**CRM APPLICATION FOR JEWEL MANAGEMENT (DEVELOPER)**

****

**College Name:**

**Tiruppur Kumaran College For Women**

**College Code:**

**bru3j**

**TEAM ID: NM2025TMID28436**

**TEAM MEMBERS:**

* **Team Leader Name: VAISHALA V**

**Email: ammuvaishalaammu@gmail.com**

* **Team Member 1: KIRUTHIKA B**

**Email**: bkiruthika105@gmail.com

* **Team Member 2: PRIYADHARSHINI J**

**Email:** pd408435@gmail.coms

* **Team Member 3: MAHALAKSHMI TL**

**Email:** mahalakshmi960021@gmail.com

## Introduction

The Jewel Inventory System is a software solution designed to help jewellery stores and manufacturers manage their inventory and sales. This system makes it easy to track different jewellery items, maintain accurate records, and handle sales smoothly. The main goal is to provide a simple and efficient way to organize jewellery business operations.

### Salesforce

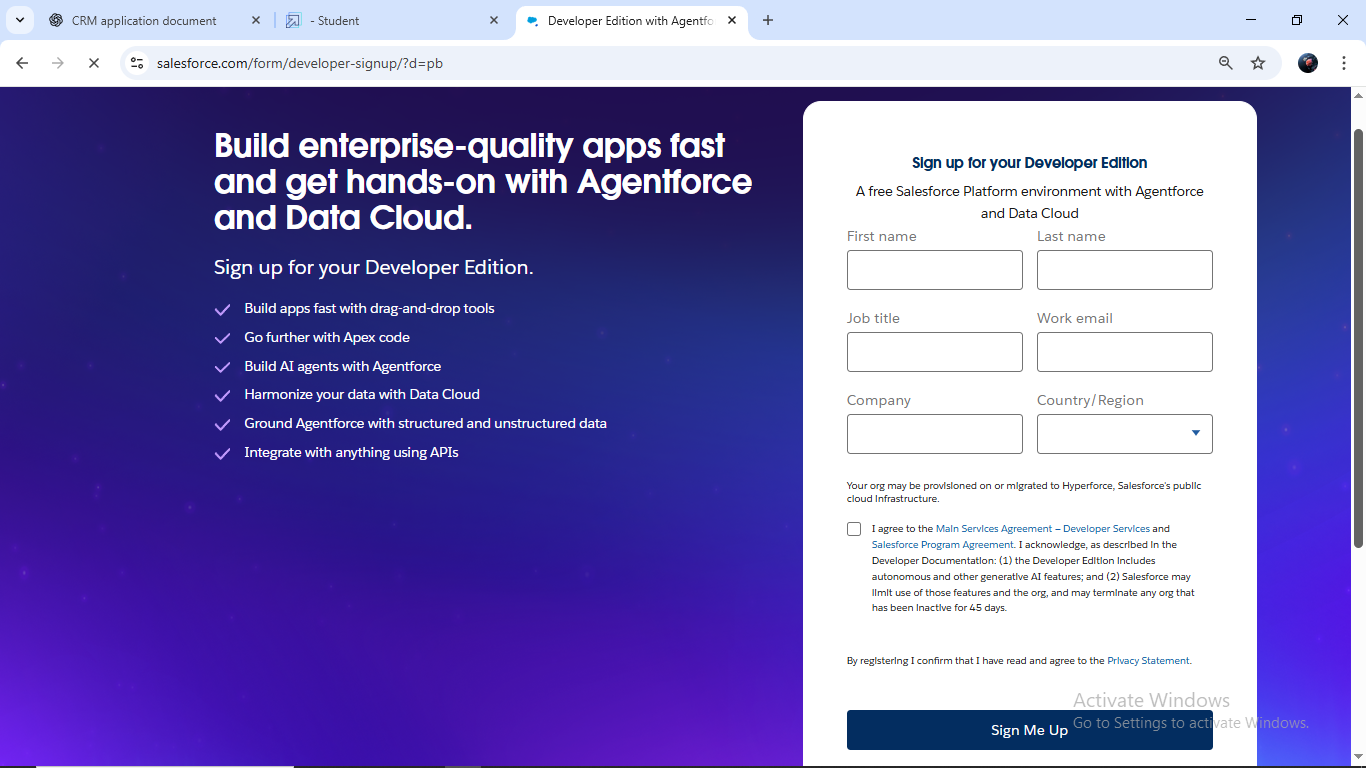
* Salesforce is your customer success platform, designed to help you sell, service, market, analyze, and connect with your customers.

