**CRM APPLICATION FOR JEWEL MANAGEMENT DEVELOPER**

**COLLEGE NAME**: SHRI NEHRU MAHA VIDYALAYA COLLEGE OF ARTS AND SCIENCE.

**COLLEGE CODE:** BRU 26

**TEAM ID: NM2025TMID20505**

**TEAM MEMBERS:**

TEAMLEADER**:** **RAAJANI.S**

**EMAIL**: raajaniraajani6@gmail.com

TEAMMEMBER: **SARANYA.R**

**EMAIL:** saranyaramdass90@gmail.com

TEAMMEMBER**: SARANYA.S**

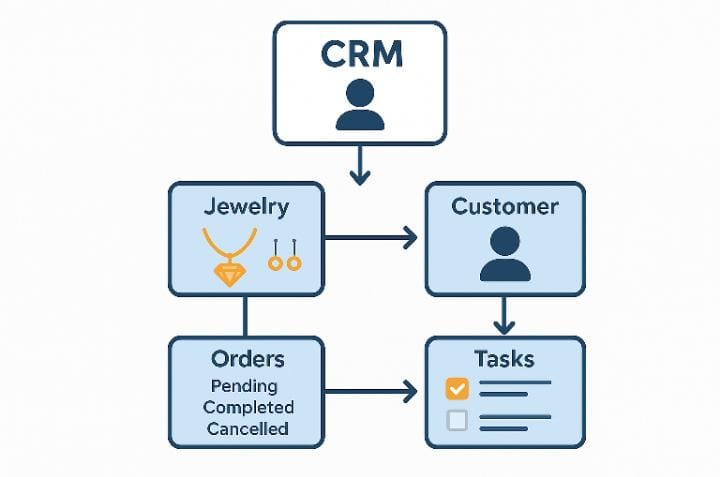
**EMAIL:**  ssaranya9537@gmail.com

TEAMMEMBER**: SANJAY KUMAR.R**

**EMAIL:** sanjaysanjay638041@gmail.com

* 1. **INTRODUCTION**
  2. **PROJECT OVERVIEW**

The Jewellery industry requires efficient management of customers, sales, inventory, and services. A **CRM (Customer Relationship Management) application** helps jewellery businesses streamline operations, maintain customer relationships, track sales, manage stock, and provide personalized experiences.This project focuses on building a **CRM application tailored for jewellery management**, providing a centralized platform for handling customers, employees, suppliers, and business activities.



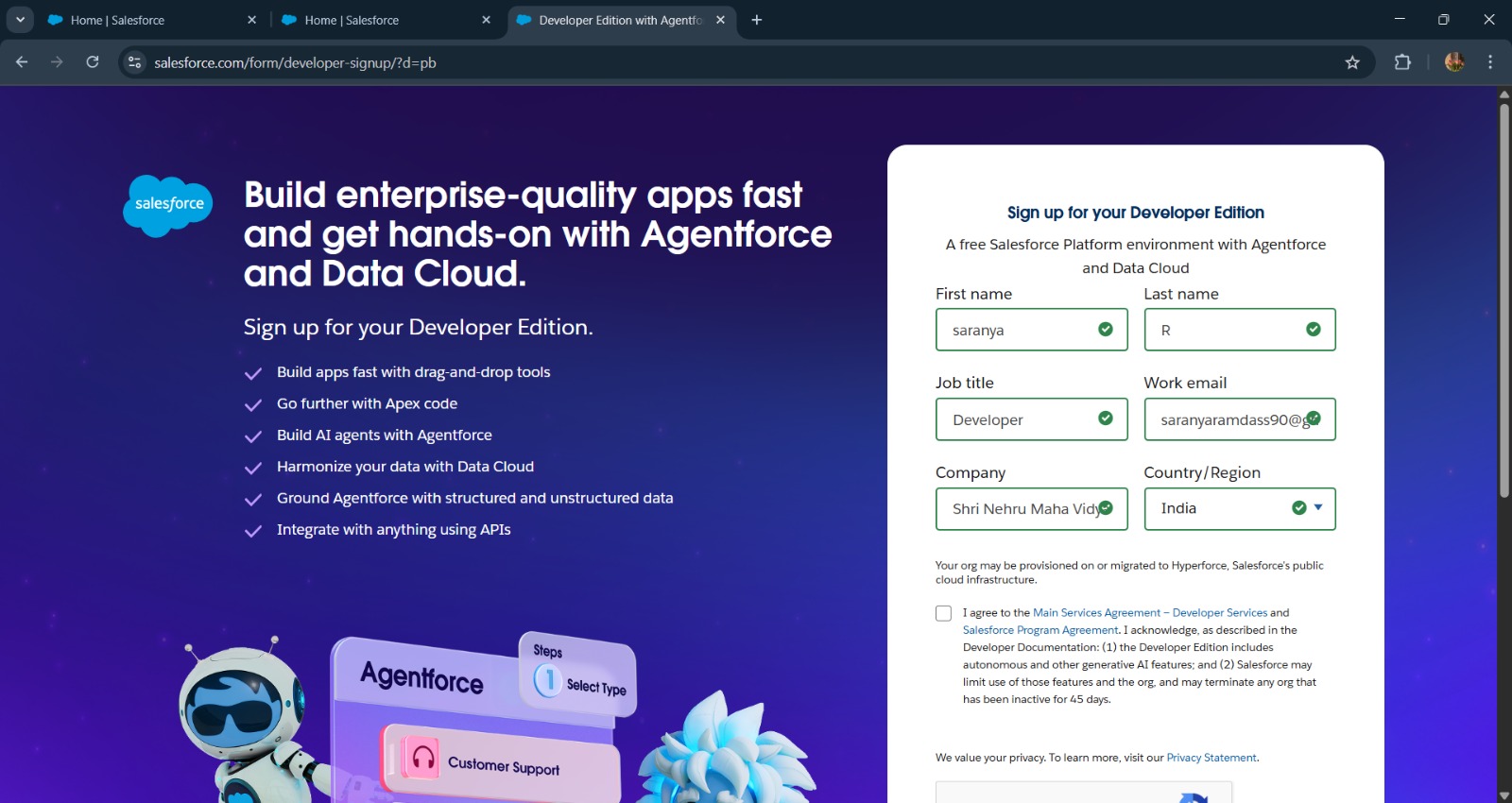
* 1. **PURPOSE**

The purpose of this project is to **streamline jewellery business operations** by providing a centralized **CRM solution** to manage customers, sales, billing, inventory, suppliers, and employees. It helps jewellery stores improve **customer relationships, reduce manual errors, track stock effectively, and generate real-time reports** for better decision-making and business growth.

