RePlastic Inventory Manager

Name: Hanvith Reddy

B Batch: Salesforce Developer with Agentblazer

Demo Video Link: [link](https://youtu.be/rzOSLILrNb8)

Trailblazer: [trailhead](https://www.salesforce.com/trailblazer/p86ptvk3bc1l00qqbp)

Date:17/07/2025

# Summary

The RePlastic Inventory Manager is a Salesforce-based CRM developed for RePlastix Innovations, a recycling company focused on sustainable plastic waste management. This project automates the flow of inventory deduction, restock management, and stakeholder communication. It is built with Apex triggers, Lightning UI, flows, test classes, and robust security roles and profiles.

# Use Case Scenario

Replastix Innovations, a pioneering organization in the field of plastic and recycle waste management, has embarked on a transformative journey by integrating the Salesforce platform into its operations. This strategic initiative is aimed at automating and streamlining key business processes, including plastic waste recycling, order management, inventory tracking, and reporting. The organization has already established a robust data security framework within Salesforce by creating roles and profiles for different users, each with distinct levels of data accessibility. This ensures that sensitive information is protected while maintaining operational efficiency and facilitating seamless data flow across various departments.

Replastix Innovations has prioritized data security as a cornerstone of its Salesforce integration. By creating detailed roles and profiles, the organization has ensured that each user has access only to the data necessary for their specific responsibilities. This granular approach to data management not only enhances security but also improves the overall user experience by providing a tailored interface for each role. The structured data framework ensures that information is organized in a manner that supports efficient retrieval and analysis, thereby empowering decision-makers with timely and accurate insights.

One of the key areas where Replastix Innovations has focused its automation efforts is inventory management. The organization has developed a sophisticated system to monitor stock levels and trigger automated actions when predefined thresholds are breached. Specifically, whenever the stock level of a product falls below a set threshold, a new task record is automatically created and associated with the same product record. This task is then assigned to the record owner, ensuring that appropriate actions are taken promptly to address the low stock situation. This automation reduces the risk of stockouts and ensures that inventory levels are consistently maintained.

In addition to inventory management, Replastix Innovations has implemented automation processes to enhance order management and streamline the replenishment of low-stock items. If the product stock is low, the system automatically updates the order object to reflect the current status. Simultaneously, a request for replenishment is generated and sent to the relevant department. This ensures that the procurement process is initiated without delay, minimizing the impact of low stock on operations.

Once the replenishment request is approved, an email notification is automatically sent to the warehouse manager, providing them with the updated product record. This step ensures that the warehouse team is fully informed and can take immediate action to restock the item. The integration of email notifications into the workflow ensures that all stakeholders are kept in the loop, facilitating seamless communication and coordination across departments.

Replastix Innovations recognizes the importance of accurate and timely reporting in driving operational excellence. The Salesforce platform provides robust reporting and analytics capabilities, enabling the organization to generate detailed insights into various aspects of its operations. By leveraging these tools, Replastix Innovations can monitor key performance indicators (KPIs), track progress towards sustainability goals, and identify areas for improvement. This data-driven approach supports informed decision-making and continuous process optimization.

## Business Need:

The warehouse team needed a system that automates restocking when products run low and alerts the responsible manager. Manual checks caused stock shortages and inefficiencies.

## End-to-End Flow

* Sales Rep places an order (Quantity: 150)
* Available stock: 50 → Not enough
* A Restock Request is automatically created (100 units)
* Manager approves → Product stock is updated to 150
* Email notification sent to inventory head

# Phase 1 – Requirement Analysis & Planning

## Stakeholder Roles

* Sales Representative: Creates orders
* Warehouse Supervisor: Approves restocks
* Inventory Manager: Oversees stock updates