## 1.1 Creating Developer Account

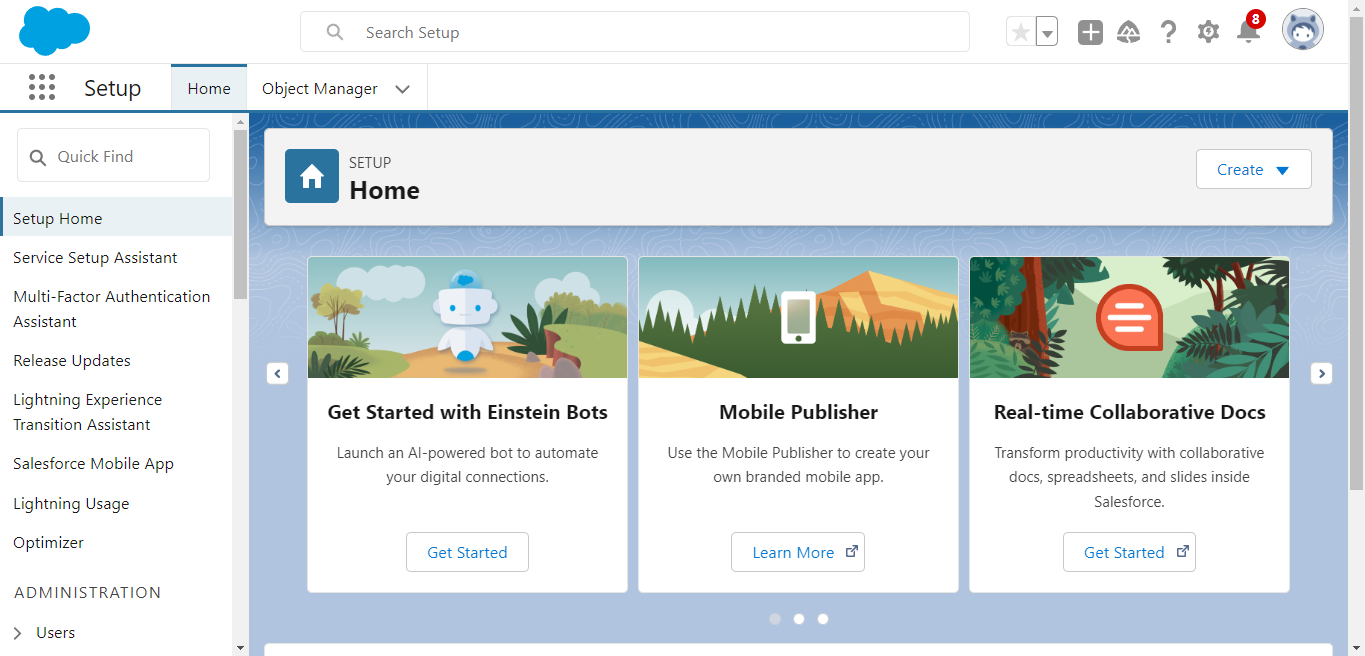
To start working on the CRM Application for Jewel Management using Salesforce, you first need to create a **Salesforce Developer Account**.

### Steps to Create Developer Account:

1. Go to: <https://developer.salesforce.com/signup>



1. Fill in the sign-up form with the following details:
   * First Name & Last Name
   * Email
   * Role: Developer
   * Company: College Name
   * Country: India
   * Postal Code: Pin Code
   * Username: A combination of your name and company (format: [username@organization.com](mailto:username@organization.com))
   * After filling in the details, click on **Sign Me Up**.

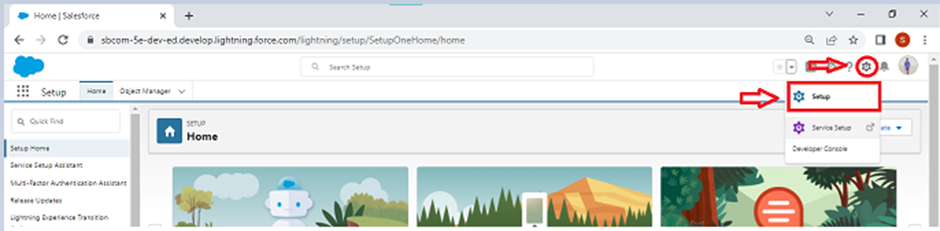


### 2.Object

# What Is an Object?

Salesforce objects are database tables that permit you to store data that is specific to an organization. What are the types of Salesforce objects

.

