SALESFORCE-SUPPORTED VIRTUAL INTERNSHIP PROGRAM 2025

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

NAME:- Kopparapu Krishna Bala Sai

BRANCH: B.TECH CSE

YEAR:4 TH YEAR

COLLEGE NAME: GITAM UNIVERSITY, HYDERABAD

PERSONAL MAIL:kkoppara@gitam.in

INTERN: SALESFORCE DEVELOPER WITH AGENTBLAZER

TRAILBLAZER:https://www.salesforce.com/trailblazer/jiam78y29a4vr6m9gf

Date:29/07/2025

Project Overview: HandsMen Threads, a dynamic organization in the fashion industry, is embarking on a Salesforce project designed to revolutionize their data management and enhance customer relations. The project involves building a robust data model tailored to store all pertinent business data, ensuring a seamless flow of information across the organization.

A key aspect of this project is the maintenance of data integrity directly from the user interface (UI). This feature will safeguard the accuracy and consistency of the data, which is crucial for informed decision-making and reliable business operations.The project will integrate several new processes into the business workflow to improve customer service and operational efficiency:

- Automated Order Confirmations: Post-order confirmation, customers will receive an email update, fostering engagement and strengthening customer relations.
- Dynamic Loyalty Program: Customer loyalty statuses will be updated based on purchase history, enabling personalized rewards and promoting repeat business.
- Proactive Stock Alerts: When stock levels drop below five units, automatic emails will notify the warehouse team, ensuring timely restocking and preventing stockouts.
- Scheduled Bulk Order Updates: Daily at midnight, the system will process bulk orders, updating financial records and adjusting inventory, ensuring accurate stock levels for daily operations.

Objectives: The main objectives for developing the HandsMen Threads: Elevating the Art of Sophistication in Men's Fashion. Build a CRM to manage tailored orders, customer profiles, and inventory. Enable automated dealer or stylist assignment based on location or service request. Implement order validation (e.g., available fabric or designer slot check). Integrate email reminders for delivery, fittings, and offers. Deliver a responsive interface for both in-house stylists and customers.

Phase 1: Architecture & Planning

This foundational phase focused on designing the structure of the Salesforce solution tailored to *HandsMen Threads*.

- Object & Field Design:
  Identified key entities like Tailored_Order__c, Customer_Profile__c, Stylist__c, and Fabric__c. Custom fields such as Measurement__c, Order_Status__c, and Preferred_Style__c were added to support business requirements.
- Relationships & Formula Fields:
  Established lookup and master-detail relationships between customer orders, stylists, and fabric records. Formula fields were used to calculate estimated delivery dates and total pricing based on measurements and style complexity.
- Validation Rules & Automation Planning:
  Created rules to ensure mandatory fields (like measurement inputs or stylist availability) are filled before submission. Planned automation processes including:
  - Auto-assignment of stylists based on availability
  - Inventory checks before confirming orders
- Apex Triggers & Batch Jobs Design:
  Designed logic to automate inventory deduction using Apex triggers and scheduled inventory reconciliation using batch Apex jobs.
- Email Template Planning:
  Mapped out notification use cases, including:
  - Order confirmations
  - Test fitting reminders
  - Promotional events

Phase 2: Development

Hands-on development was carried out in a Salesforce Developer Org and tested in a sandbox environment.

- Custom Object & Field Implementation:
  Created all required custom objects and fields based on the planning phase. Configured picklists for options like style types, status, and preferred tailoring time.
- Automation Implementation:
  - Developed Record-Triggered Flows for auto-dealer/stylist assignments and test fitting reminders.
  - Implemented Apex Triggers to prevent orders if fabric is out of stock and reduce stock on confirmation.
  - Configured Batch Apex Jobs for nightly stock synchronization.
- Data Security Setup:
  Defined Profiles and Permission Sets for Admins, Stylists, and Assistants.
  Set up Role Hierarchy and Sharing Rules to ensure stylists only view their assigned orders.

- Email Template Configuration:
  Designed and tested HTML-based email templates for order communication, branded to HandsMen Threads' aesthetic.