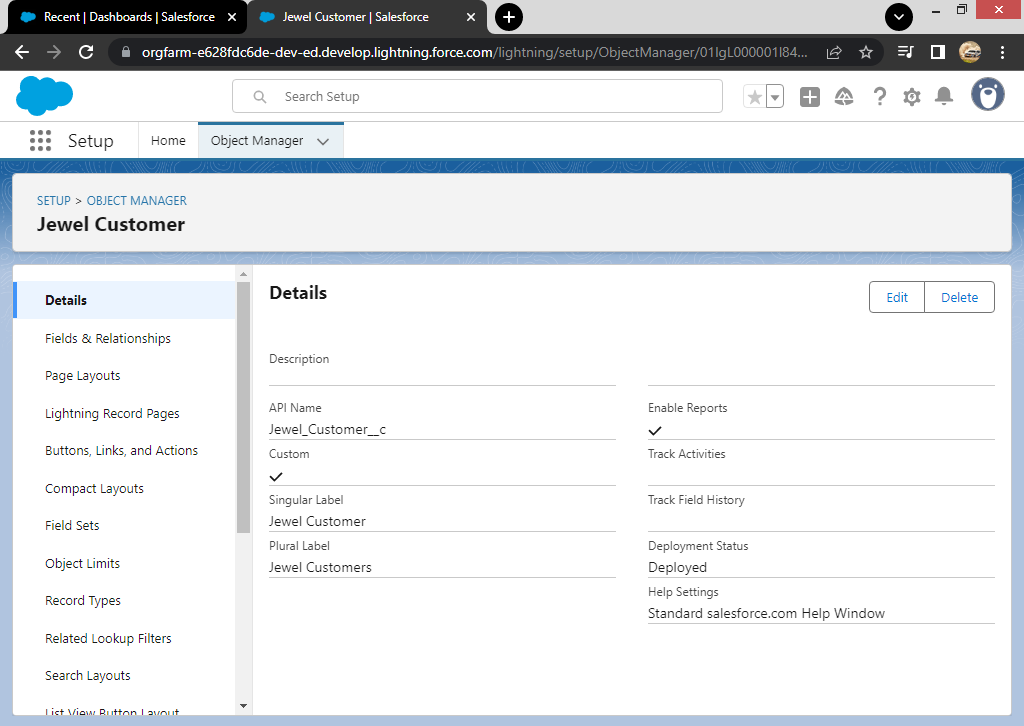
**DEVELOPMENT PHASE**

**CREATING DEVELOPER ACCOUNT:**

By using this url: <https://www.salesforce.com/form/developer-signup/?d=pb>

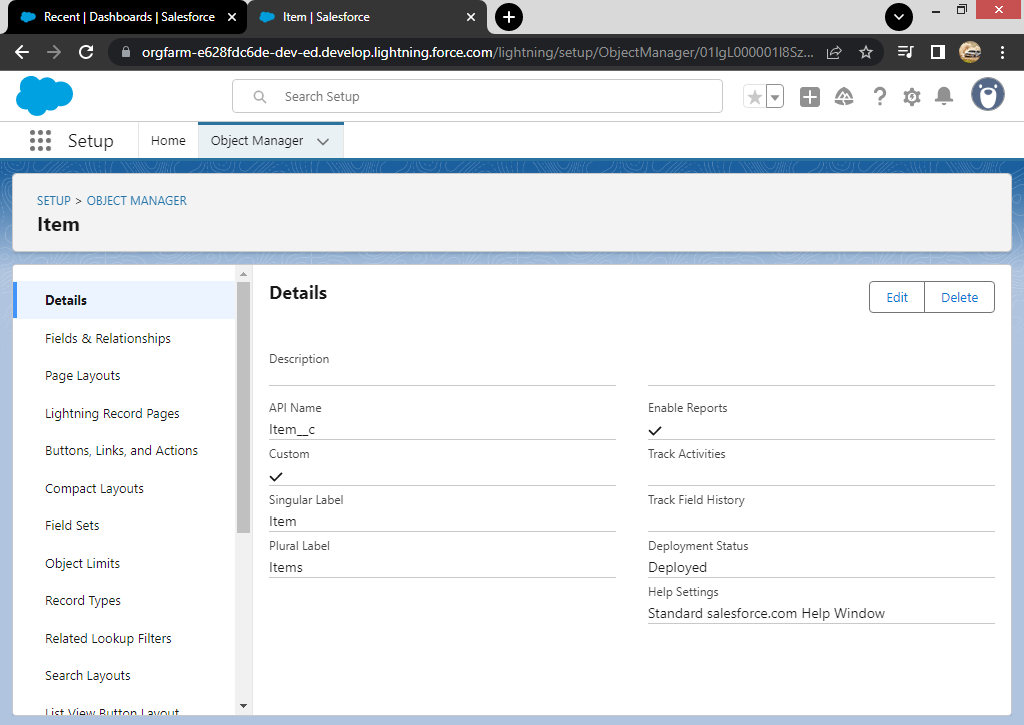


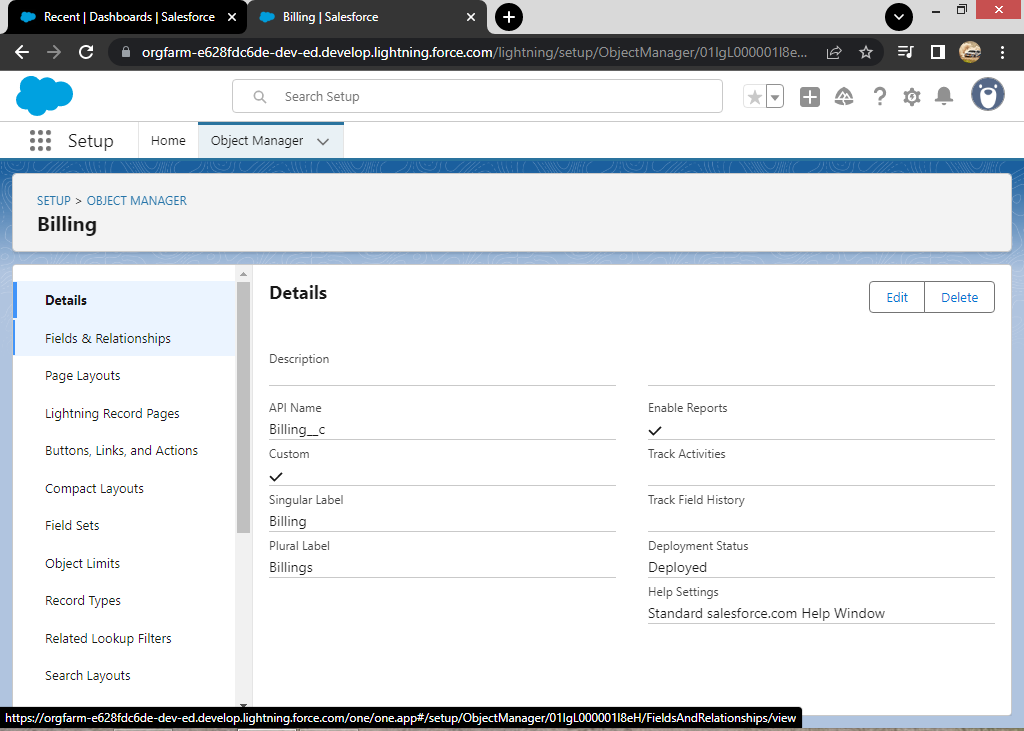
**Create a jewel customer object**

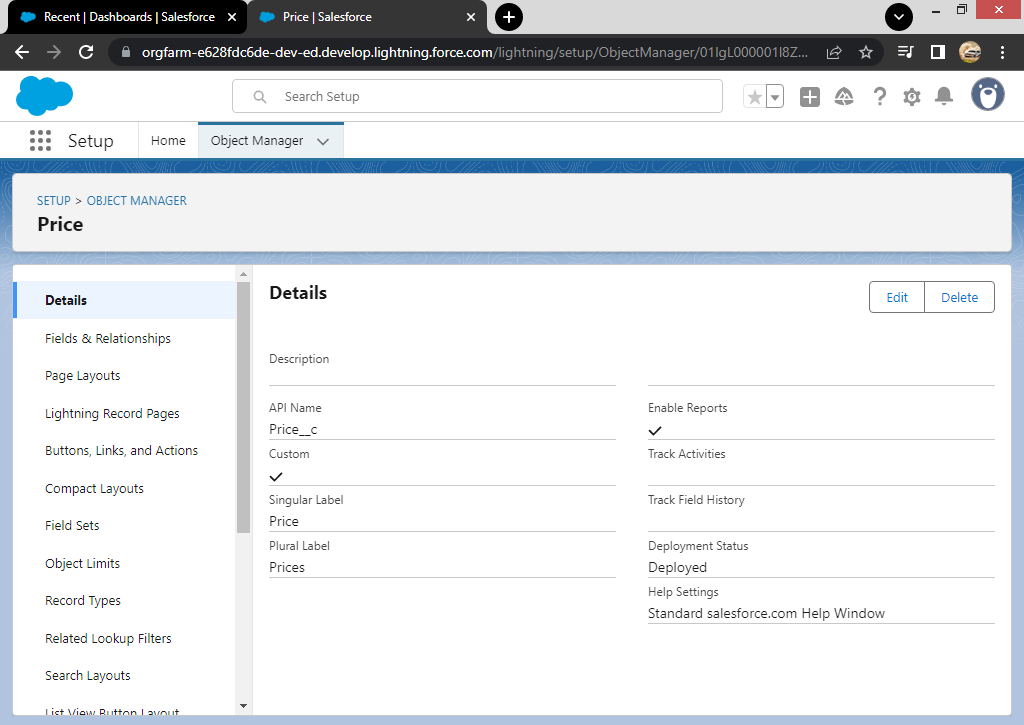
**  
  
  
  
  
OBJECTS**

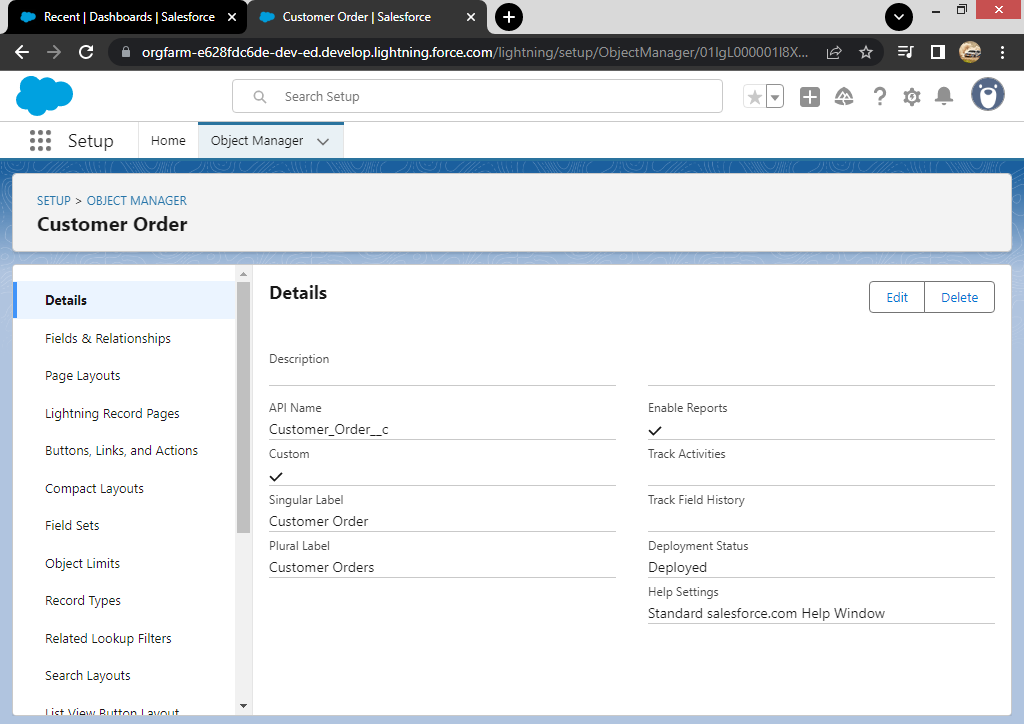
CREATED OBJECTS:

