    <p>Thank you for shopping with us.</p>
    <p>Best Regards,</p>
    <p>Sales Team</p>
    ```

2. **Low Stock Alert Template**
3. **Loyalty Program Email Template**

B. Record-Triggered Flows

1. Order Confirmation Flow

- **Trigger:** Order status changes to "Confirmed"
- **Action:** Send confirmation email to customer
- **Components:** Email Alert action

2. Stock Alert Flow

- **Trigger:** Inventory stock quantity < 5
- **Action:** Send alert to inventory manager
- **Frequency:** Every time condition is met

C. Scheduled Flow

Loyalty Status Update Flow

- **Schedule:** Daily execution
- **Logic:**
  - If Total_Purchases > 1000 → Gold Status
  - If Total_Purchases < 500 → Bronze Status
  - Else → Silver Status

Phase 4: Advanced Development

A. Apex Classes

1. OrderTriggerHandler Class

apex

```apex
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 == null || order.Quantity__c <= 500) {
                    order.Quantity__c.addError('For Status "Confirmed", Quantity must be more than 500.');
                }
            } else if (order.Status__c == 'Pending') {
                if (order.Quantity__c == null || order.Quantity__c <= 200) {
                    order.Quantity__c.addError('For Status "Pending", Quantity must be more than 200.');
                }
            } else if (order.Status__c == 'Rejection') {
                if (order.Quantity__c == null || order.Quantity__c != 0) {
                    order.Quantity__c.addError('For Status "Rejection", Quantity must be 0.');
                }
            }
        }
        System.debug('All records validated successfully.');
    }
}
```

2. InventoryBatchJob Class

apex

```apex
global class InventoryBatchJob implements Database.Batchable<SObject>, Schedulable {
    global Database.QueryLocator start(Database.BatchableContext BC) {
        return Database.getQueryLocator(
            'SELECT Id, Stock_Quantity__c FROM HandsMen_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>();

        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');
    }

    global void execute(SchedulableContext SC) {
        InventoryBatchJob batchJob = new InventoryBatchJob();
        Database.executeBatch(batchJob, 200);
    }
}
```

B. Triggers

OrderTrigger

```apex
trigger OrderTrigger on HandsMen_Order__c (before insert, before update) {
    if (Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
        OrderTriggerHandler.validateOrderQuantity(Trigger.new);
    }
}
```

C. Scheduled Jobs

- **Daily Inventory Sync:** Scheduled using cron expression `0 0 0 * * ?`
- **Execution:** Midnight daily processing for bulk operations

4. Application Configuration

Lightning App: HandsMen Threads

- **Navigation Items:** All custom objects, Reports, Dashboards
- **User Access:** System Administrator profile
- **Branding:** Custom styling and colors

Custom Tabs

- Individual tabs created for each custom object
- Added to Lightning App navigation

5. Testing Strategy

Unit Testing

- Apex class testing with various scenarios
- Trigger testing for all DML operations
- Validation rule testing

Integration Testing

- End-to-end flow testing
- Email template and alert testing
- Batch job execution testing

Data Testing

- Sample data creation and validation
- Performance testing with bulk records

6. Key Learning Outcomes

Technical Skills Developed

1. **Data Modeling:** Custom objects, relationships, field types
2. **Data Quality:** Validation rules, required fields, data integrity
3. **Lightning App Builder:** Custom app creation and navigation
4. **Process Automation:** Record-triggered flows, scheduled flows
5. **Apex Development:** Classes, triggers, batch processing
6. **Asynchronous Processing:** Batch jobs, scheduled execution

Business Process Understanding

- Customer relationship management
- Inventory management systems
- Order processing workflows
- Email marketing automation
- Loyalty program implementation

7. Deployment Checklist

Pre-Deployment

- [ ] All custom objects created and configured
- [ ] Validation rules tested and working
- [ ] Flows activated and tested
- [ ] Apex classes deployed and tested
- [ ] Email templates configured
- [ ] User permissions assigned

Post-Deployment

- [ ] User training completed
- [ ] Data migration (if applicable)
- [ ] Go-live monitoring
- [ ] Performance optimization
- [ ] User feedback collection

8. Maintenance and Support

Regular Tasks

- Monitor batch job execution

- Review email delivery reports
- Update loyalty program criteria
- Perform data quality checks
- User access review

Troubleshooting

- Debug logs monitoring
- Error handling in Apex code
- Flow troubleshooting
- Data validation issues

9. Future Enhancements

Potential Improvements

- Integration with external payment systems
- Advanced reporting and analytics
- Mobile app development
- AI-powered recommendations
- Advanced inventory forecasting

Scalability Considerations

- Governor limits monitoring
- Performance optimization
- Data archiving strategies
- Integration capabilities

10. Conclusion

The HandsMen Threads Salesforce implementation successfully demonstrates a comprehensive CRM solution that addresses key business requirements while showcasing advanced Salesforce development skills. The project covers the complete spectrum from basic configuration to advanced Apex development, providing a solid foundation for enterprise-level CRM operations.

Key Success Metrics:

- Automated order processing
- Real-time inventory management
- Customer loyalty program automation

- Efficient staff role management
- Comprehensive data quality controls

This implementation serves as a practical example of how Salesforce can transform business operations in the fashion retail industry while maintaining data integrity and enhancing customer experience.