Phase 3: Testing & Quality Assurance (QA)

- Unit Testing:
  Validated individual components such as Apex methods, flows, and validation rules using mock data.
- Integration Testing:
  Simulated complete order lifecycles to ensure that automation, inventory updates, and email communications worked in sync.
- Performance & Security Testing:
    - Assessed flow performance with large datasets.
    - Verified role-based access and object-level security.
    - Ensured field-level restrictions for sensitive data like customer measurements and contact info.

Phase 4: Deployment & Training

- Production Deployment:
  Deployed components to the production environment using Change Sets. Verified all automation and object-level settings post-deployment.
- User Training & Documentation:
  Delivered a walkthrough session for stylists and back-office staff covering:
    - Order creation
    - Fitting schedule management
    - Report usage

Shared documentation on order workflows, role responsibilities, and system limitations.

- Post-Go-Live Support:
  Monitored user activity, collected feedback, and resolved post-deployment bugs.
  Scheduled regular checks of scheduled jobs and stock logs using Apex Logs and Flow Error emails.

Data Management – Objects

The following are the Custom objects that we need to create

- HandsMen Customer
- HandsMen Order
- HandsMen Product
- Inventory
- Marketing Campaign

To navigate to the Setup page in Salesforce, click on the gear icon at the top right corner of the screen and select Setup from the dropdown menu.

In Activity 1, we begin by creating a custom object named HandsMen Customer. From the Setup page, navigate to Object Manager, click on Create, and select Custom Object. Enter the label name as HandsMen Customer and the plural label name also as HandsMen Customer. For the record name, enter HandsMen Customer Name and choose the data type as Text. Make sure to check the boxes for Allow Reports and Allow Search. Finally, click Save to create the object.

Next, we create the HandsMen Product object. Again, go to Object Manager, click Create, and choose Custom Object. Enter HandsMen Product as the label name and HandsMen Products as the plural label name. Set the record name to HandsMen Product Name and choose the data type as Text. Enable Allow Reports and Allow Search, and click Save.

Following that, we create the HandsMen Order object. From the Setup page, go to Object Manager, click on Create, and select Custom Object. Enter the label name as HandsMen Order and the plural label name as HandsMen Orders. Set the record name to HandsMen OrderNumber, choose Auto Number as the data type, and use the display format O-{0000} with a starting number of 001. Make sure to check both Allow Reports and Allow Search, then click Save.

Then, we proceed to create the Inventory object. From Object Manager, click on Create, then Custom Object. Set the label name to Inventory and the plural label name to Inventorys. Use Inventory Number as the record name, select Auto Number as the data type, set the display format to I-{0000}, and start from 001. Enable Allow Reports and Allow Search, then save the object.

Lastly, we create the Marketing Campaign object. Navigate to Object Manager, click on Create, and select Custom Object. Enter Marketing Campaign as the label name and Marketing Campaigns as the plural label name. Set the record name to Marketing Campaign Number, choose Auto Number as the data type, and set the display format to MC-{0000} with a starting number of 001. Check Allow Reports and Allow Search, and save the object.

Data Management – Tabs

To create a custom tab for the HandsMen Customer object, first go to the Setup page in Salesforce. In the Quick Find bar, type "Tabs" and click on the Tabs option that appears. Under the Custom Object Tabs section, click on the New button. Select the object HandsMen Customer from the dropdown, then choose any tab style of your preference. Click Next to proceed to the "Add to profiles" page and leave it as the default selection. Click Next again on the "Add to Custom Apps" page, keeping the default settings, and then click Save to finish creating the tab.

Repeat the same steps to create custom tabs for the other objects that were created earlier, including HandsMen Product, HandsMen Order, Inventory, and Marketing Campaign. This ensures that all custom objects are easily accessible from the Salesforce app navigation.

Data Management - App Manager

To create a Lightning App page in Salesforce, go to the Setup page and search for “App Manager” in the Quick Find bar. Select “App Manager” from the results and click on the “New Lightning App” button. In the App Details and Branding section, enter the App Name as HandsMen Threads. The Developer Name field will be auto-populated. In the Description field, provide a meaningful description of the app. Adding an image is optional and not mandatory. Leave the Primary Color Hex Value as the default value. Click Next to proceed to the App Options page and keep all the options as default. Click Next again to go to the Utility Items section, and keep that section as default too. Click Next to continue.