## 

## 

## Custom Objects

1. Recycled Product
2. Order
3. Restock Request
4. Plastic Waste
5. Recycling Center

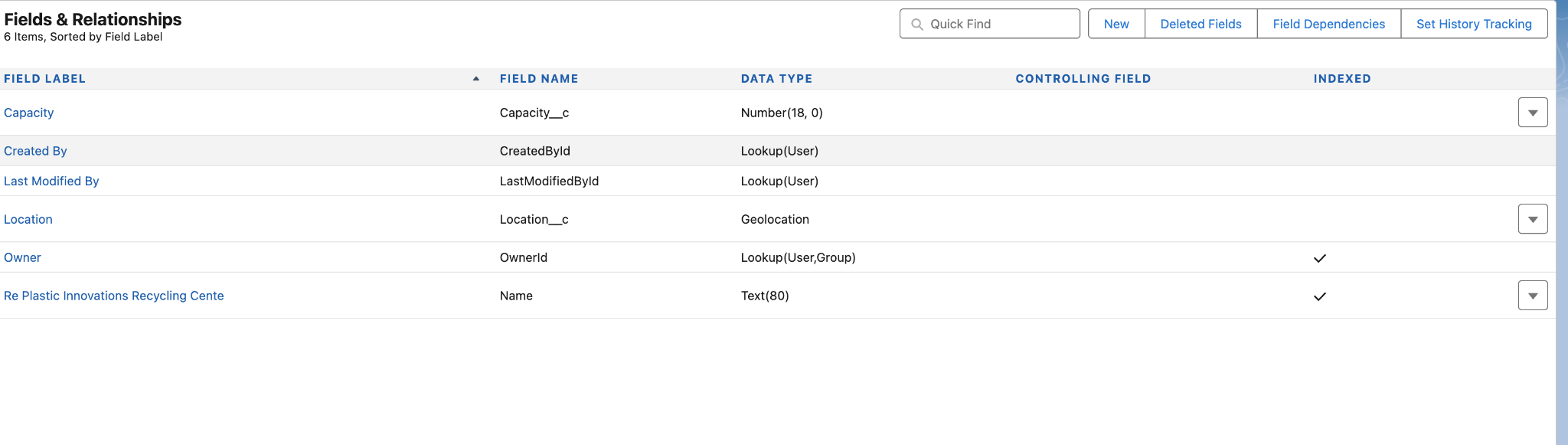
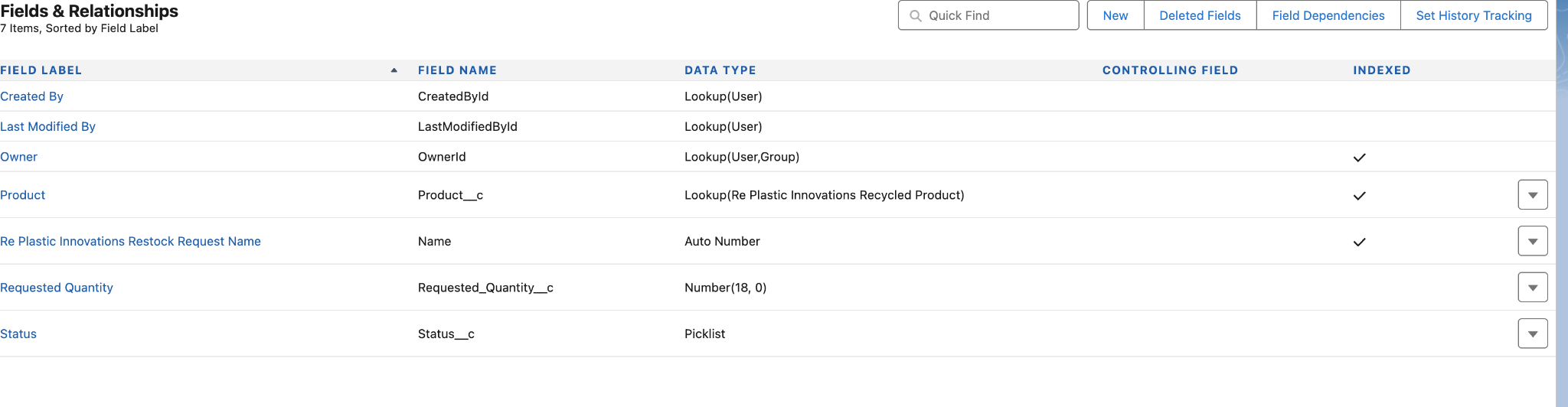
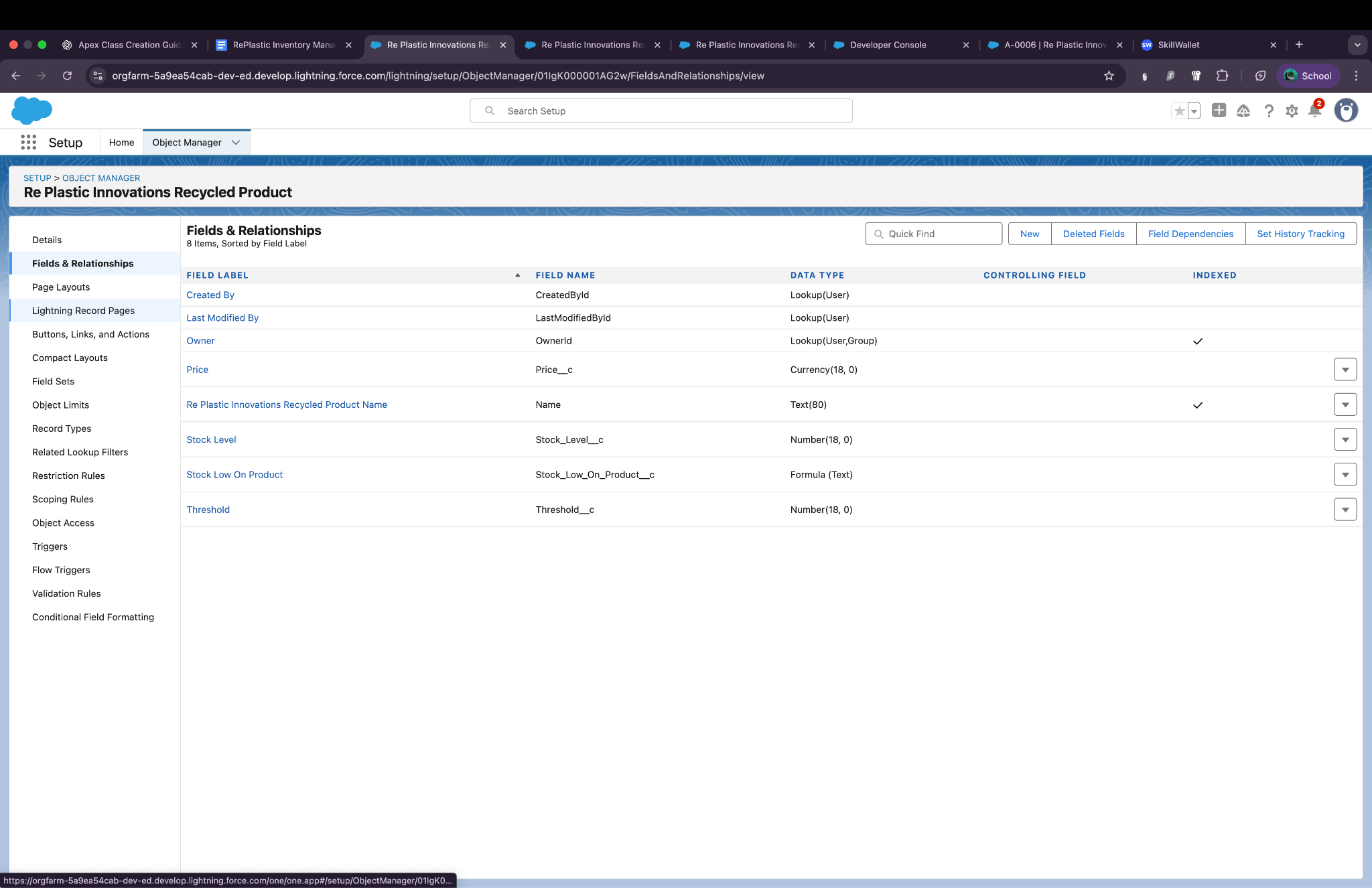
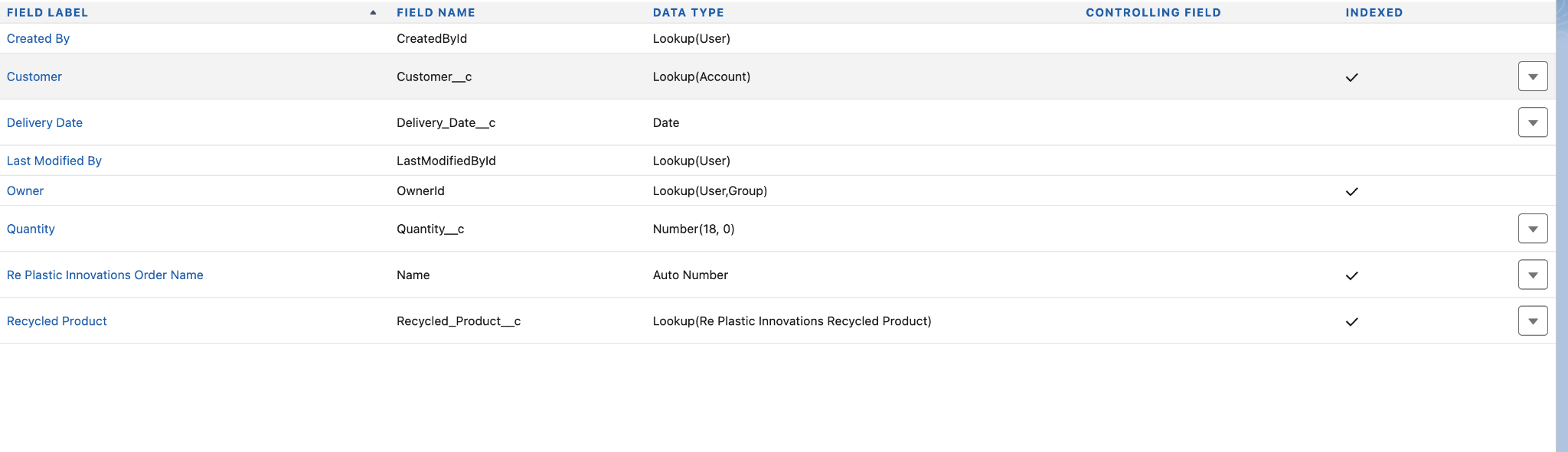
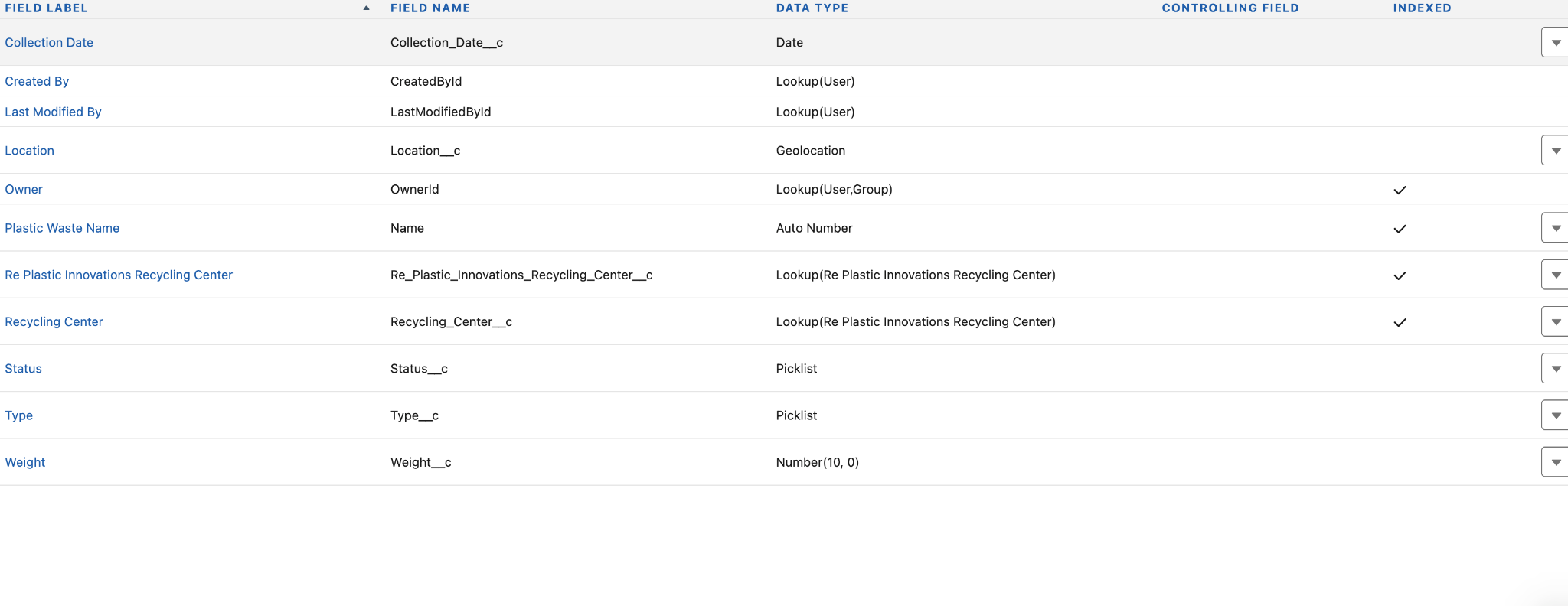
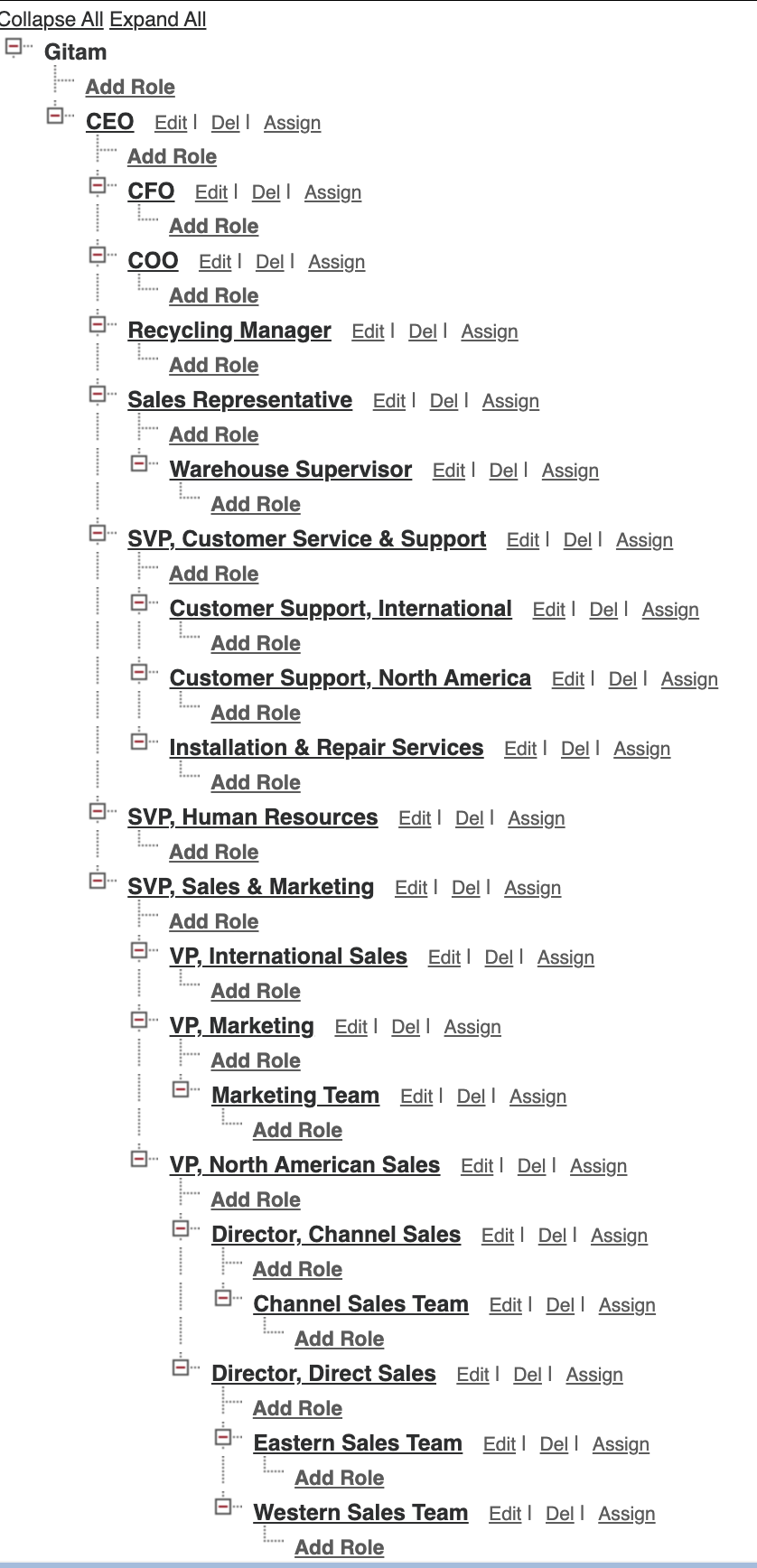
## Entity Relationships