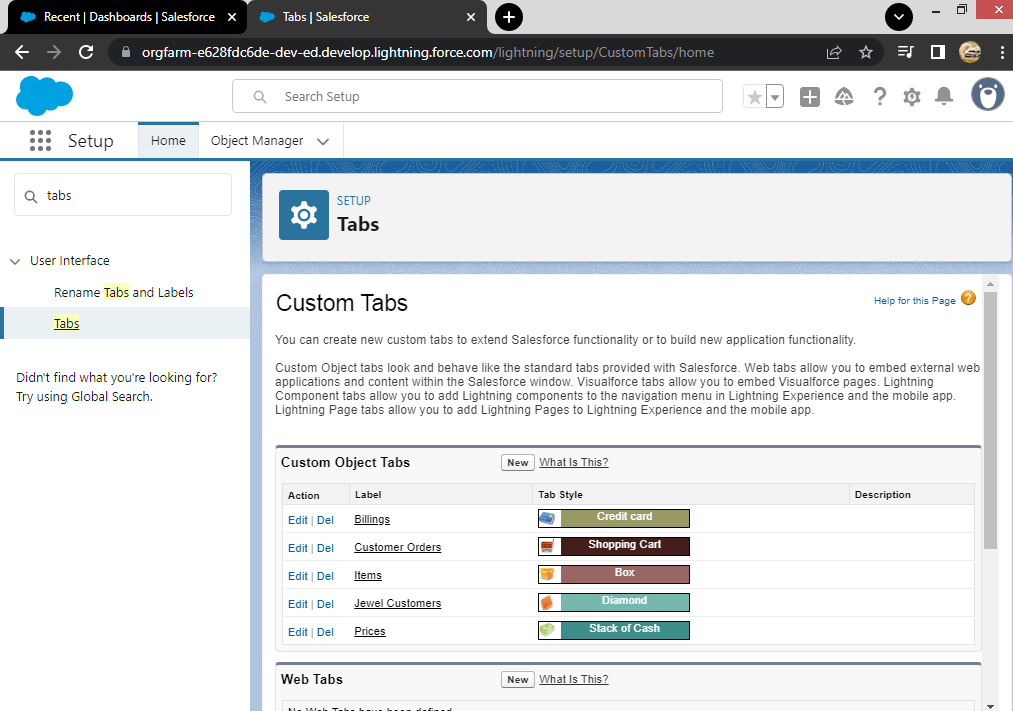
* **ITEMS**
* **BILLINGS**
* **PRICES**
* **CUSTOMER ORDER**