In the Navigation Items step, search and select the required items from the search bar. These items include HandsMen Customer, HandsMen Order, Inventory, HandsMen Product, Reports, Dashboard, Account, Contact, and Marketing Campaign. Select each item and move it to the selected list using the arrow button. These items include both standard and custom objects created in the previous activities. After selecting all necessary items, click Next.

In the User Profiles section, search for the profile named System Administrator in the search bar. Once found, move it to the selected list using the arrow button. After completing all steps, click Save and Finish to create the Lightning App successfully.

Data Management - Fields

To create fields in the HandsMen Customer object, go to the Setup page, click on Object Manager, and type the object name HandsMen Customer in the Quick Find bar. Click on the object name, then go to Fields & Relationships and click on New. Select the data type as Email and click Next. In the field creation screen, set the Field Label as Email. The Field Name will be auto-generated. Click Next, then Next again, and finally click Save & New to continue creating more fields.

To create a Phone field in the HandsMen Customer object, go to Object Manager, search for HandsMen Customer, and click on it. Under Fields & Relationships, click on New. Select the data type as Phone and click Next. On the next screen, enter the Field Label as Phone. The Field Name will auto-populate. Click Next twice and then Save & New.

To create a Picklist field in the HandsMen Customer object, go to Object Manager, find and open the HandsMen Customer object. Click on Fields & Relationships and then click on New. Choose the data type as Picklist and click Next. Set the Field Label as Loyalty Status. Under the values section, choose “Enter values, with each value separated by a new line” and enter the following values: Gold, Silver, Bronze. Then click Next, click Next again, and finally click Save & New.

In **Unit 1**, to create a Lookup Relationship between the Marketing Campaign and HandsMen Customer objects, go to the Setup page, click on Object Manager, and search for Marketing Campaign. Open the object and go to Fields & Relationships, then click on New. Select the data type as Lookup Relationship and click Next. For the related object, select HandsMen Customer and click Next. Set the Field Label as HandsMen Customer, then click Next, Next again, and finally click Save.

In **Unit 2**, to create a Lookup Relationship between the HandsMen Product and HandsMen Order objects, go to Setup, open Object Manager, and search for HandsMen Product. Click on the object, then go to Fields & Relationships and click on New. Choose Lookup Relationship as the data type and click Next. Set the related object to HandsMen Order and click Next. Enter the Field Label as Order, click Next, proceed with the defaults, and then click Save.

In **Unit 3**, to create a Lookup Relationship between HandsMen Order and HandsMen Customer, go to Setup, open Object Manager, and search for HandsMen Order. Click on the object, then go to Fields & Relationships and click New. Select Lookup Relationship and click Next. Set the related object to HandsMen Customer, click Next, then enter the Field Label as Customer. Continue by clicking Next, Next, and then Save.

In **Unit 4**, to create a Master-Detail Relationship between Inventory and HandsMen Product, go to Setup, open Object Manager, and search for Inventory. Click on the object, go to Fields & Relationships, and click New. Choose Master-Detail Relationship as the data type and click Next. Set the related object to HandsMen Product and click Next. Enter the Field Label as Product, then click Next, proceed through the steps, and click Save.

To create the Stock Status formula field in the Inventory object, go to Object Manager, search for Inventory, and click on Fields & Relationships, then click on New. Choose Formula as the data type and click Next. Enter the Field Label and Field Name as Stock_Status__c, set the formula return type as Text, and click Next. In the

Advanced Formula section, enter the formula: IF(Stock_Quantity__c > 10, "Available", "Low Stock"). Click Check Syntax to verify the formula, then click Next, Next, and Save.

To create the Full Name formula field in the HandsMen Customer object, first create two custom fields named FirstName and LastName. After creating them, go to Fields & Relationships and click on New. Choose Formula as the data type and click Next. Set the Field Label and Field Name as Full_Name__c and choose Text as the formula return type. In the Advanced Formula section, enter the formula: FirstName__c + " " + LastName__c. Click Check Syntax, then click Next twice, and finally click Save & New.

Below is a description of the objects and key fields used in the system:

The HandsMen Customer object is a custom object that stores customer details. Key fields include the record name (HandsMen Customer Name), Email, Phone, Loyalty_Status__c (a picklist with values Bronze, Gold, and Silver), and Total_Purchases__c (Number).