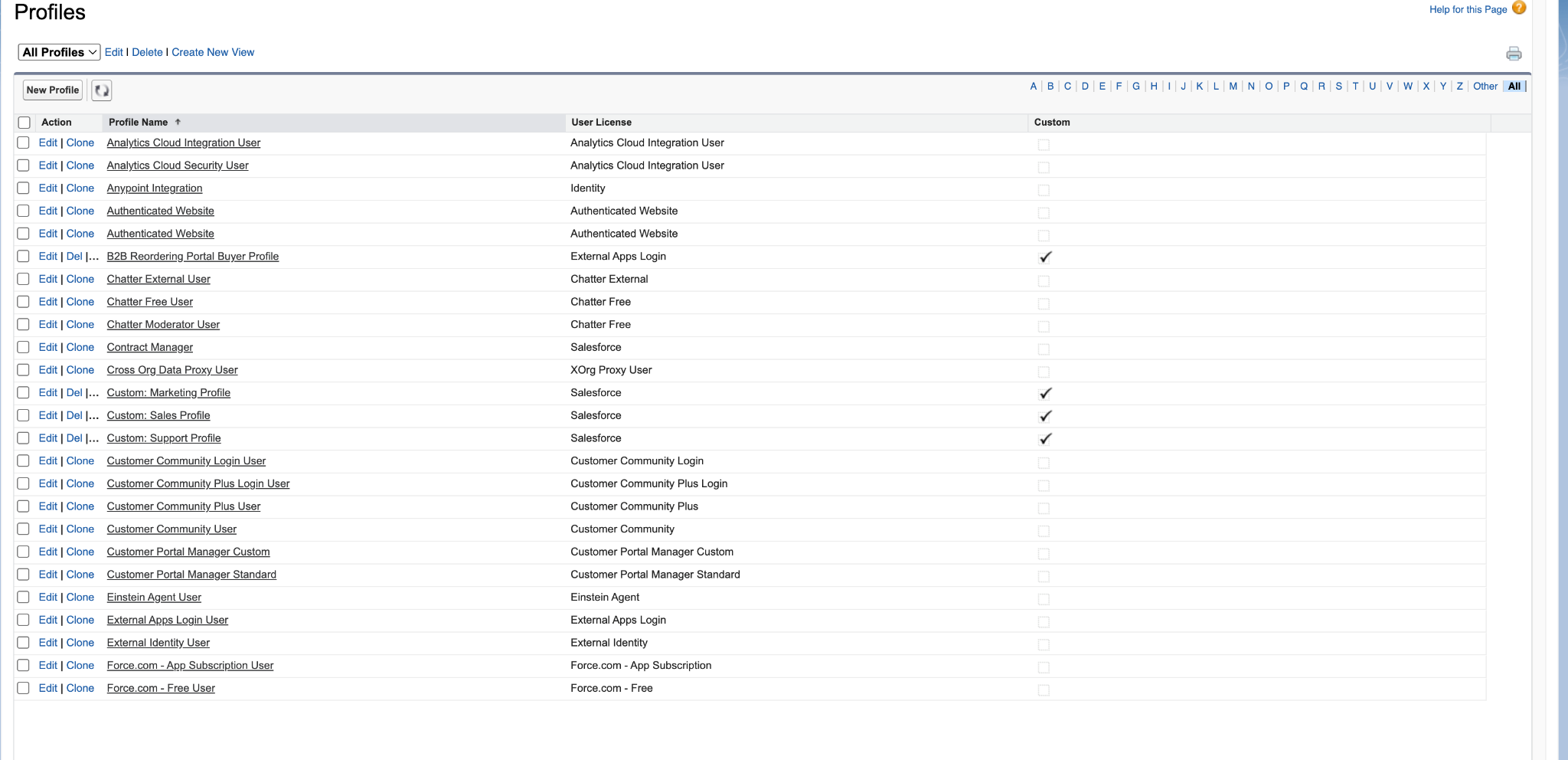
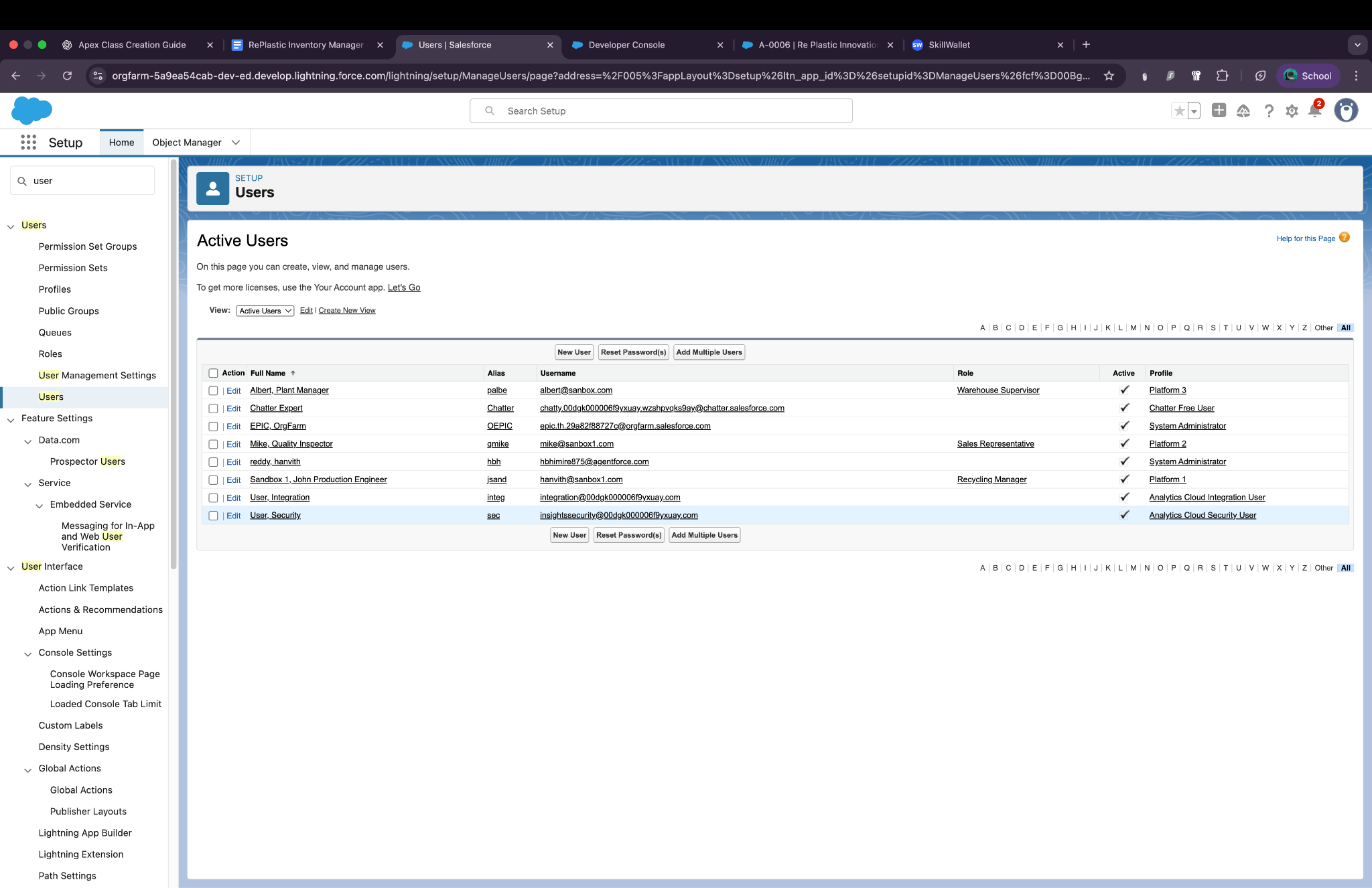
* Order → Lookup to Recycled Product
* Restock Request → Lookup to Recycled Product
* Plastic Waste → Lookup to Recycling Center

## Role Hierarchy

* CEO → Sales Representative → Warehouse Supervisor
* Profiles: Platform User 1, 2, 3 (with FLS and CRUD configured)

Screenshot: ER diagram of object relationships





# Phase 2 – Salesforce Development & Backend Automation

## Apex Classes

* InventoryManager:
  + processOrderStock(): Checks product stock and creates restock request if needed
  + processRestockApproval(): Updates stock on restock approval and calls EmailNotificationHelper
* EmailNotificationHelper:
  + Sends email notification using Messaging.sendEmail()