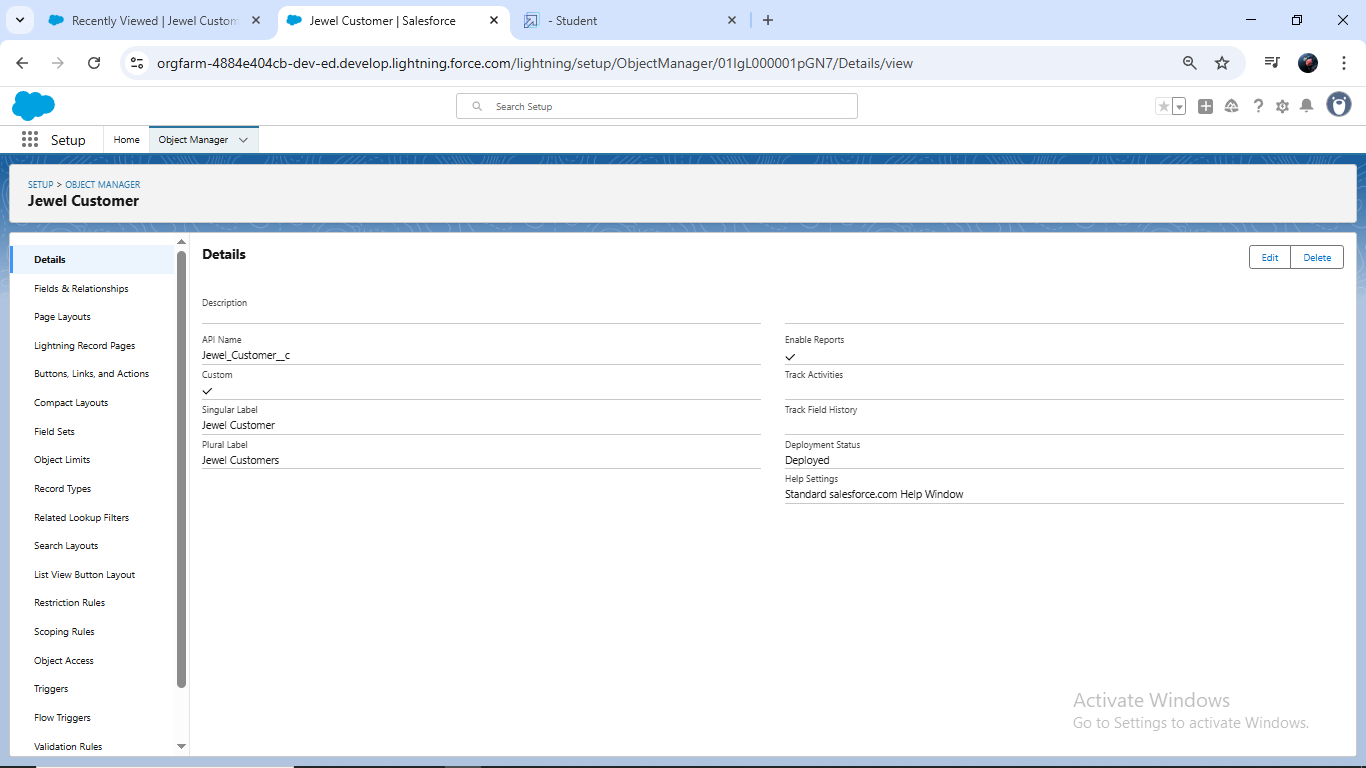


### 2.1 Create Jewel Customer Object

The purpose of creating a Jewel Customer custom object is to store and manage information about Customer.