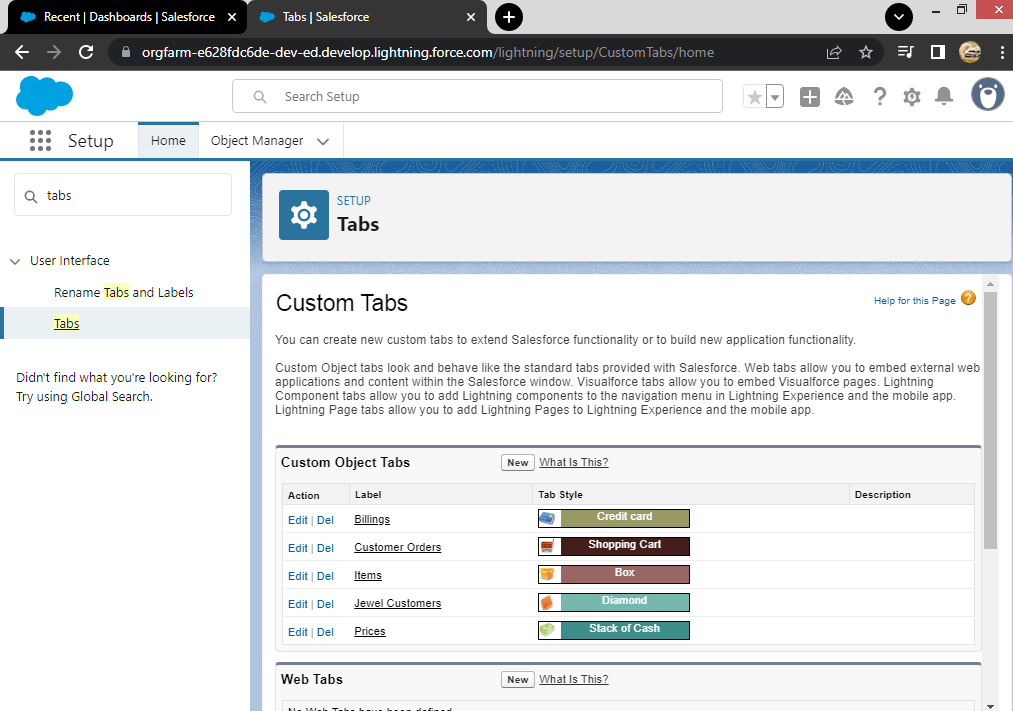


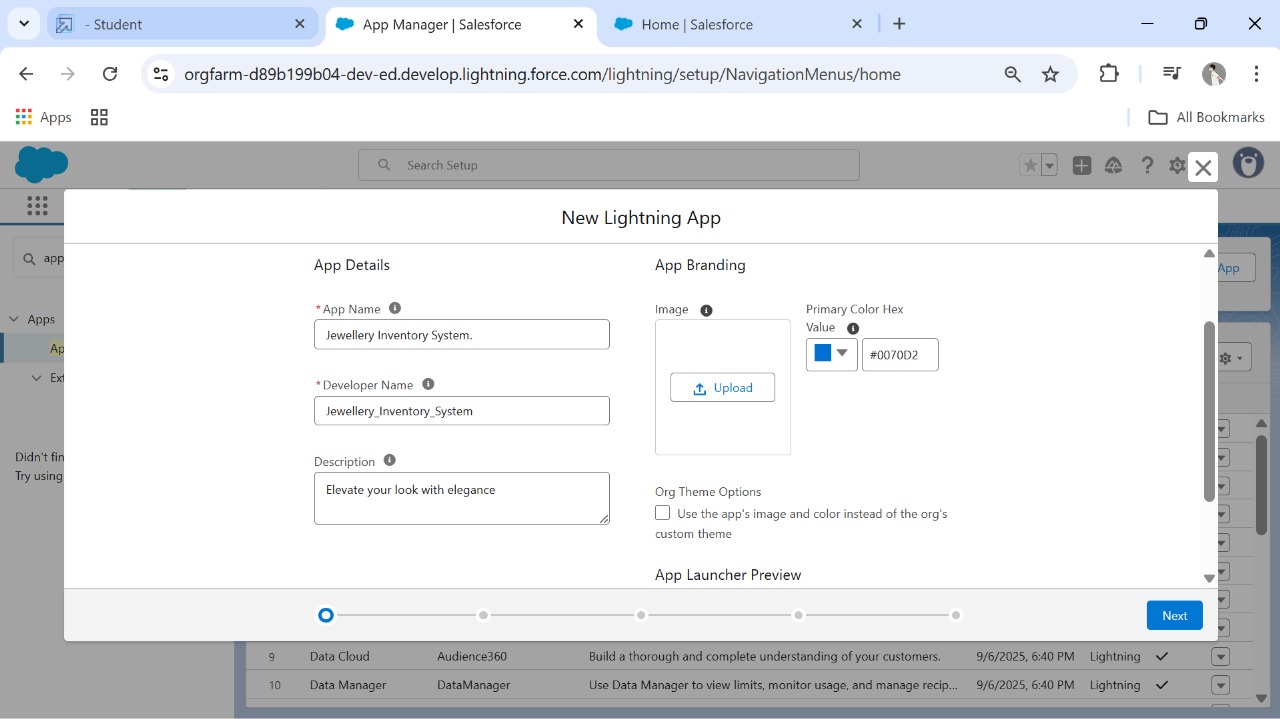


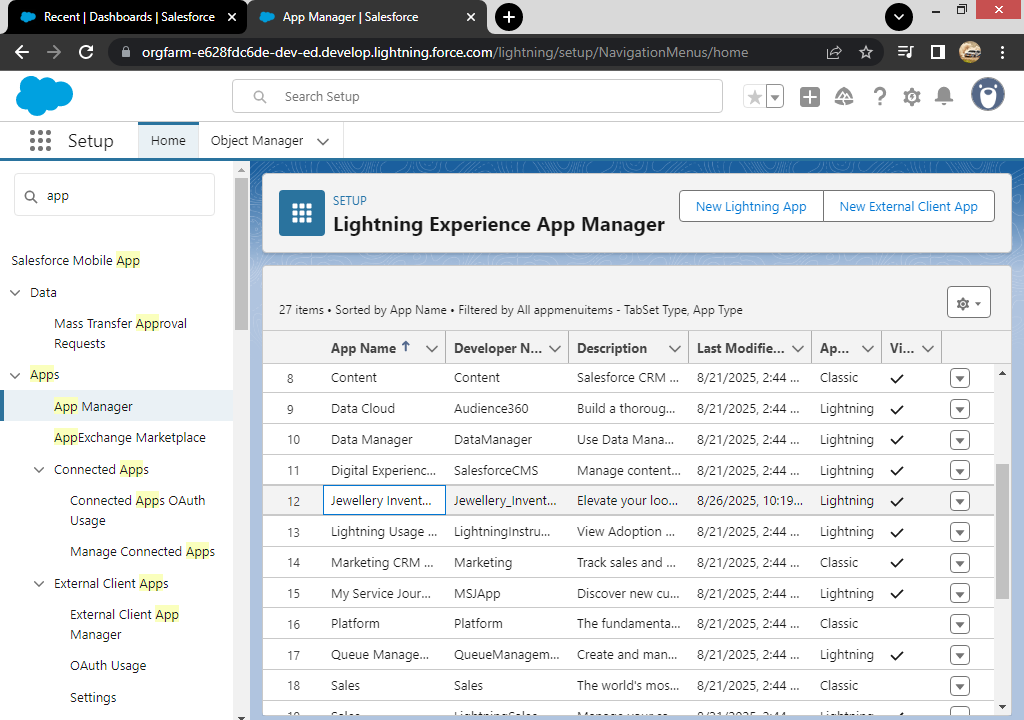




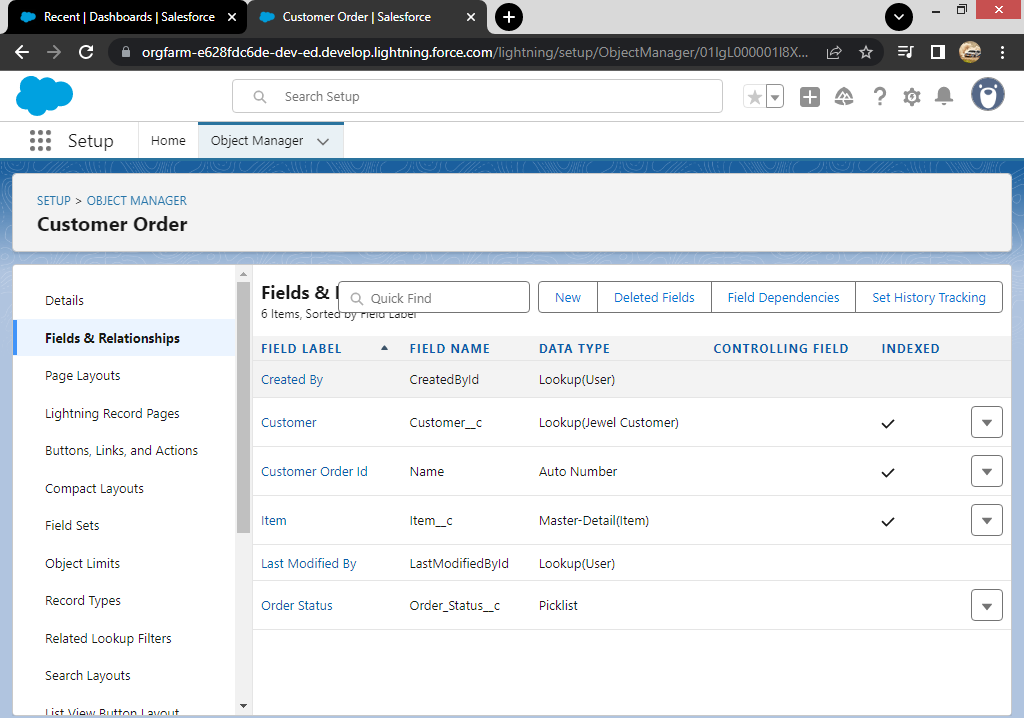
**CREATED TABS:** 

**Created A Tab:  
  
  
Created A Lightning App:**

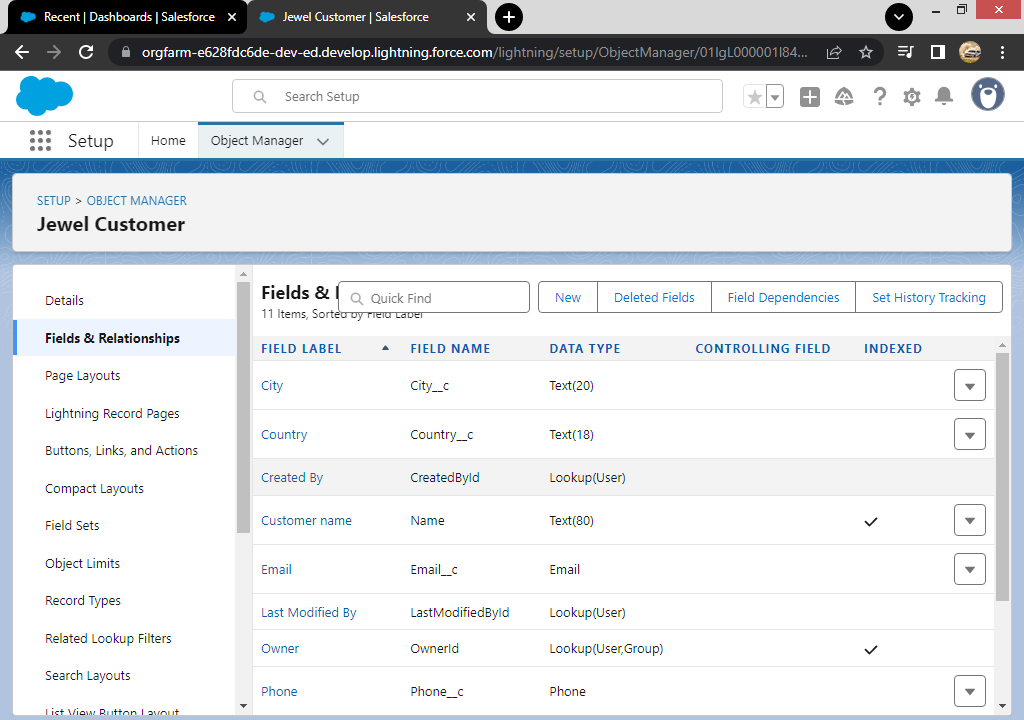