## Triggers

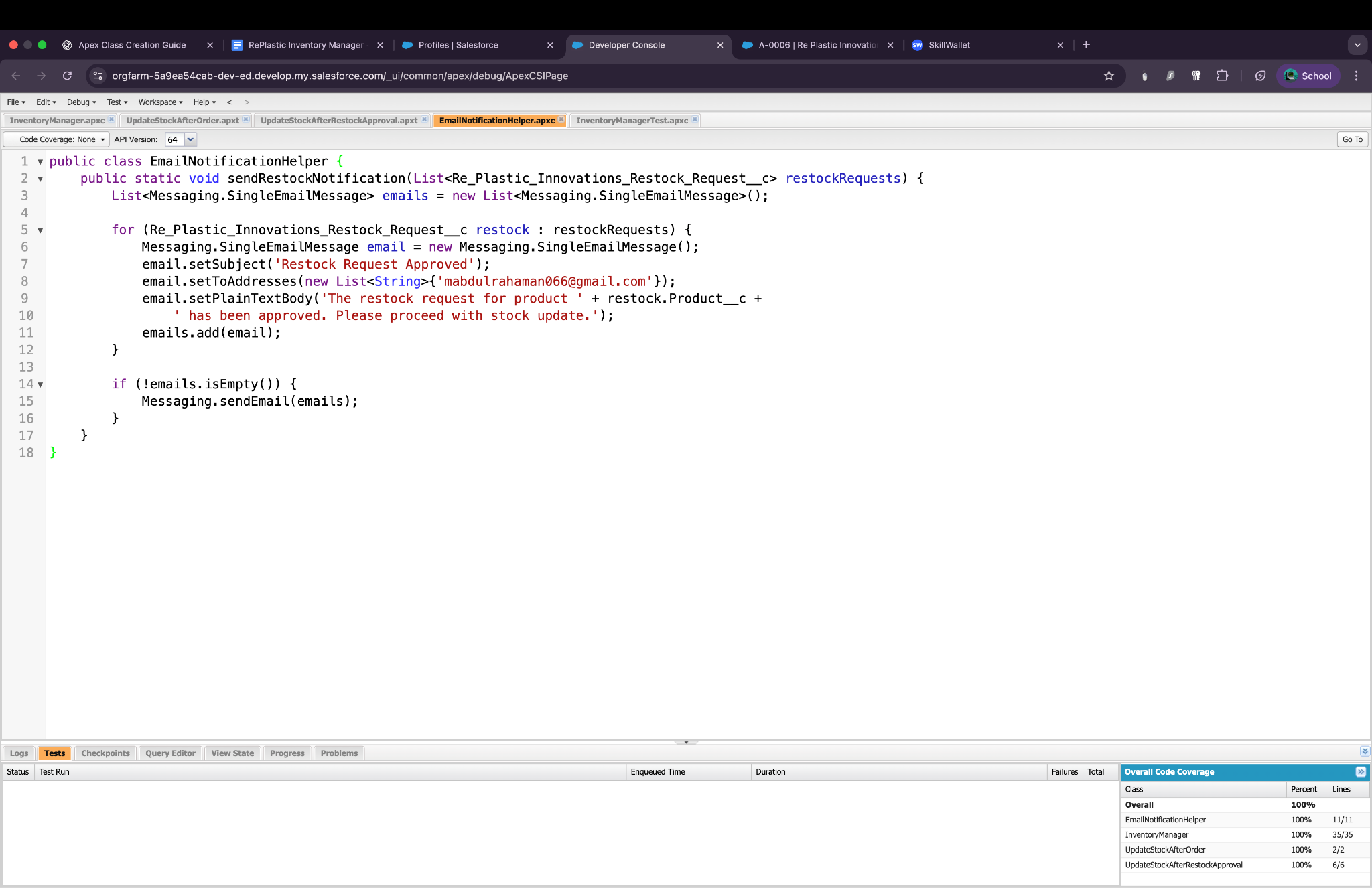
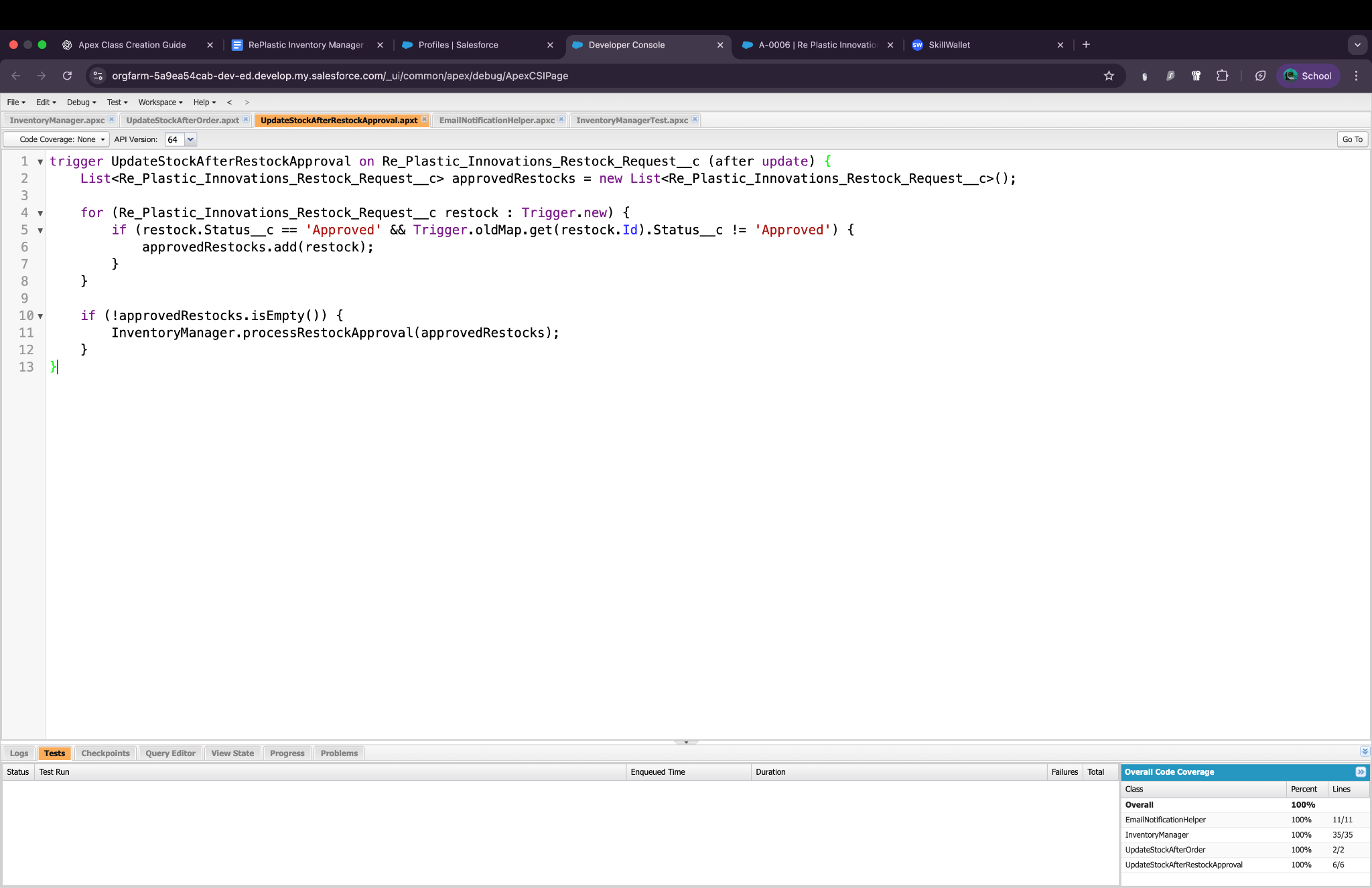
* UpdateStockAfterOrder: Executes after Order insert
* UpdateStockAfterRestockApproval: Executes after Restock update