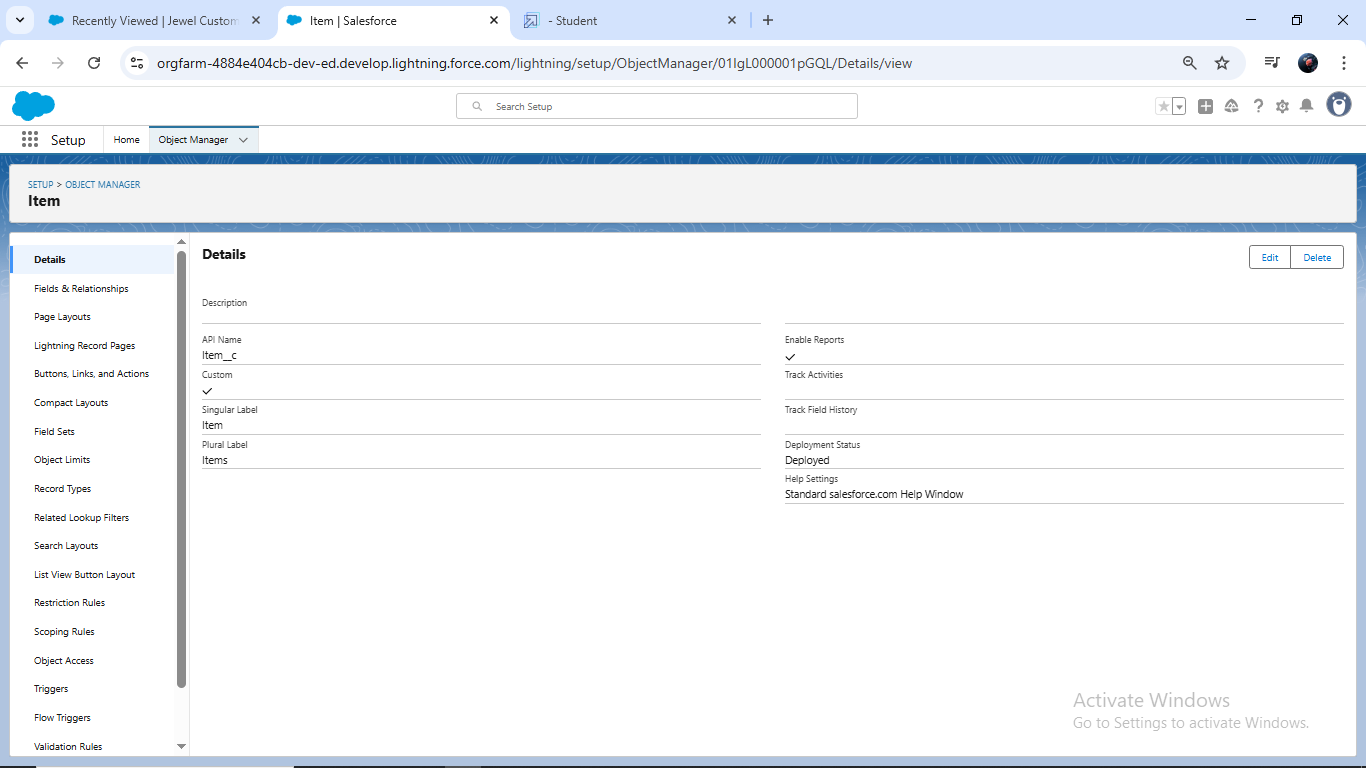
.