****

****

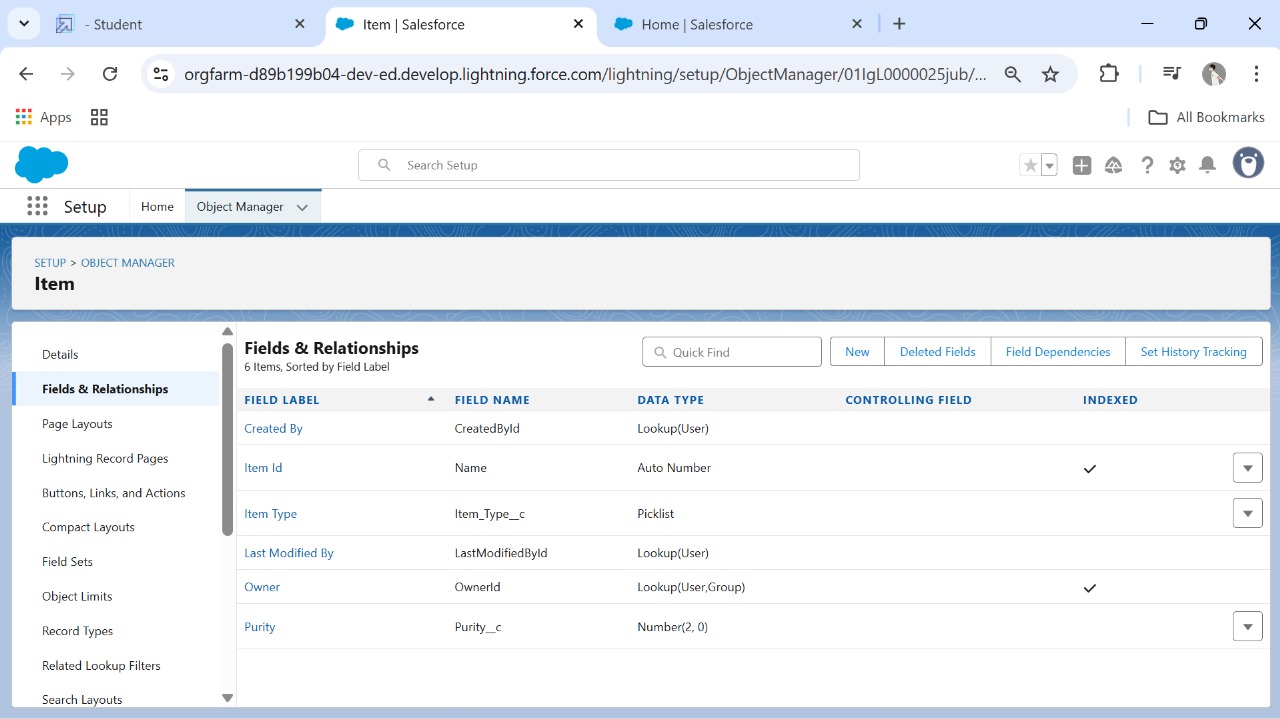
**Created Lookup Relationship:**

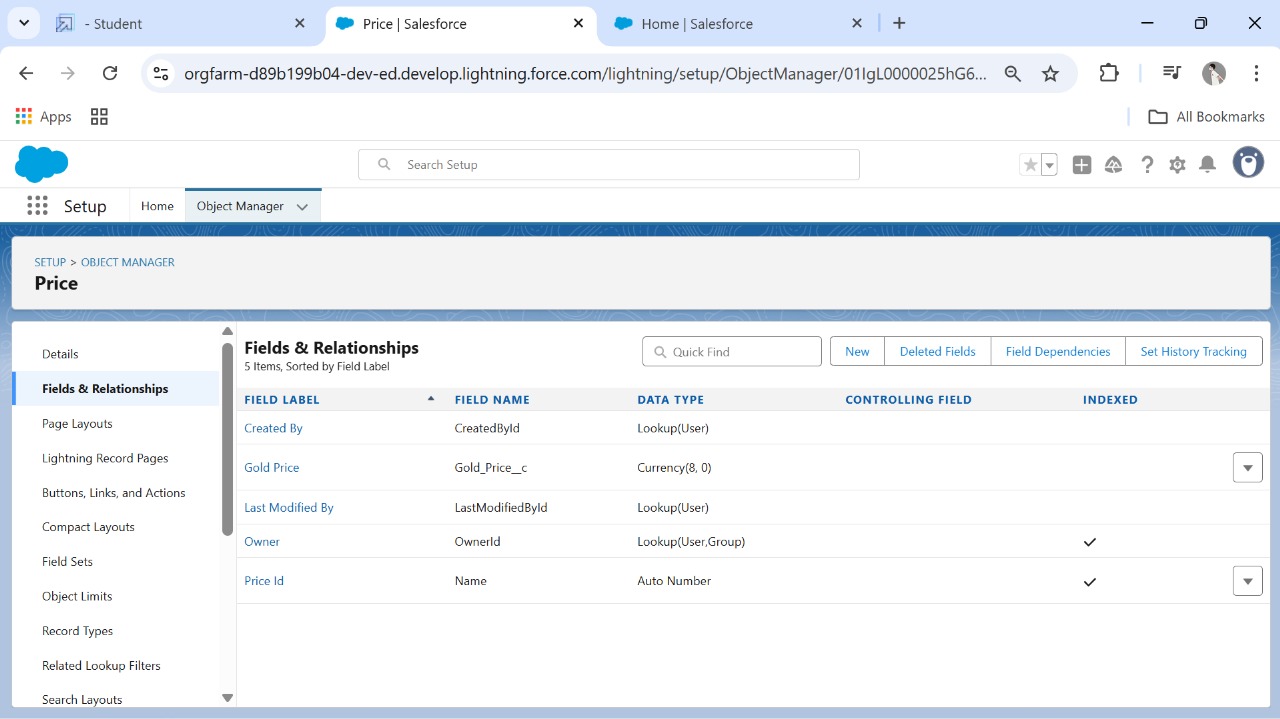


**Created Text, phone, Email Field In Jewel Customer Object:**

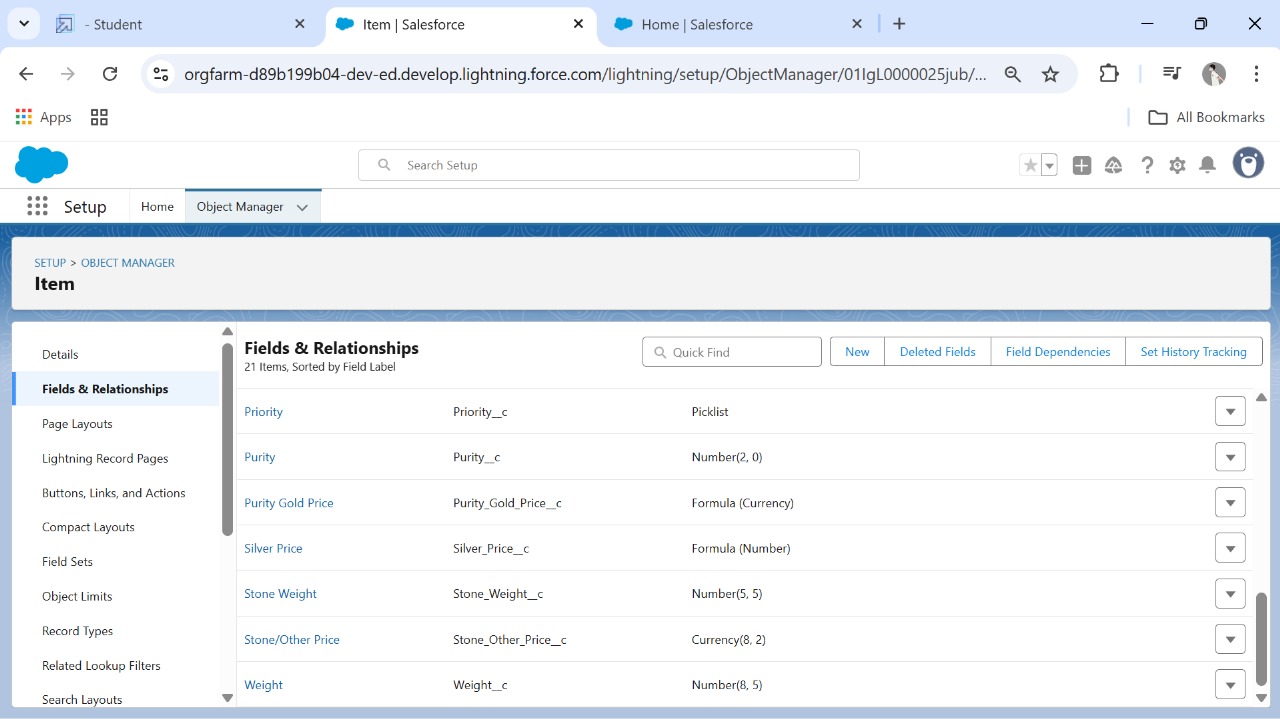
****

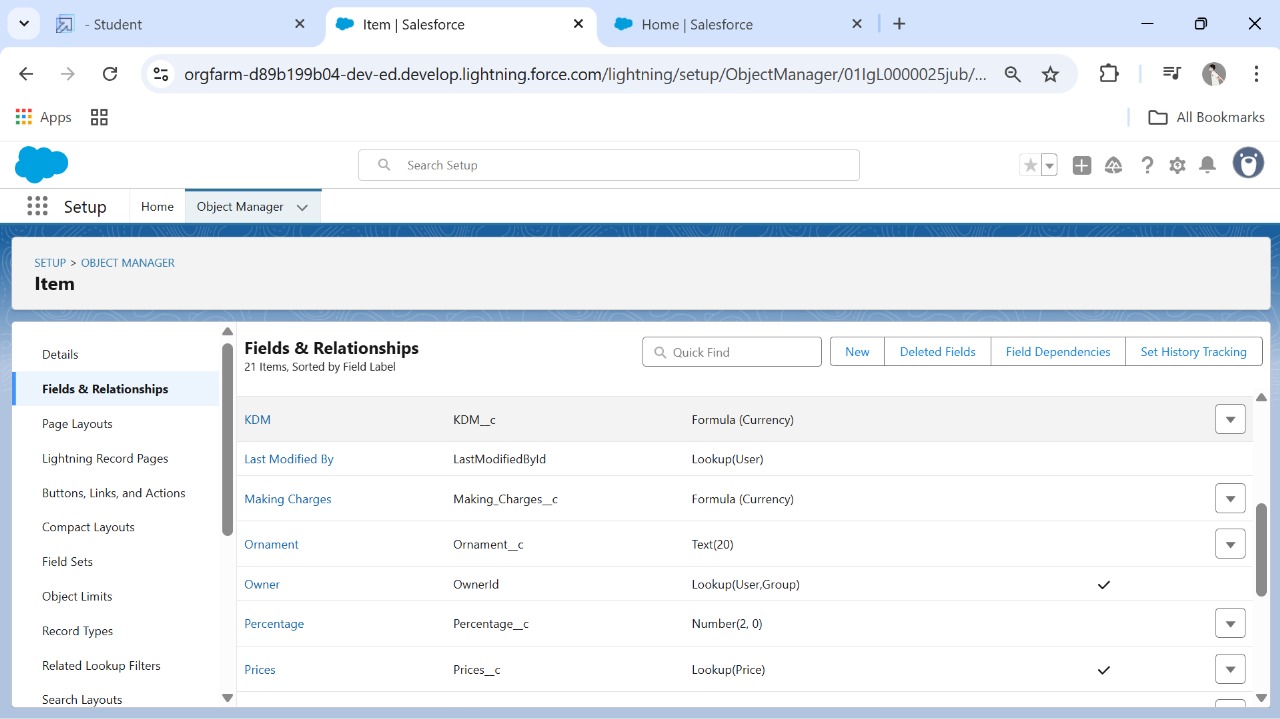
**Created The Number, picklist Field In Item Object:**