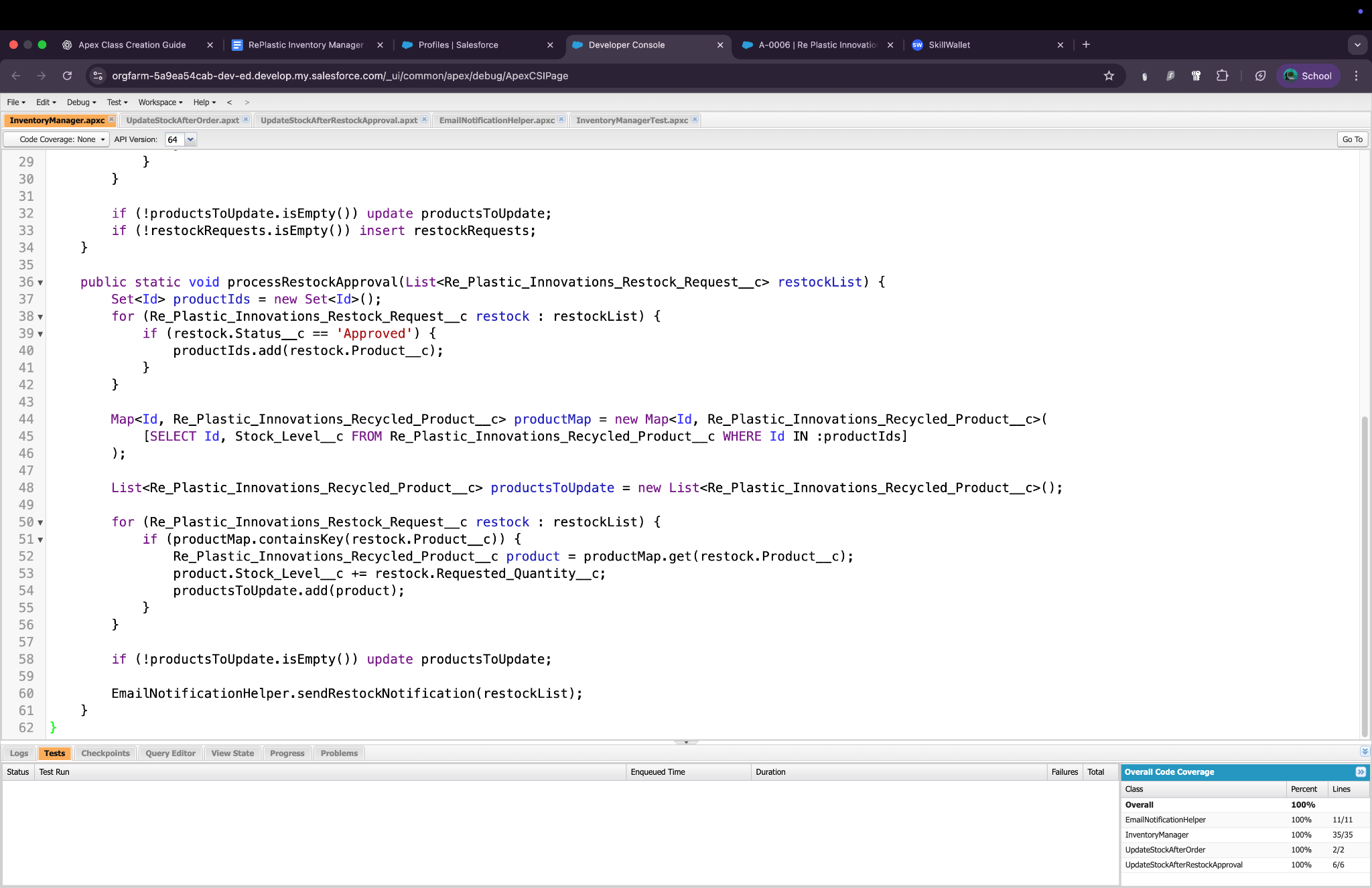
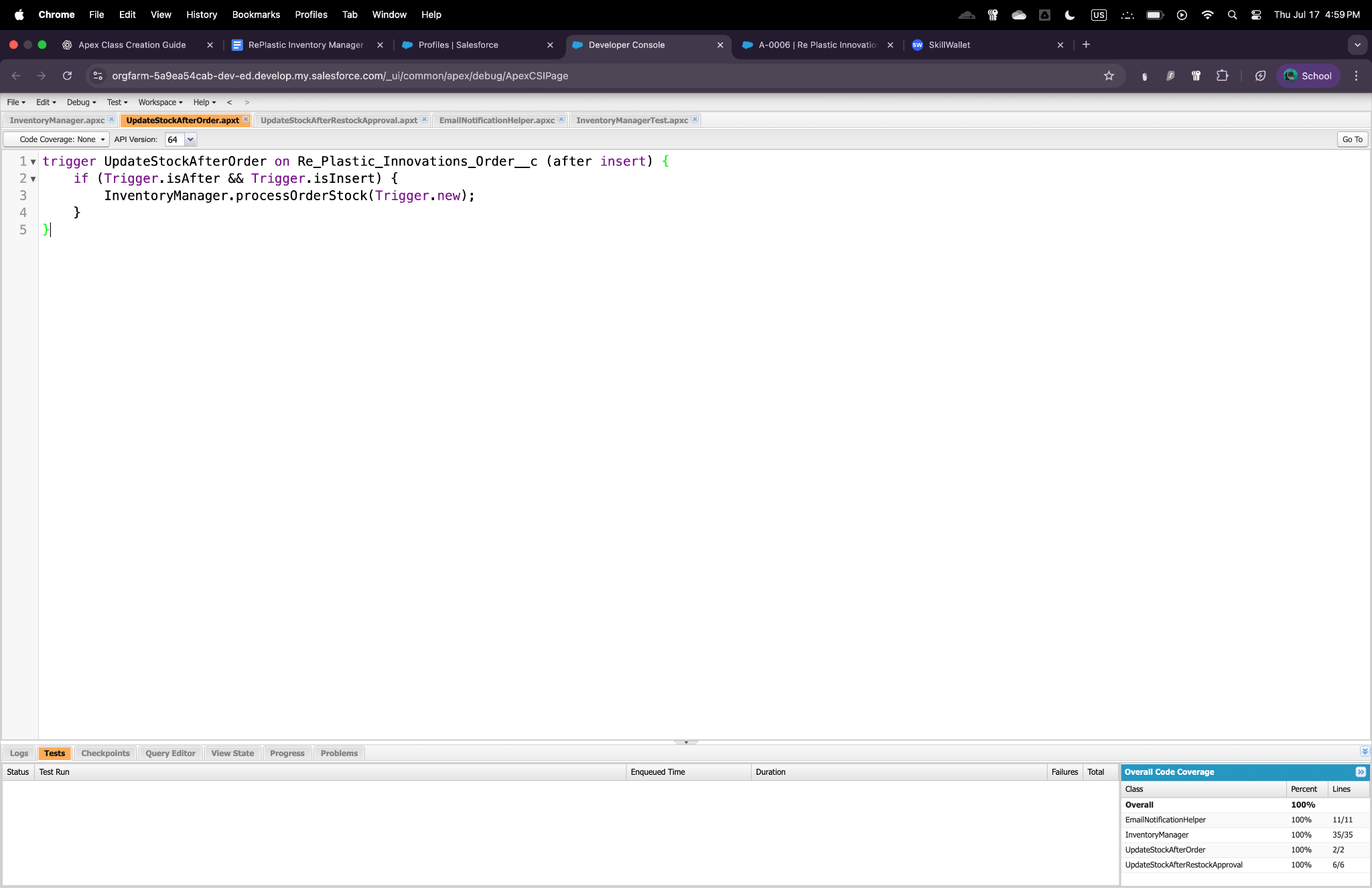
## Validation Rules

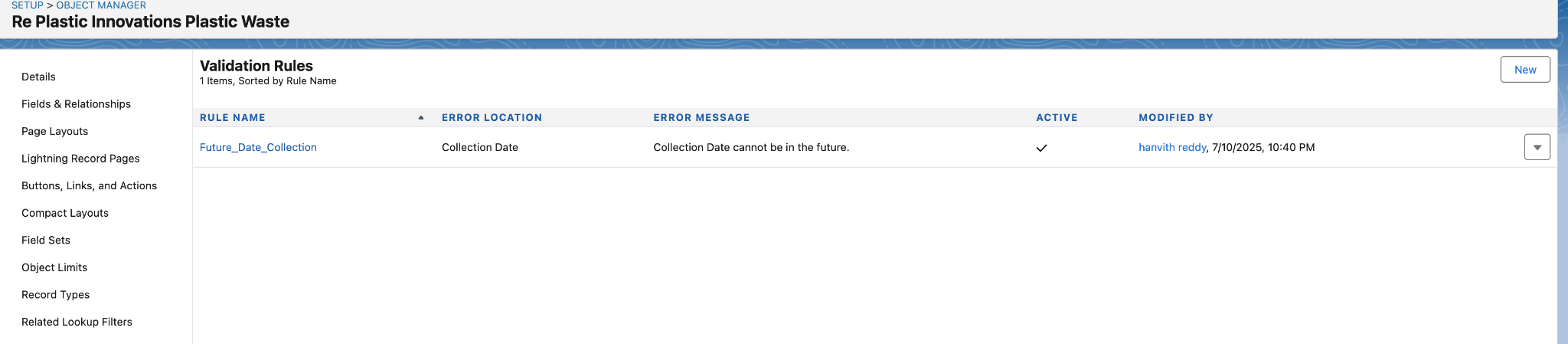
* Quantity > 0 (Order)
* Collection\_Date\_\_c <= TODAY() (Plastic Waste)

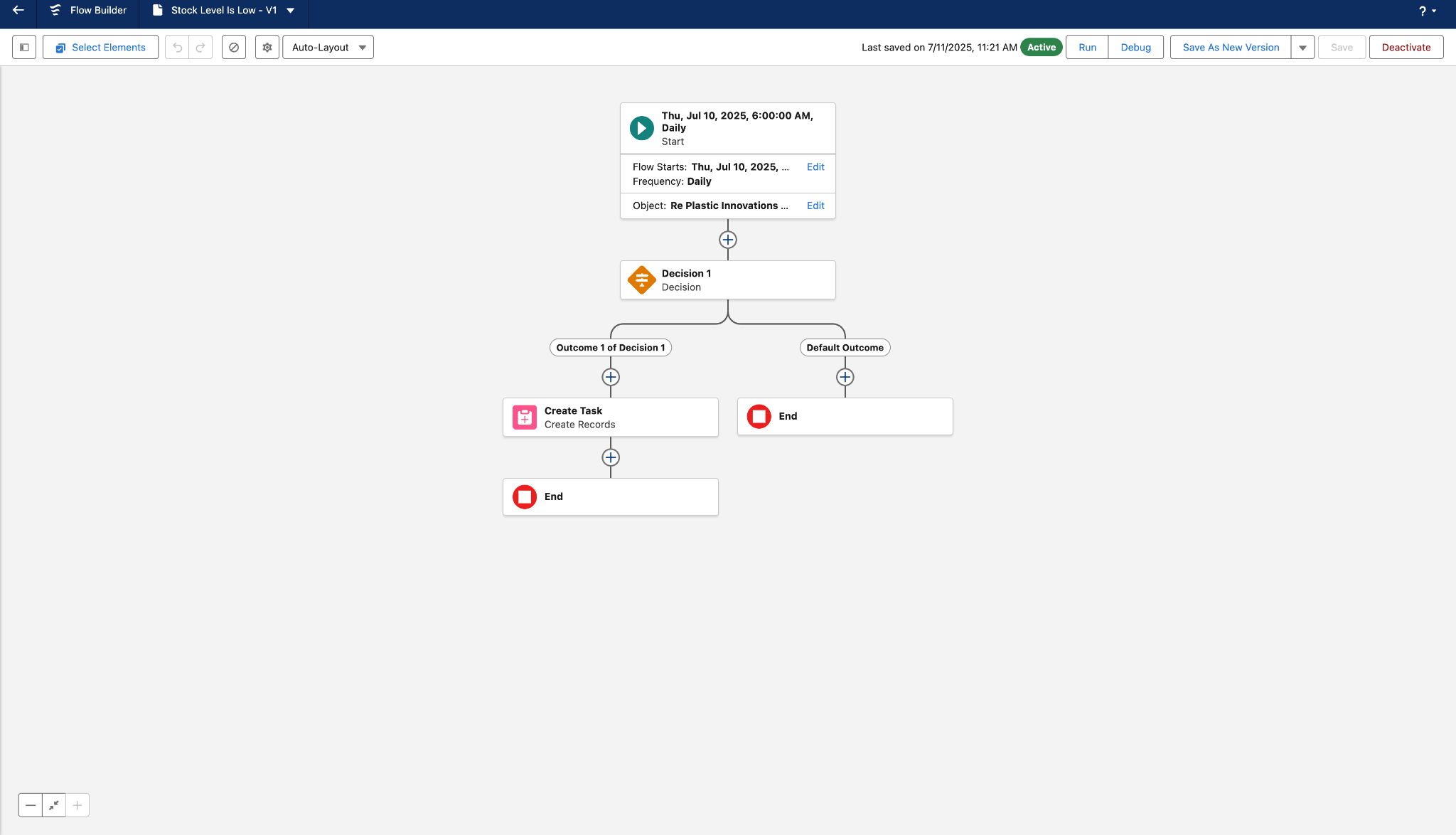
## Flow

* Scheduled Flow (Daily 6 AM): Creates Tasks for products below threshold

Screenshot: Apex Classes, Triggers, Flow UI







# Phase 3 – UI/UX Customization

## App & Tabs

* App Name: Re Plastic Innovations
* Tabs Added: Orders, Recycled Products, Restock Requests