1. Enter the label name : Jewel Customer
2. Plural label name : Jewel Customers



### 2.2 Create Item Object

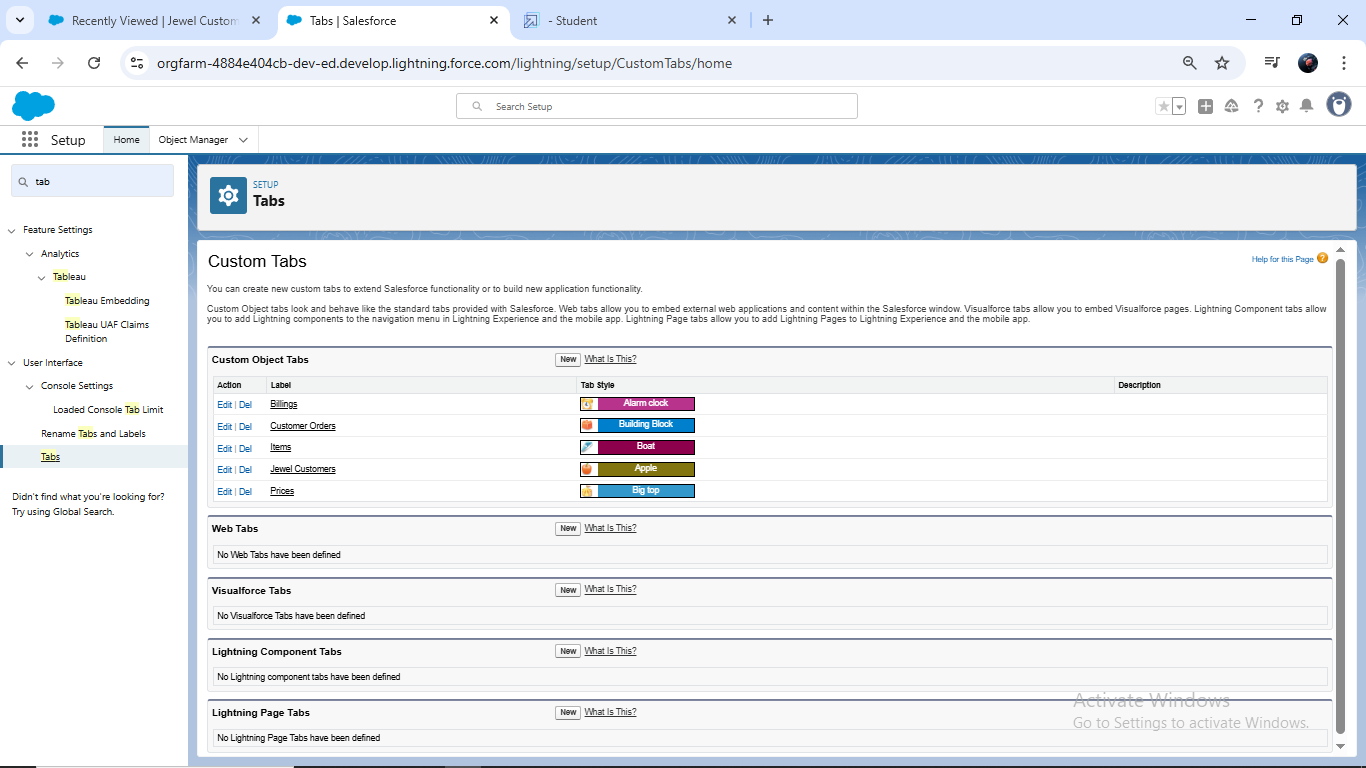
* The purpose of creating a Item object is to manage the inventory