The HandsMen Product object is a custom object used to store the product catalog. Key fields include the record name (HandsMen Product Name), SKU (Text), Price (Currency), and Stock_Quantity__c (Number).

The HandsMen Order object is a custom object used to store customer orders. Key fields include the record name (Order Number), Status (a picklist with values Pending, Confirmed, and Rejection), Quantity__c (Number), and Total_Amount__c (Number).

The Inventory object is a custom object used to track inventory levels. The key fields include the auto-numbered record name, Warehouse (Text), and Stock_Quantity__c (Number).

The Marketing Campaign object is a custom object used to manage promotions and campaigns. Key fields include the record name (Campaign Name), Start_Date (Date), and End_Date (Date).

Data Configuration

Validation rules are used to ensure that the data entered into Salesforce meets specific business criteria. In this activity, we will create validation rules on multiple custom objects to enforce data quality and prevent incorrect input.

To create a validation rule for the Total_Amount__c field in the HandsMen Order object, first go to the Setup page, click on Object Manager, and type HandsMen Order__c in the Quick Find bar. Click on the object and then click on Validation Rules. Click the New button to create a new rule. Enter the rule name as Total Amount. In the Error Condition Formula field, enter the formula: Total_Amount__c <= 0. This rule ensures that the total amount cannot be zero or negative. Enter the error message as "Please Enter Correct Amount". For the error location, select Field and choose the Total Amount field. Then click Save to finish creating the rule.

Next, create a validation rule for the Inventory object. Go to Object Manager, search for Inventory__c, and click on it. Navigate to Validation Rules and click New. Enter the rule name as Stock Quantity. In the Error Condition Formula field, enter: Stock_Quantity__c <= 0. This rule ensures that the inventory stock cannot be less than or equal to zero. Enter the error message as "the inventory count is never less than zero." Set the error location to Top of Page and click Save.

Now create a validation rule for the HandsMen Customer object. Go to Object Manager, search for HandsMen Customer__c, and click on it. Under Validation Rules, click New. Enter the rule name as Email. In the Error Condition Formula field, enter: NOT CONTAINS(Email, "@gmail.com"). This rule ensures that the email entered contains "@gmail.com". Enter the error message as "Please fill Correct Gmail". Set the error location to Top of Page and click Save.

Before creating these validation rules, ensure that all the fields used in the formula such as Total_Amount__c, Stock_Quantity__c, and Email exist in their respective objects. If any of these fields are not yet created, make sure to create them before applying the validation rules.

Data security – Profiles

To create a new profile in Salesforce, go to the Setup page and type "Profiles" in the Quick Find box. Click on Profiles from the search results. Locate the Standard User profile and click Clone next to it. When prompted, enter the profile name as Platform 1 and click Save.

After the profile is created, you will be directed to the new profile's detail page. Click the Edit button to make changes. Scroll down to the Custom Object Permissions section and provide the necessary access permissions for the HandsMen Product and Inventory objects. Once the required permissions are granted, scroll further down and click Save to apply the changes.

Data Security – Roles

To create the Sales Manager role in Salesforce, go to the Setup page and type "Roles" in the Quick Find box. Click on Set Up Roles from the search results. On the roles hierarchy page, click on Expand All to view the existing role structure. Locate the CEO role and click on Add Role beneath it to create a new role reporting to the CEO. In the role creation page, enter the Label as Sales. The Role Name will be auto-populated based on the label. Verify that the role reports to the correct parent role, in this case, the CEO, and then click Save to create the role.

Following the same steps, create two additional roles named Inventory and Marketing. For each role, click Add Role under the CEO role, enter the respective label as Inventory or Marketing, confirm that the role is reporting to the correct parent role, and click Save. This ensures that all three roles—Sales, Inventory, and Marketing—are properly structured in the role hierarchy under the CEO.

Data Security - Users

To create a new user in Salesforce, go to the Setup page, type "Users" in the Quick Find box, and select Users from the results. Click on the New User button to begin the process. In the user creation form, enter the first name as Niklaus and the last name as Mikaelson. Provide an alias name and a personal email ID. The username should follow the format text@text.text. Enter a nickname of your choice. Set the role as Sales, choose the user license as Salesforce Platform, and select the profile as Platform 1. Once all the required fields are filled, click Save to create the user.