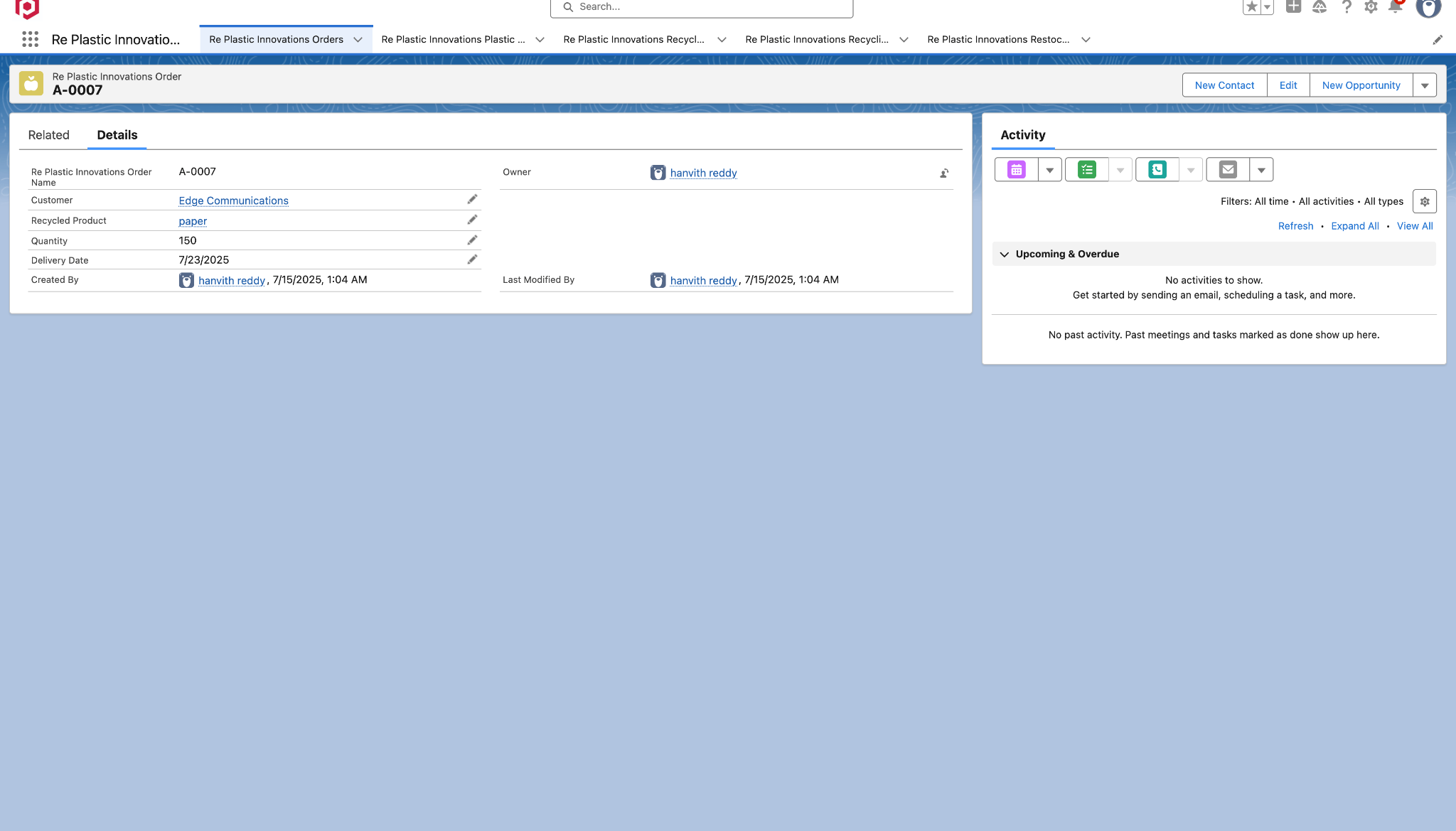
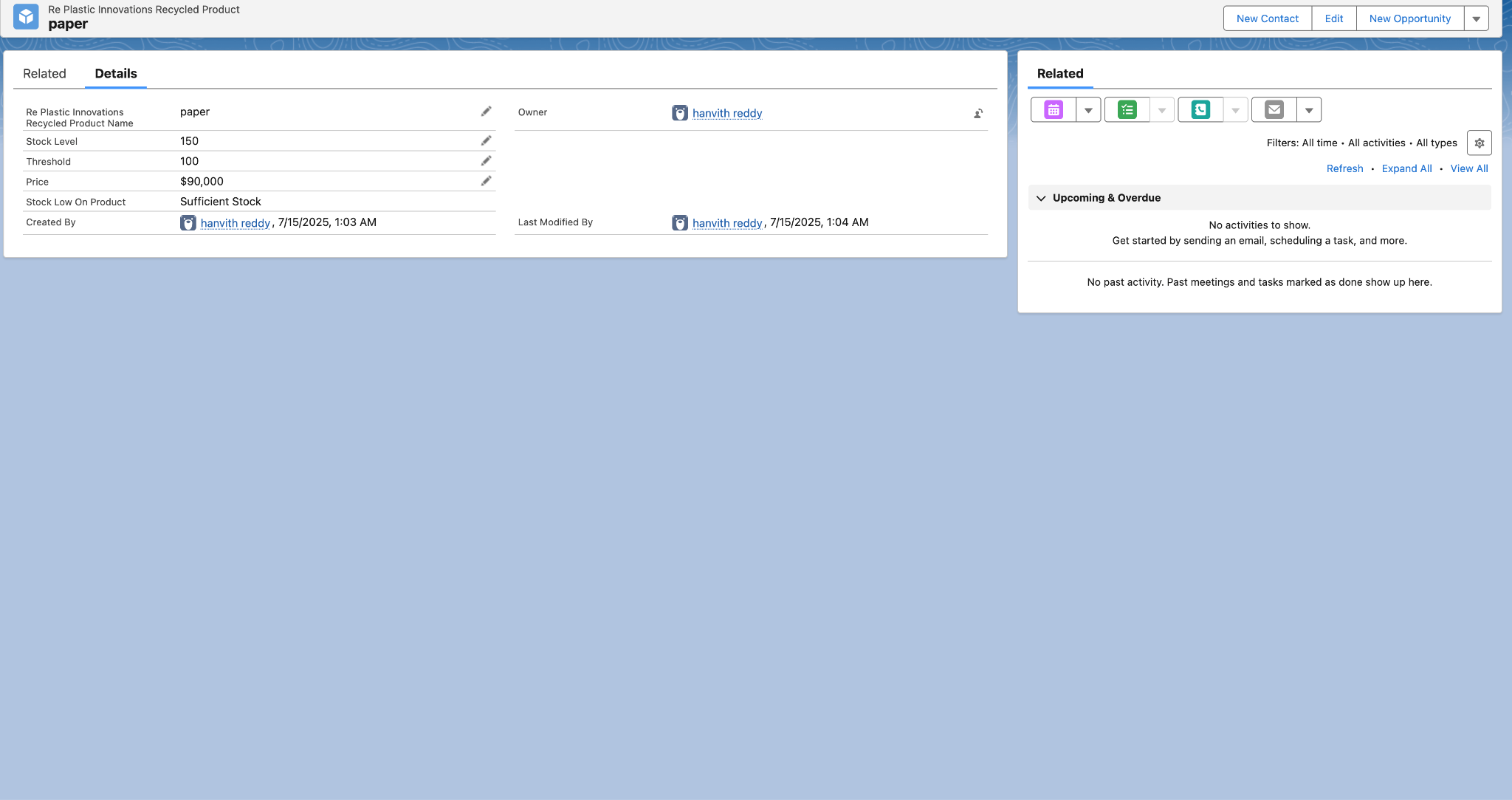
## Record Pages