of gold and silver items.



### 3.Tabs

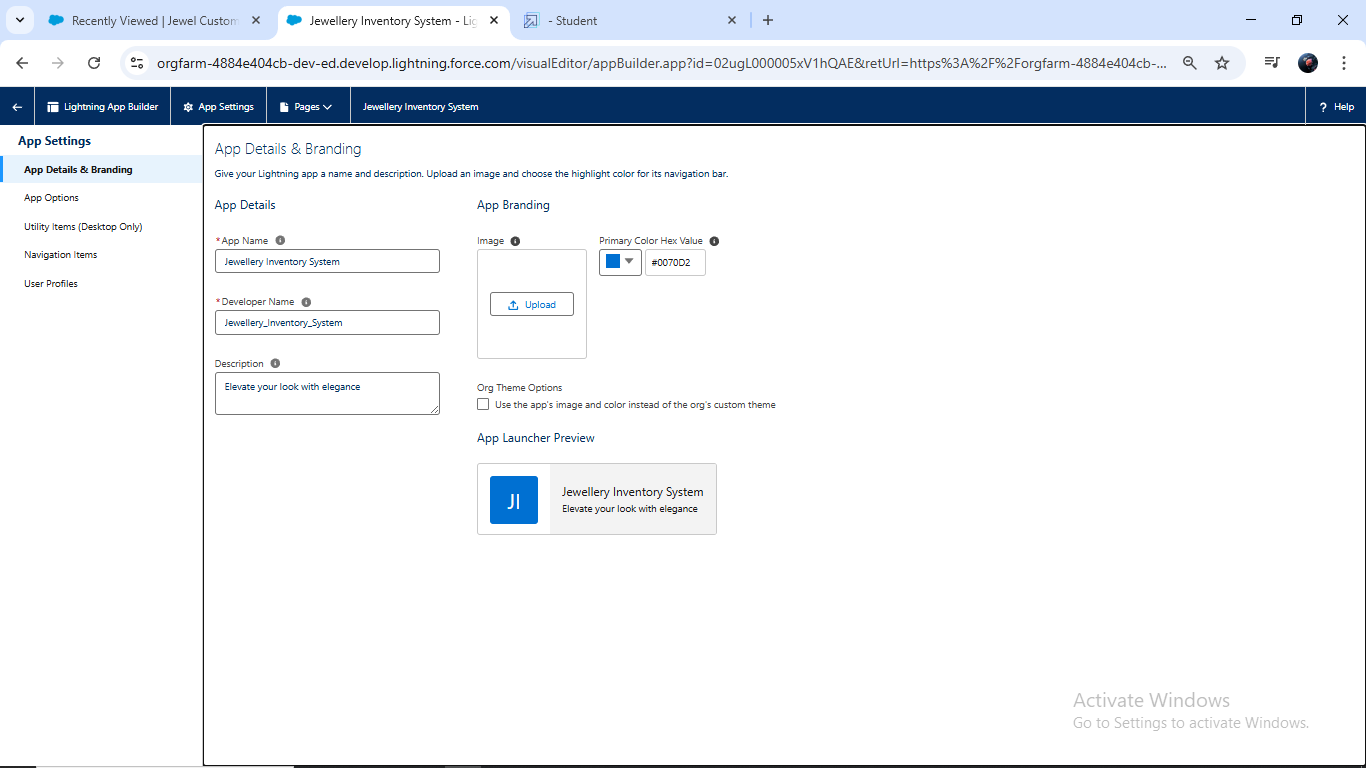
**What is Tab:**A tab is like a user interface that is used to build records for objects and to view the records in the objects.