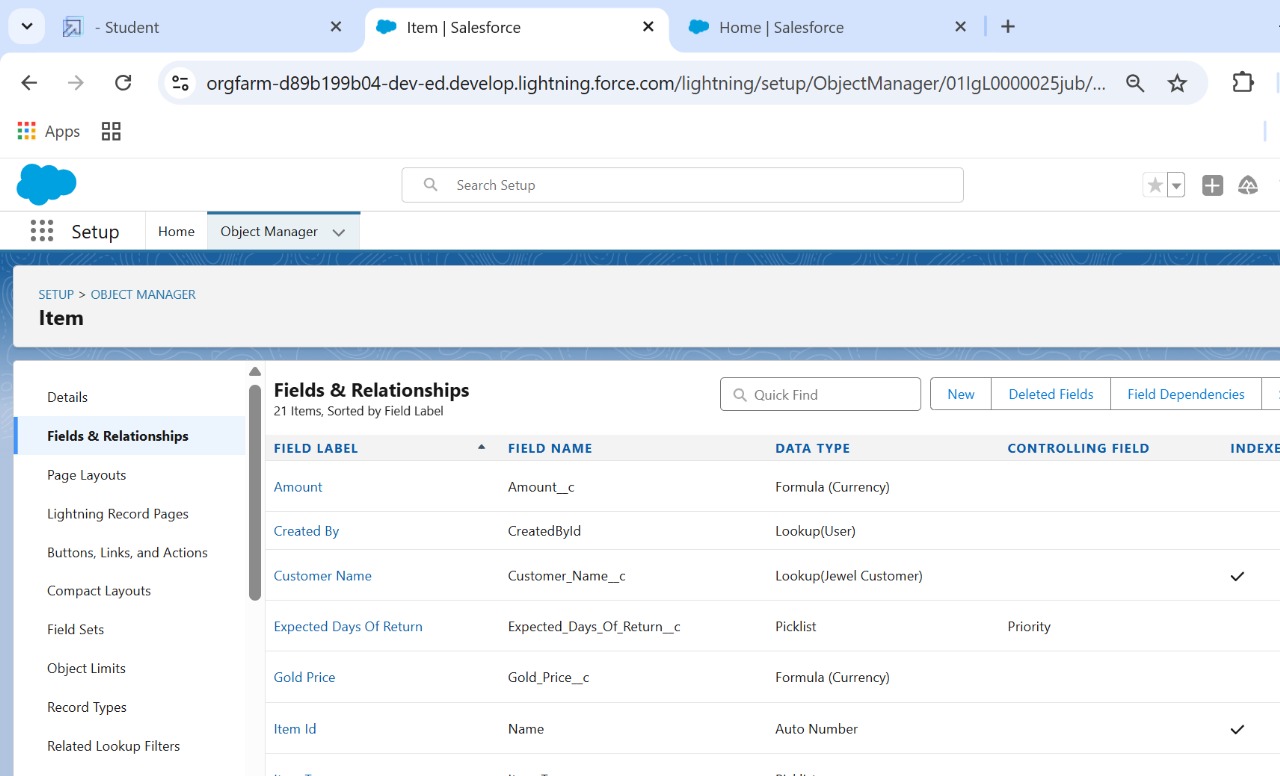
**  
  
Created Currency Field In Price Object:**

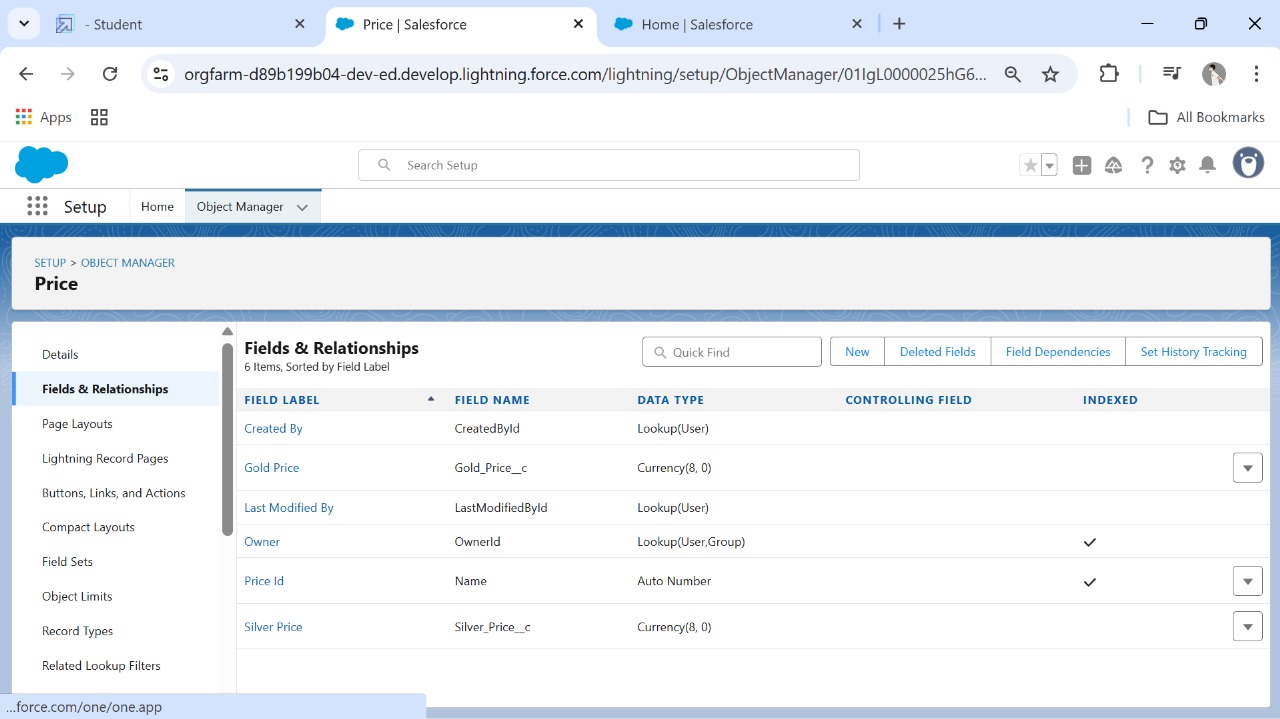
****

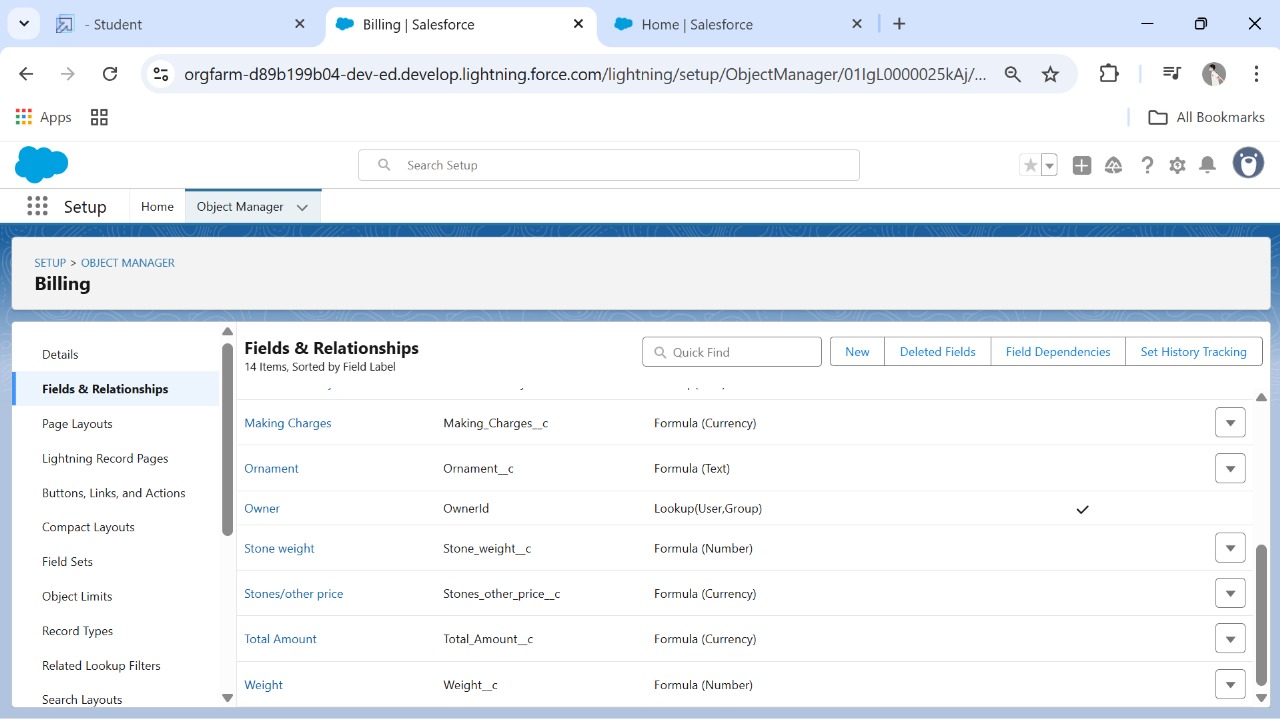
**Created Formula Fields:**

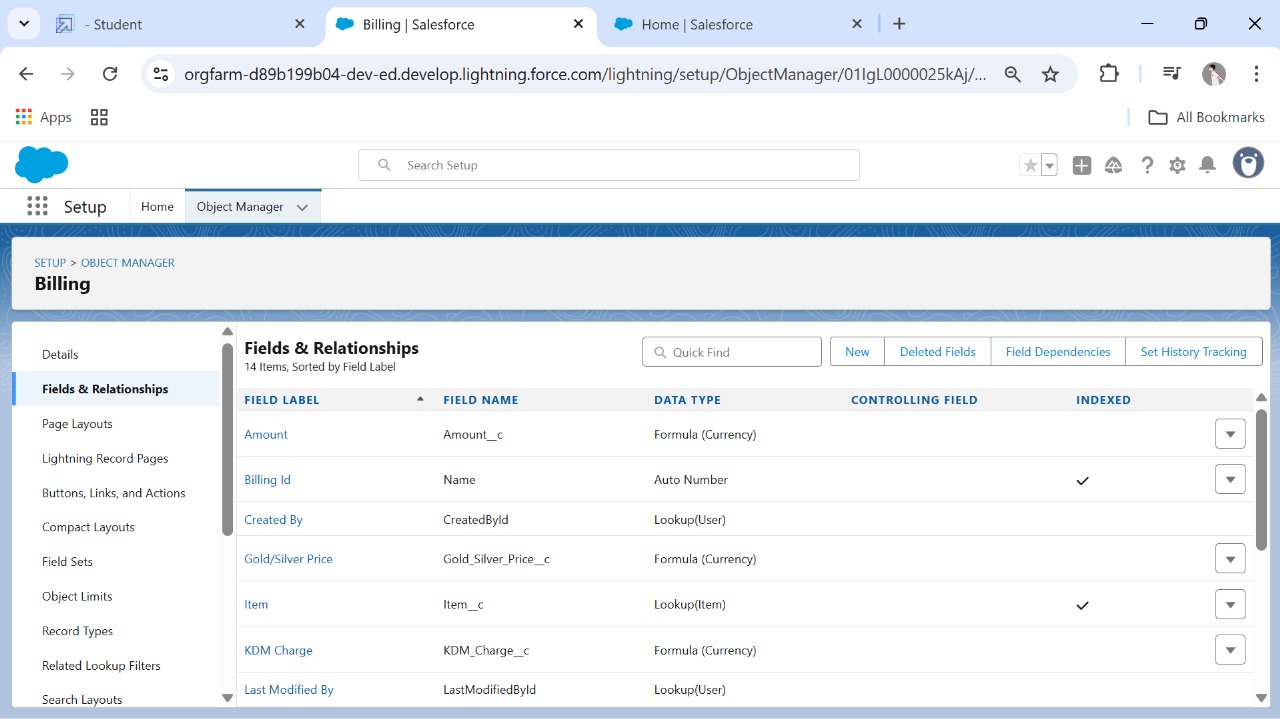


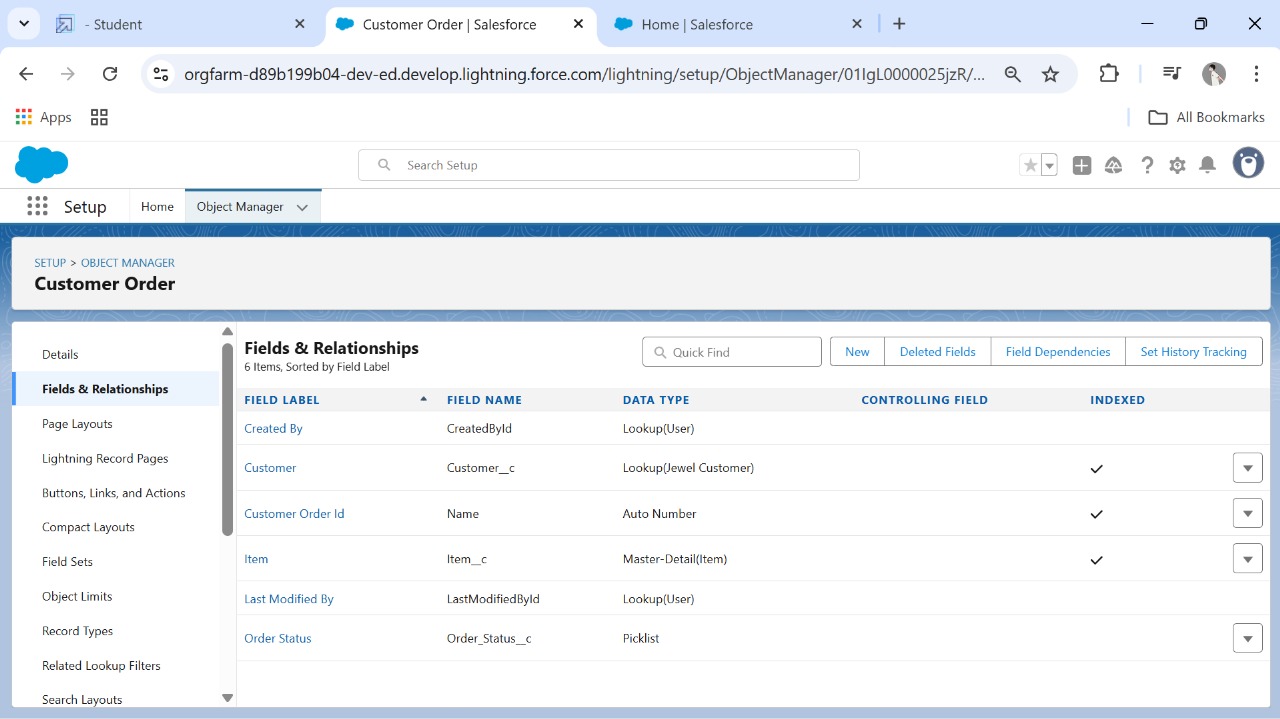