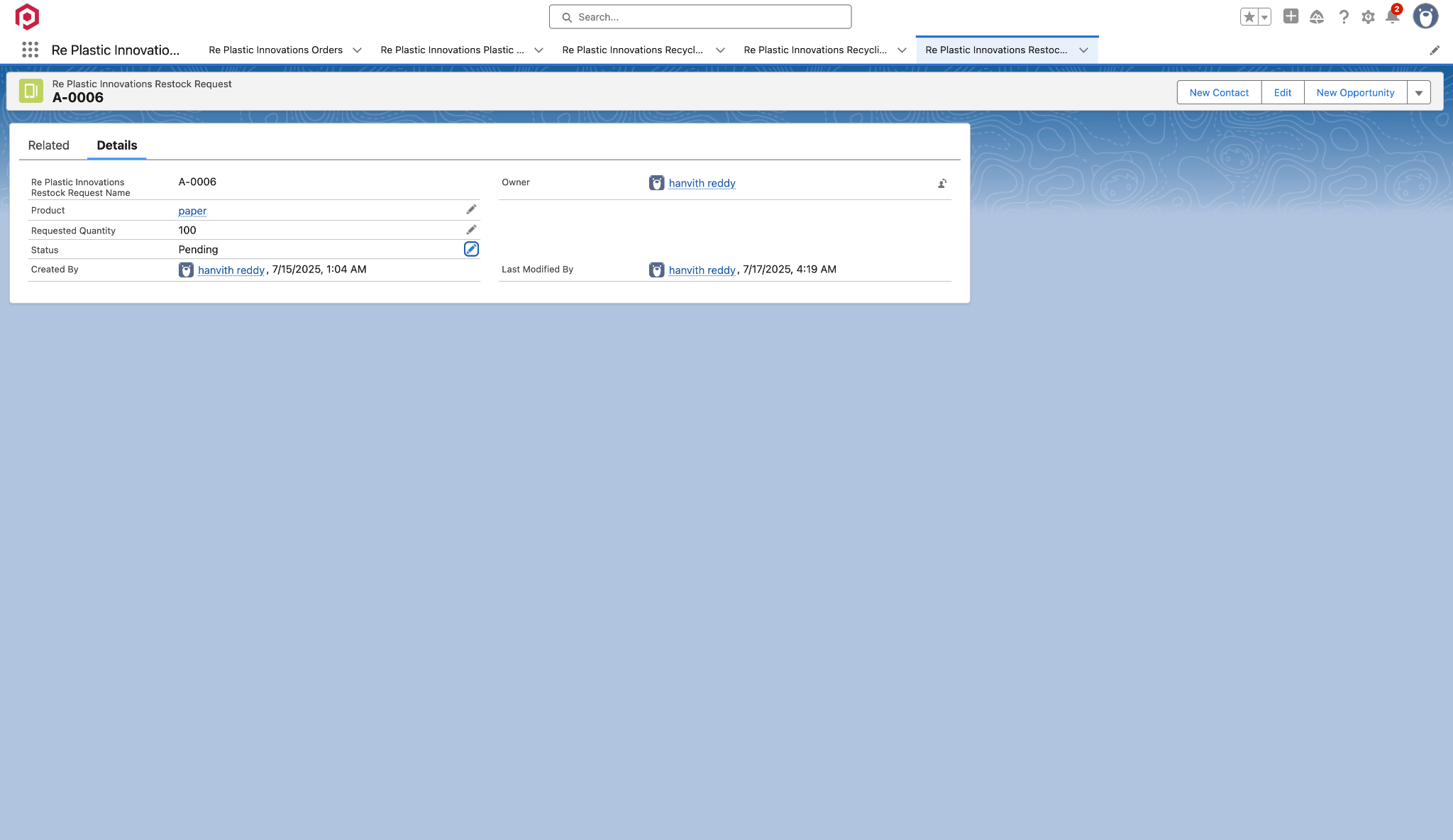
* Enhanced Lightning Page Layouts
  + Related lists for Orders and Restocks
  + Conditional field visibility (e.g., Status, Quantity)

## Example Walkthrough

1. Create Product: Stock 50, Threshold 100
2. Place Order: Quantity 150
3. System auto-generates Restock Request (100 units)
4. Status changed to Approved
5. Product Stock updated to 150
6. Email sent to manager

Screenshot: Order form, Restock record, Product record with updated stock





# Phase 4 – Data Migration, Testing & Security

## Sample Records Created

* Product: Plastic Bottles
* Order: 150 Units

## Testing

* Class: InventoryManagerTest
* Tests order stock deduction, restock approval, email
* Code Coverage: 100%

## Users & Profiles

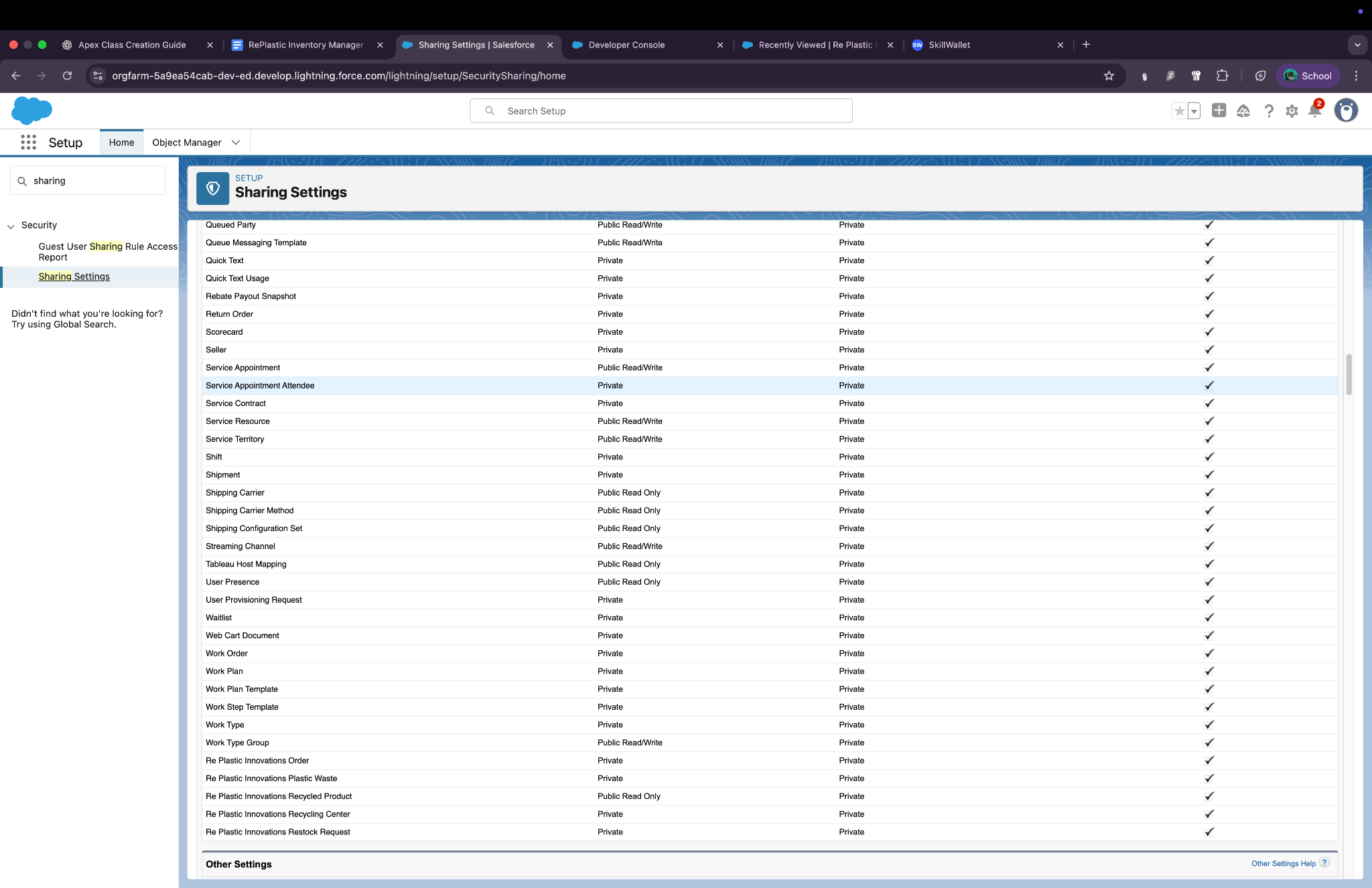
* User 1: John (Platform 1 – Manager)
* User 2: Mike (Platform 2 – Sales)
* User 3: Albert (Platform 3 – Warehouse)

## Security Model

* Sharing Rules set for Restock visibility
* Profiles with CRUD and FLS configured

Screenshot: Developer Console Test Results,





# Phase 5 – Deployment & Maintenance

## Deployment

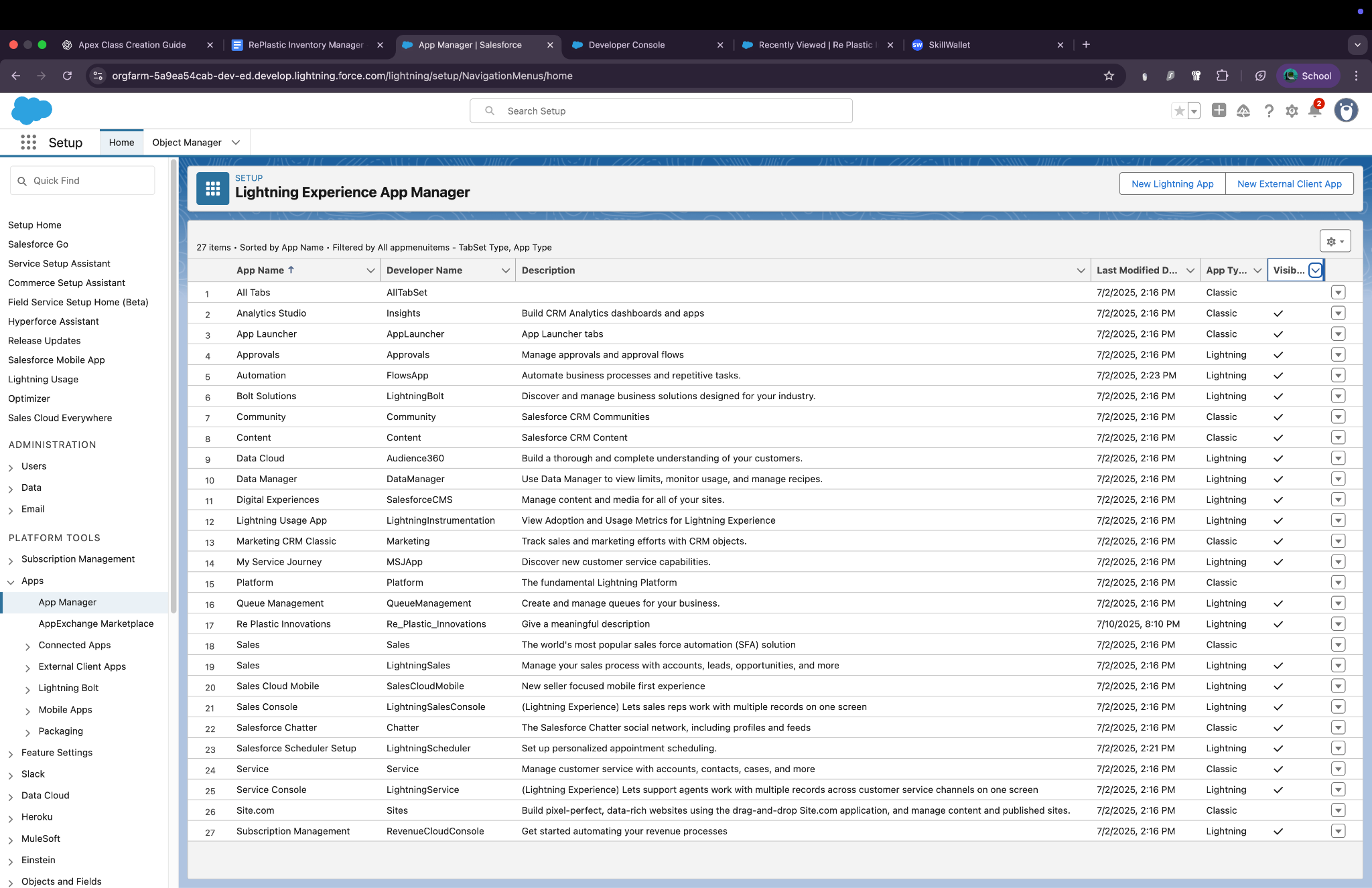
* Components moved using Change Sets
* Verified on sandbox → deployed to production