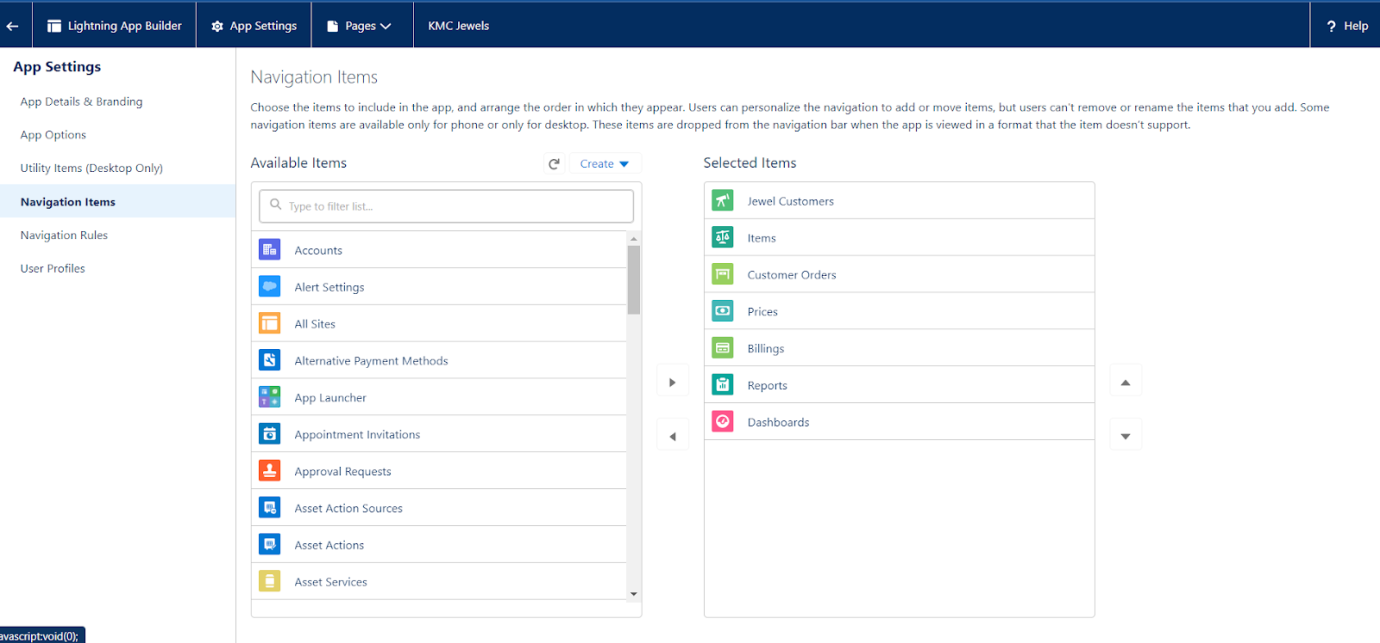
### Creating a Custom Tab

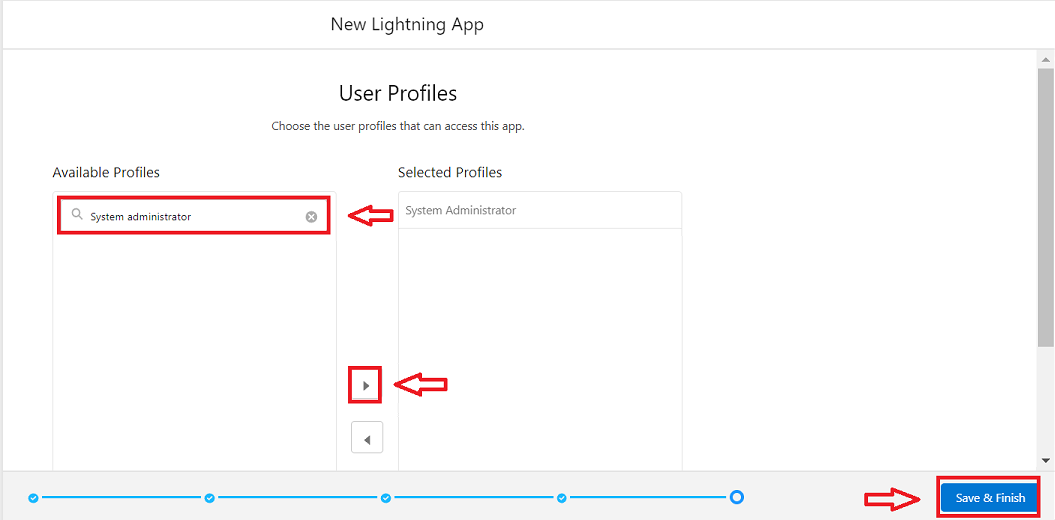


### 4.The Lightning App

* An app is a collection of items that work together to serve a particular function.

4.1 Create a Lightning App





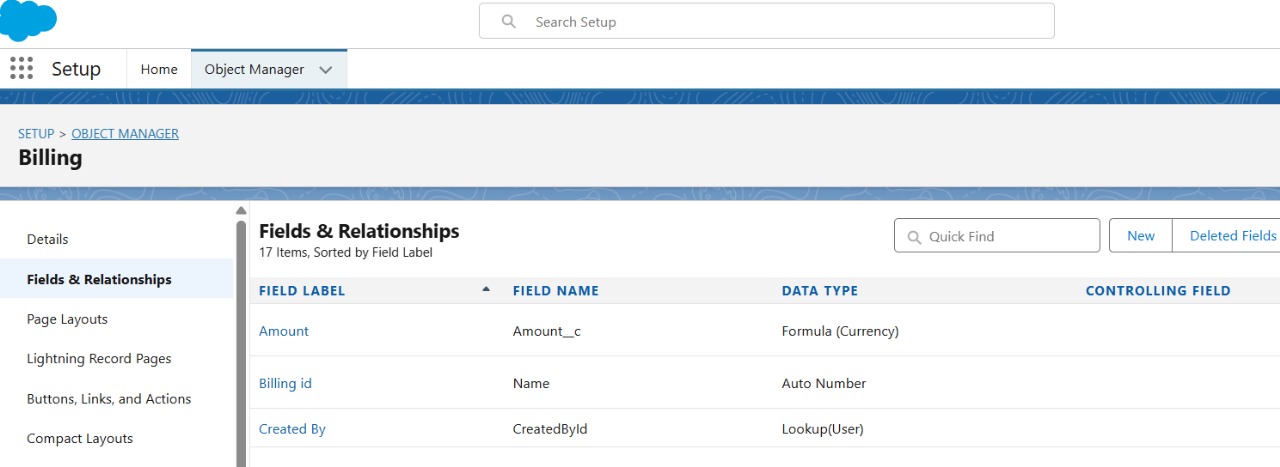
.