To create another user, repeat the same steps by going to Users in Setup and clicking New User. This time, enter the first name as Kol and the last name as Mikaelson. Provide an alias, a valid personal email ID, and set the username in the required format. Enter a nickname for the user. Assign the role as Inventory, select the user license as Salesforce Platform, and choose Platform 1 as the profile. After completing the form, click Save.

Similarly, create two more users as mentioned in Activity 2. Follow the same procedure, entering the respective user details, roles, and assigning the Salesforce Platform license with the Platform 1 profile. This completes the user creation task for all required roles.

Data Security - Permission Set

To create a permission set in Salesforce, go to the Setup page, type “Permission Sets” in the Quick Find box, and select Permission Sets from the results. Click on the New button to create a new permission set. Enter the label name as Permission_Platform_1 and click Save.

Once the permission set is created, scroll down to the Apps section and select Object Settings. In the list of objects, click on HandsMen Customer. Then click Edit and check the permissions for Read, Create, Edit, and Delete. After selecting the permissions, click Save.

Repeat the same steps to add object permissions for the HandsMen Order object. Go back to Object Settings, select HandsMen Order, click Edit, and again enable Read, Create, Edit, and Delete permissions. Click Save to confirm.

After saving the required permissions, click on Manage Assignments within the permission set. Then click on Add Assignments. From the list of users, select any user who has the Platform 1 profile, then click Next. Click Assign to apply the permission set to the selected user, and finally click Done to complete the process.

Email Template

To create a Classic Email Template in Salesforce, begin by navigating to the Setup area. Click on the gear icon located in the top-right corner of the page and select "Setup" from the dropdown menu. In the Quick Find box, type "Classic Email Templates" and select the corresponding option. Once there, click on "New Template." You will be given multiple format options such as Text, HTML (with Classic Letterhead), Custom (without Classic Letterhead), and Visualforce. For this activity, choose "HTML (with Classic Letterhead)" to create a professionally formatted email.

After selecting the format, proceed to fill in the template details. Choose the folder "Unfiled Public Email Templates" or create a new folder if needed. Make sure to check the box labeled "Available for Use" to activate the template. Enter "Order_Confirmation_Email" as the Email Template Name. Leave the encoding as UTF-8, which is the default setting. In the subject line, type "Your Order has been Confirmed!" For the HTML body, input the following content:

```
<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>
```

Once all the details have been filled in, click on "Save" to store the template.

Repeat the same steps to create two additional email templates. The first one should be named "Low Stock Alert" and the second one should be named "Loyalty Program Email." Fill in their respective content as mentioned in your template description.

After creating the templates, the next step is to set up an email alert that will send a notification when an order is confirmed. To do this, go to Setup, type "Email Alerts" in the Quick Find box, and select the "Email Alerts" option. Click on "New Email Alert." Enter the description as "Order Confirmation Email Alert." Choose "Order__c" as the object. Select the "Order_Confirmation_Email" template that you created earlier. For the recipient type, select "Related Record" and then choose "Customer__c" as the target recipient. Finally, click "Save" to complete the email alert configuration.

Flows

To create the Order Confirmation Email Flow, begin by navigating to Setup and typing "Flows" in the Quick Find box. Select "Flows" and then click on New Flow. Choose the Record-Triggered Flow option and click Create. Configure the trigger details by selecting the object as Order__c. Set the trigger condition to run When a record is updated, and define the criteria by specifying that the field Order__c.Status__c must equal "Confirmed". Ensure you select Only when a record is updated to meet the condition. Click Done to move to the next step.

Next, add an Action element by clicking the "+" icon and selecting Action. Choose Send Email Alert as the action type. For the Email Alert, select the previously created Order Confirmation Email Alert. Provide a label such as Send Order Confirmation Email, and set the Record ID to {!$Record.Id}. Click Save, then give your flow a name like Order Confirmation Flow, and activate it by clicking Activate.