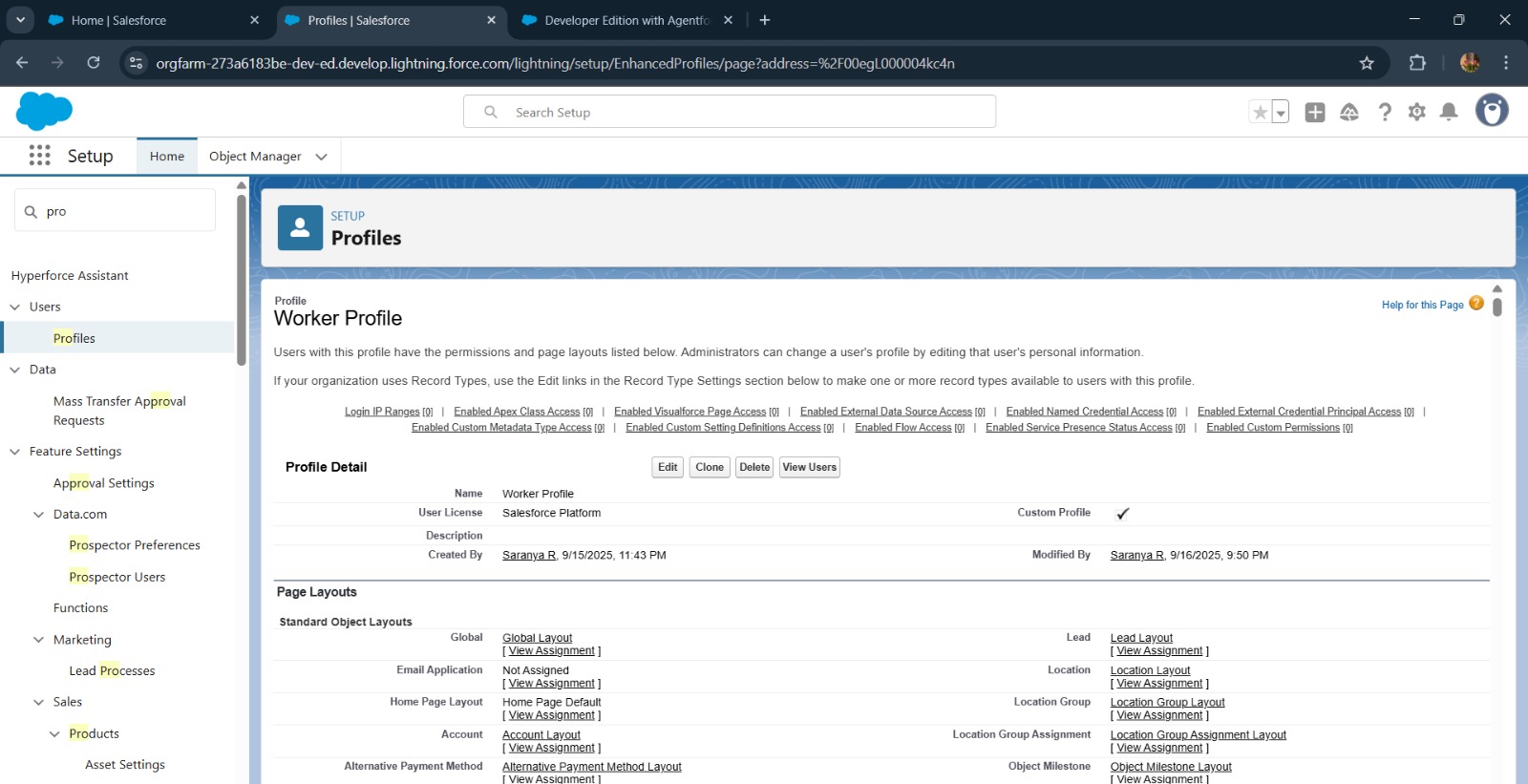








**Profiles:**

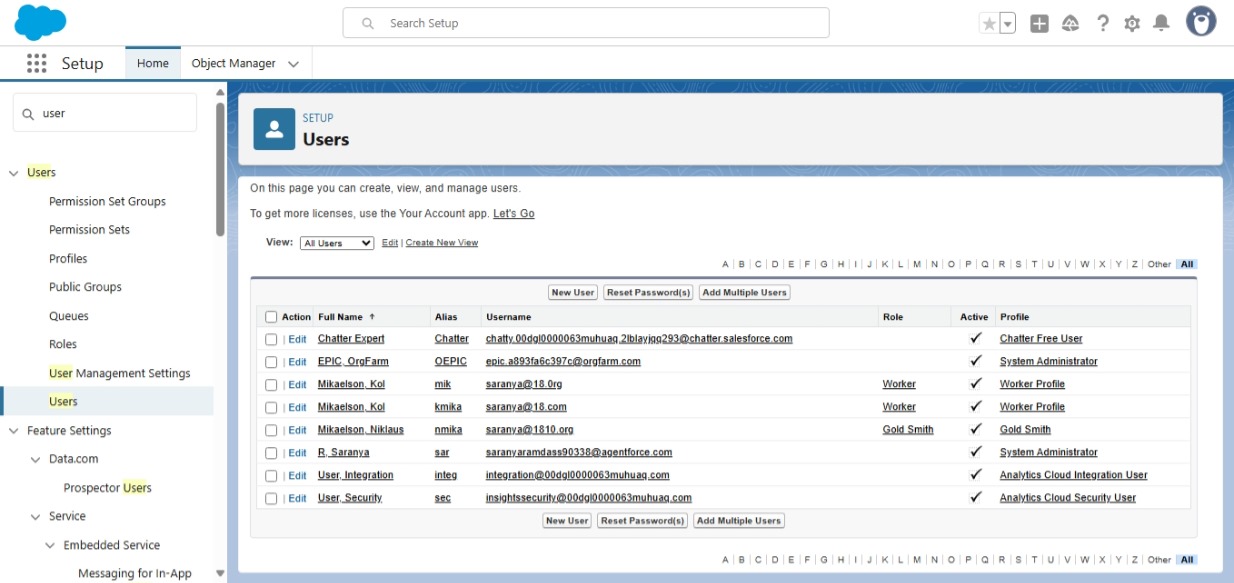


**Roles:**

A screenshot of a computer

AI-generated content may be incorrect.

**Users:**



**Page Layouts:**

A screenshot of a computer

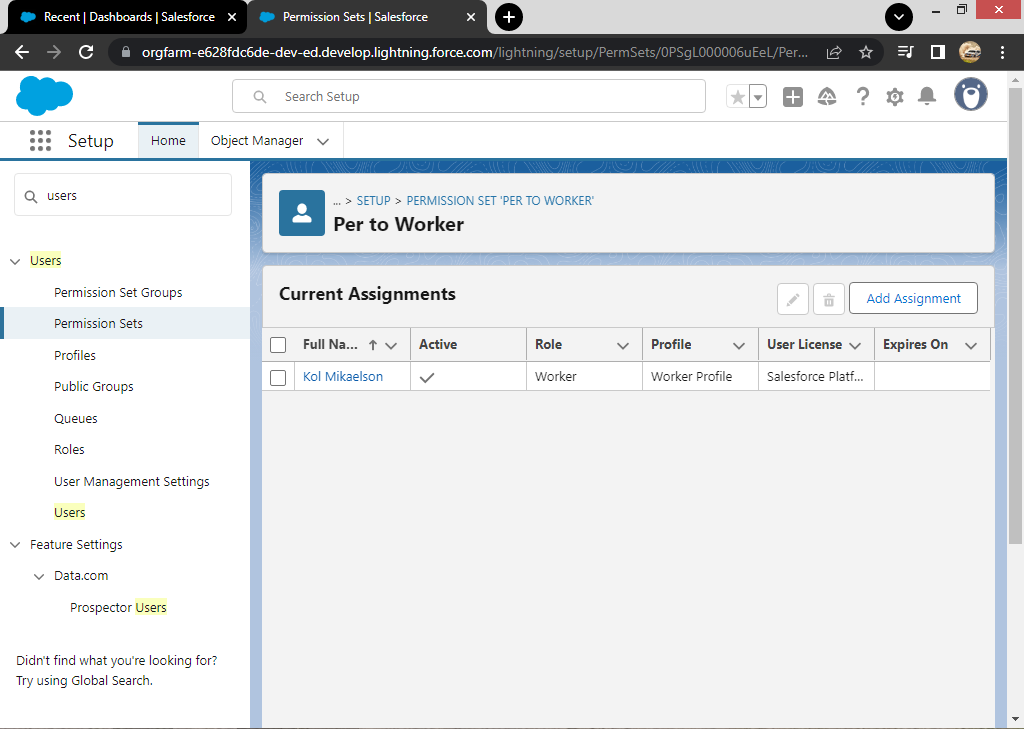
AI-generated content may be incorrect.