## Maintenance

* Use Apex logs for monitoring
* Threshold or validation logic can be updated via Setup

## Suggested Improvements

* Convert email notification to flow-based email alert
* Add field history tracking for Stock changes

Screenshot: Change Set screen, Deployment status

# Conclusion

The RePlastic Inventory Manager streamlines the recycled product tracking workflow using Salesforce automation tools. It removes the need for manual stock verification and ensures real-time updates through triggers and scheduled flows.

## Benefits

* Saves time and prevents stockouts
* Ensures accuracy with test validation
* Role-based access enhances security

## Future Scope

* Add Slack/WhatsApp alerts
* Develop dashboard with LWC for live stock tracking
* Multi-level approval process for restocking

# Apex Codes

# Apex Classes

**Inventory manager ;**

public class InventoryManager {

public static void processOrderStock(List<Re\_Plastic\_Innovations\_Order\_\_c> orderList) {

Set<Id> productIds = new Set<Id>();

for (Re\_Plastic\_Innovations\_Order\_\_c order : orderList) {

productIds.add(order.Recycled\_Product\_\_c);

}

Map<Id, Re\_Plastic\_Innovations\_Recycled\_Product\_\_c> productMap = new Map<Id, Re\_Plastic\_Innovations\_Recycled\_Product\_\_c>(

[SELECT Id, Stock\_Level\_\_c, Threshold\_\_c FROM Re\_Plastic\_Innovations\_Recycled\_Product\_\_c WHERE Id IN :productIds]

);

List<Re\_Plastic\_Innovations\_Recycled\_Product\_\_c> productsToUpdate = new List<Re\_Plastic\_Innovations\_Recycled\_Product\_\_c>();

List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c> restockRequests = new List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c>();

for (Re\_Plastic\_Innovations\_Order\_\_c order : orderList) {

Re\_Plastic\_Innovations\_Recycled\_Product\_\_c product = productMap.get(order.Recycled\_Product\_\_c);

if (product != null) {

if (product.Stock\_Level\_\_c >= order.Quantity\_\_c) {

product.Stock\_Level\_\_c -= order.Quantity\_\_c;

productsToUpdate.add(product);

} else {

restockRequests.add(new Re\_Plastic\_Innovations\_Restock\_Request\_\_c(

Product\_\_c = product.Id,

Requested\_Quantity\_\_c = order.Quantity\_\_c - product.Stock\_Level\_\_c,

Status\_\_c = 'Pending'

));

}

}

}

if (!productsToUpdate.isEmpty()) update productsToUpdate;

if (!restockRequests.isEmpty()) insert restockRequests;

}

public static void processRestockApproval(List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c> restockList) {

Set<Id> productIds = new Set<Id>();

for (Re\_Plastic\_Innovations\_Restock\_Request\_\_c restock : restockList) {

if (restock.Status\_\_c == 'Approved') {

productIds.add(restock.Product\_\_c);

}

}

Map<Id, Re\_Plastic\_Innovations\_Recycled\_Product\_\_c> productMap = new Map<Id, Re\_Plastic\_Innovations\_Recycled\_Product\_\_c>(

[SELECT Id, Stock\_Level\_\_c FROM Re\_Plastic\_Innovations\_Recycled\_Product\_\_c WHERE Id IN :productIds]

);

List<Re\_Plastic\_Innovations\_Recycled\_Product\_\_c> productsToUpdate = new List<Re\_Plastic\_Innovations\_Recycled\_Product\_\_c>();

for (Re\_Plastic\_Innovations\_Restock\_Request\_\_c restock : restockList) {

if (productMap.containsKey(restock.Product\_\_c)) {

Re\_Plastic\_Innovations\_Recycled\_Product\_\_c product = productMap.get(restock.Product\_\_c);

product.Stock\_Level\_\_c += restock.Requested\_Quantity\_\_c;

productsToUpdate.add(product);

}

}

if (!productsToUpdate.isEmpty()) update productsToUpdate;

EmailNotificationHelper.sendRestockNotification(restockList);

}

}

### **EmailNotificationHelper**

public class EmailNotificationHelper {

public static void sendRestockNotification(List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c> restockRequests) {

List<Messaging.SingleEmailMessage> emails = new List<Messaging.SingleEmailMessage>();

for (Re\_Plastic\_Innovations\_Restock\_Request\_\_c restock : restockRequests) {

Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();

email.setSubject('Restock Request Approved');

email.setToAddresses(new List<String>{'mabdulrahaman066@gmail.com'});

email.setPlainTextBody('The restock request for product ' + restock.Product\_\_c +

' has been approved. Please proceed with stock update.');

emails.add(email);

}

if (!emails.isEmpty()) {

Messaging.sendEmail(emails);

}

}

}

**InventoryManagerTest**

@isTest

public class InventoryManagerTest {

@testSetup

static void setupTestData() {

// Create sample product

Re\_Plastic\_Innovations\_Recycled\_Product\_\_c product = new Re\_Plastic\_Innovations\_Recycled\_Product\_\_c(

Stock\_Level\_\_c = 50,

Threshold\_\_c = 10

);

insert product;

// Create two orders - one valid, one exceeding stock

List<Re\_Plastic\_Innovations\_Order\_\_c> orders = new List<Re\_Plastic\_Innovations\_Order\_\_c>{

new Re\_Plastic\_Innovations\_Order\_\_c(Recycled\_Product\_\_c = product.Id, Quantity\_\_c = 20),

new Re\_Plastic\_Innovations\_Order\_\_c(Recycled\_Product\_\_c = product.Id, Quantity\_\_c = 40)

};

insert orders;

}

@isTest

static void testProcessOrderStock() {

List<Re\_Plastic\_Innovations\_Order\_\_c> orders = [

SELECT Id, Recycled\_Product\_\_c, Quantity\_\_c

FROM Re\_Plastic\_Innovations\_Order\_\_c

];

Test.startTest();

InventoryManager.processOrderStock(orders);

Test.stopTest();

Re\_Plastic\_Innovations\_Recycled\_Product\_\_c product = [

SELECT Stock\_Level\_\_c

FROM Re\_Plastic\_Innovations\_Recycled\_Product\_\_c

LIMIT 1

];

System.assertEquals(10, product.Stock\_Level\_\_c, 'Stock should reduce correctly');

List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c> restocks = [

SELECT Id

FROM Re\_Plastic\_Innovations\_Restock\_Request\_\_c

WHERE Status\_\_c = 'Pending'

];

System.assertEquals(1, restocks.size(), 'Only one restock request should be created');

}

@isTest

static void testProcessRestockApproval() {

List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c> restocks = [

SELECT Id, Product\_\_c, Requested\_Quantity\_\_c

FROM Re\_Plastic\_Innovations\_Restock\_Request\_\_c

];

for (Re\_Plastic\_Innovations\_Restock\_Request\_\_c req : restocks) {

req.Status\_\_c = 'Approved';

}

update restocks;

Re\_Plastic\_Innovations\_Recycled\_Product\_\_c productBefore = [

SELECT Stock\_Level\_\_c

FROM Re\_Plastic\_Innovations\_Recycled\_Product\_\_c

LIMIT 1

];

Decimal initialStock = productBefore.Stock\_Level\_\_c;

Decimal totalRestockQuantity = 0;

for (Re\_Plastic\_Innovations\_Restock\_Request\_\_c req : restocks) {

totalRestockQuantity += req.Requested\_Quantity\_\_c;

}

Test.startTest();

InventoryManager.processRestockApproval(restocks);

Test.stopTest();

Re\_Plastic\_Innovations\_Recycled\_Product\_\_c productAfter = [

SELECT Stock\_Level\_\_c

FROM Re\_Plastic\_Innovations\_Recycled\_Product\_\_c

LIMIT 1

];

Decimal expectedStock = initialStock + totalRestockQuantity;

System.assertEquals(expectedStock, productAfter.Stock\_Level\_\_c, 'Stock should update correctly after approval');

}

}

## Apex Triggers

**UpdateStockAfterOrder**

trigger UpdateStockAfterOrder on Re\_Plastic\_Innovations\_Order\_\_c (after insert) {

if (Trigger.isAfter && Trigger.isInsert) {

InventoryManager.processOrderStock(Trigger.new);

}

}

**UpdateStockAfterRestockApproval**

trigger UpdateStockAfterRestockApproval on Re\_Plastic\_Innovations\_Restock\_Request\_\_c (after update) {

List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c> approvedRestocks = new List<Re\_Plastic\_Innovations\_Restock\_Request\_\_c>();

for (Re\_Plastic\_Innovations\_Restock\_Request\_\_c restock : Trigger.new) {

if (restock.Status\_\_c == 'Approved' && Trigger.oldMap.get(restock.Id).Status\_\_c != 'Approved') {

approvedRestocks.add(restock);

}

}

if (!approvedRestocks.isEmpty()) {

InventoryManager.processRestockApproval(approvedRestocks);

}

}

**THANK YOU**