To set up the Stock Alert Flow, again go to Setup → Flows → New Flow. Select Record-Triggered Flow and click Create. For this flow, select the object Inventory__c and trigger it to run When a record is created or updated. Define the condition where Stock_Quantity__c < 5, and choose the option to trigger the flow every time a record is updated and meets the condition requirements. Click Done.

Add an Action element by clicking the "+" icon and selecting Action. Choose Send Email Alert. If an alert does not yet exist for stock updates, create a new email alert similar to the previous process. The recipient should be set as the Inventory Manager. After completing the configuration, name the flow Stock Alert Flow, click Save, and then Activate it.

For the Loyalty Status Update Flow, navigate to Setup → Flows → New Flow. Select Schedule-Triggered Flow and click Create. Set the start date and time to your preferred daily time, and set the frequency as Daily. Click Done to proceed.

Add a Get Records element by clicking the "+" icon and selecting Get Records. Choose the object HandsMen_Customer__c, and set the filter conditions to retrieve all records without sorting. Click Done. Then add a Loop element to iterate over the collection from the Get Records step, choosing the direction from First to Last, and click Done.

Inside the loop, click the "+" icon and add a Decision element to evaluate the Total_Purchases__c field. If Total_Purchases__c > 1000, the customer's Loyalty_Status__c should be updated to Gold. Add an Update Records element for this outcome and configure it to specify conditions for updating the HandsMen_Customer__c object, setting Loyalty_Status__c = Gold.

Add another outcome to the Decision element for Total_Purchases__c < 500, and configure another Update Records action to set Loyalty_Status__c = Bronze. For the default outcome (between 500 and 1000), add an Update Records element to set the status to Silver.

After all outcomes are handled, click Done, then Save the flow with the name Loyalty Status Update Flow, and finally, Activate it.

Automation using Apex

To create an Apex Class named OrderTriggerHandler, follow these steps:

First, go to **Setup** in Salesforce. Click on the **gear icon** in the top-right corner and select **Developer Console**. Once the Developer Console opens in a new window, click on the **File** menu → select **New** → then choose **Apex Class**. In the popup, enter the class name as OrderTriggerHandler and click **OK**. This creates a new Apex class where you can write your custom logic.

Source Code:

```
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.');
  }
}
```

To create an Apex Trigger in Salesforce, begin by navigating to the Developer Console. Click on the gear icon at the top right of the Salesforce interface and select “Developer Console.” Once the Developer Console opens in a new window, go to the File menu, then choose “New” and click on “Apex Trigger.” A dialog box will appear asking for the trigger name and the sObject to associate it with. Enter the trigger name as “OrderTrigger” and from the dropdown list, select the custom object “HandsMen_Order__c,” which represents the Order object in your application. Click on the Submit button to proceed.

A new trigger editor will open where you can write the logic. In this case, the trigger will handle events after an order is inserted or updated. The logic will call specific methods from a previously created handler class named “OrderTriggerHandler.” This separation ensures that business logic is managed in a dedicated Apex class, keeping the trigger clean and maintainable. After writing the required code, which checks if the trigger is executing after an insert or update operation and then calls the respective handler methods, save the trigger by clicking on File and then Save. The Apex Trigger is now ready to respond to order creation and updates in your Salesforce org.

Source Code:

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

Save the code.(click on file → Save)

Batch Jobs

Create an 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 += 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);

}

}
```

Save the code.(click on file → Save)

Activity 2 :

To schedule an Apex class in Salesforce, start by opening the Developer Console. Go to Setup, then click on the gear icon in the top-right corner and select “Developer Console.” Once the Developer Console opens in a new window, click on the “Debug” menu, and choose “Open Execute Anonymous Window.” This window allows you to execute Apex code directly.

In the Execute Anonymous Window, enter the following line of code:

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

This command schedules the InventoryBatchJob class to run daily at midnight. The cron expression '0 0 0 * * ?' represents the time 12:00 AM every day.

After pasting the code, ensure the “Open Log” checkbox is selected if you want to verify execution details, then click the “Execute” button.

To confirm the job is scheduled, go back to the Setup page in Salesforce. In the Quick Find box, type “Jobs,” and then click on “Scheduled Jobs.” This page will display a list of all scheduled Apex jobs. You should see “Daily Inventory Sync” listed, indicating that your batch class has been successfully scheduled to run every day at midnight.

THANK YOU