**Record Types:**

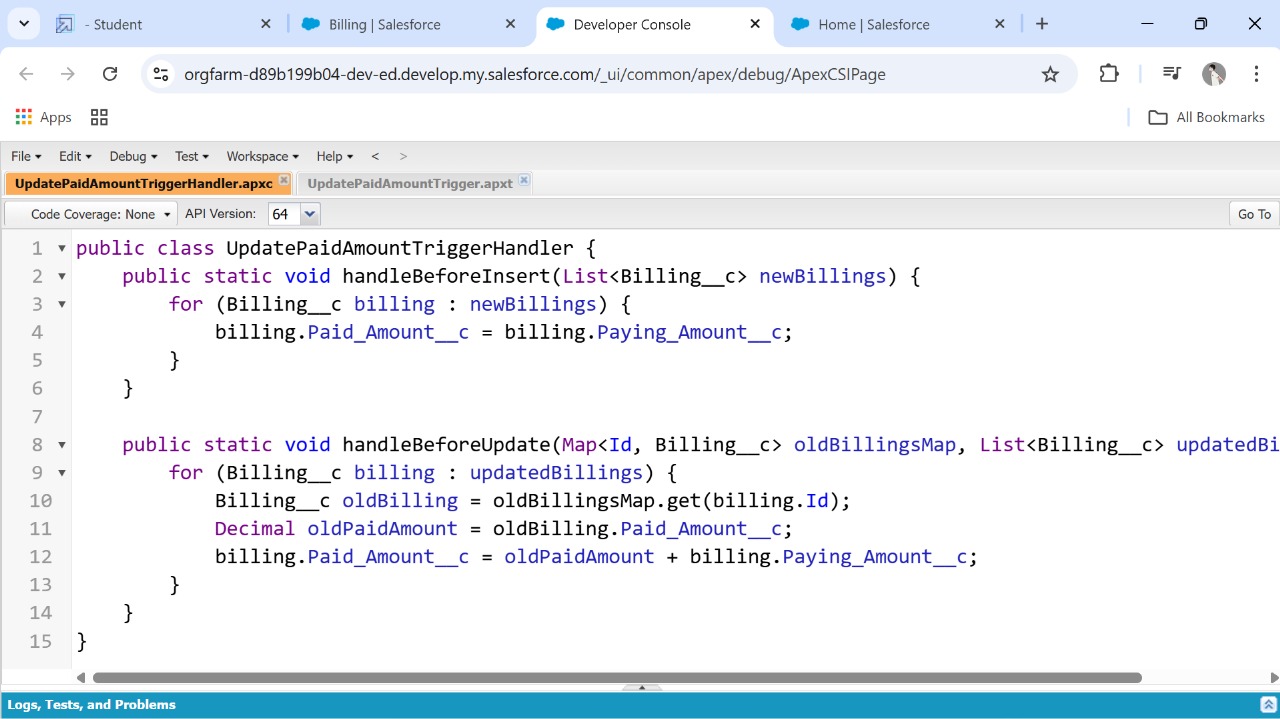
A screenshot of a computer

AI-generated content may be incorrect.

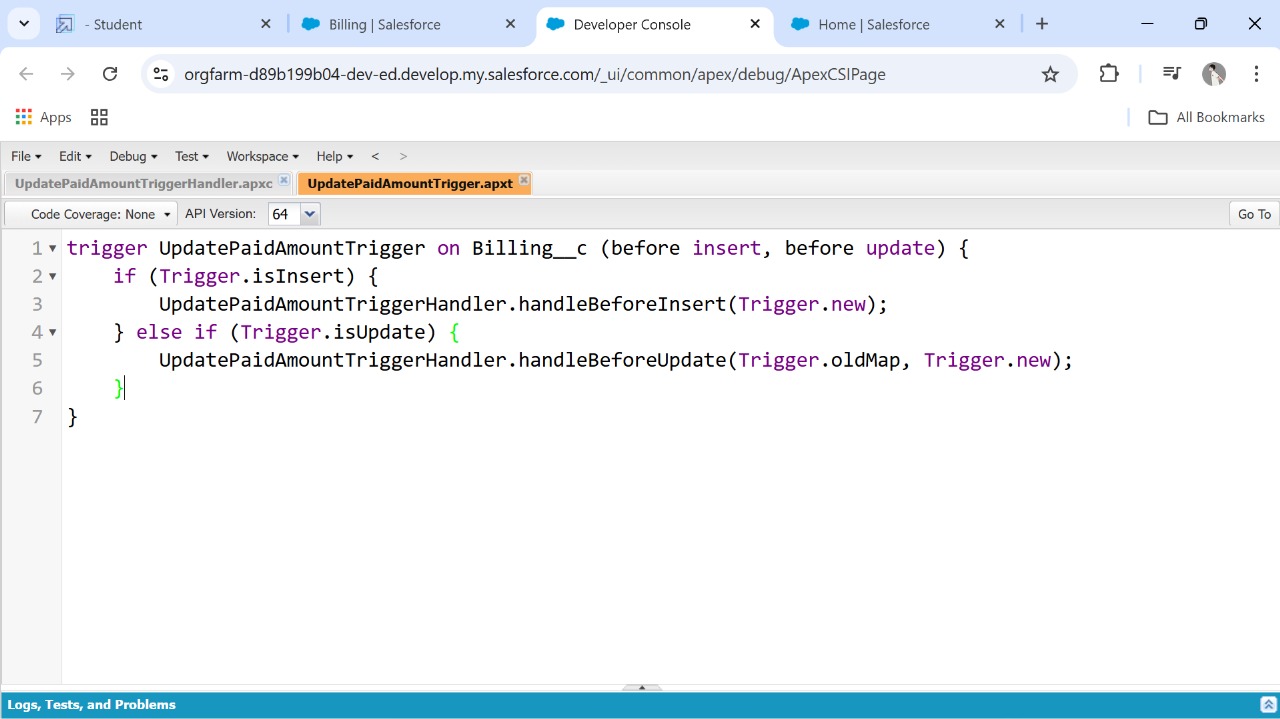
**Created Permission Set:**

****

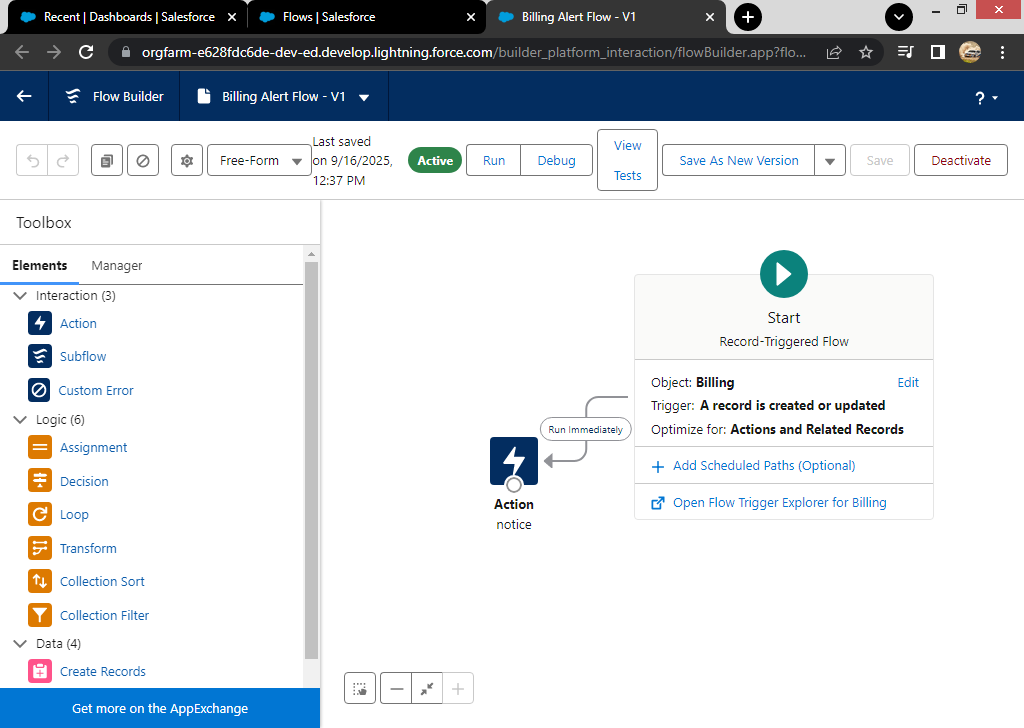
**Created A Trigger Handler Class:**

****

**Create The Trigger:**

****

**Created Flows:**

****

**CONCLUSION:**

This project leverages Salesforce CRM to provide a cloud-based jewellery management system. It integrates customer relationship management with sales and inventory, helping jewellery businesses improve efficiency, profitability, and customer engagement.  
 **APPENDIX:  
  
Source Code:** Provided in Apex Classes and Triggers  
  
**UpdatePaidAmountTriggerHandler.apxc:**  
  
public class UpdatePaidAmountTriggerHandler {

public static void handleBeforeInsert(List<Billing\_\_c> newBillings) {

for (Billing\_\_c billing : newBillings) {

billing.Paid\_Amount\_\_c = billing.Paying\_Amount\_\_c;

}

}

public static void handleBeforeUpdate(Map<Id, Billing\_\_c> oldBillingsMap, List<Billing\_\_c> updatedBillings) {

for (Billing\_\_c billing : updatedBillings) {

Billing\_\_c oldBilling = oldBillingsMap.get(billing.Id);

Decimal oldPaidAmount = oldBilling.Paid\_Amount\_\_c;

billing.Paid\_Amount\_\_c = oldPaidAmount +   
  
billing.Paying\_Amount\_\_c;

}

}

}  
  
**UpdatePaidAmountTrigger.apxt:**trigger UpdatePaidAmountTrigger on Billing\_\_c (before insert, before update) {

if (Trigger.isInsert) { UpdatePaidAmountTriggerHandler.handleBeforeInsert(Trigger.new);

} else if (Trigger.isUpdate) { UpdatePaidAmountTriggerHandler.handleBeforeUpdate(Trigger.oldMap, Trigger.new);

}

}