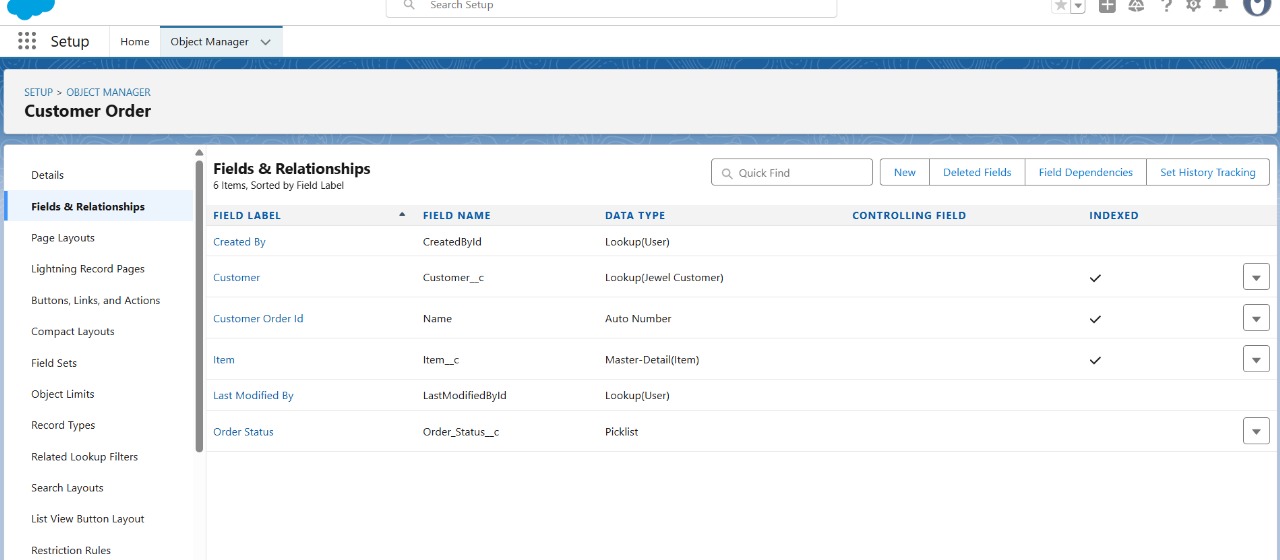
### Fields

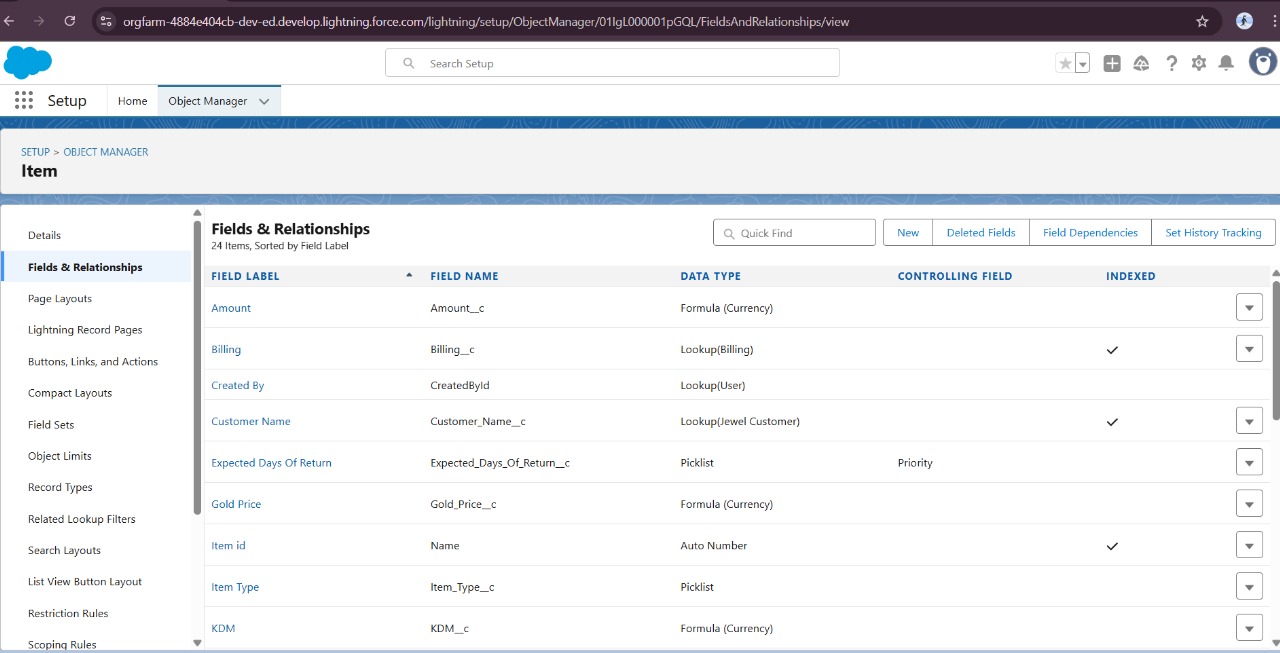
* When we talk about Salesforce, Fields represent the data stored in the columns of a relational database.

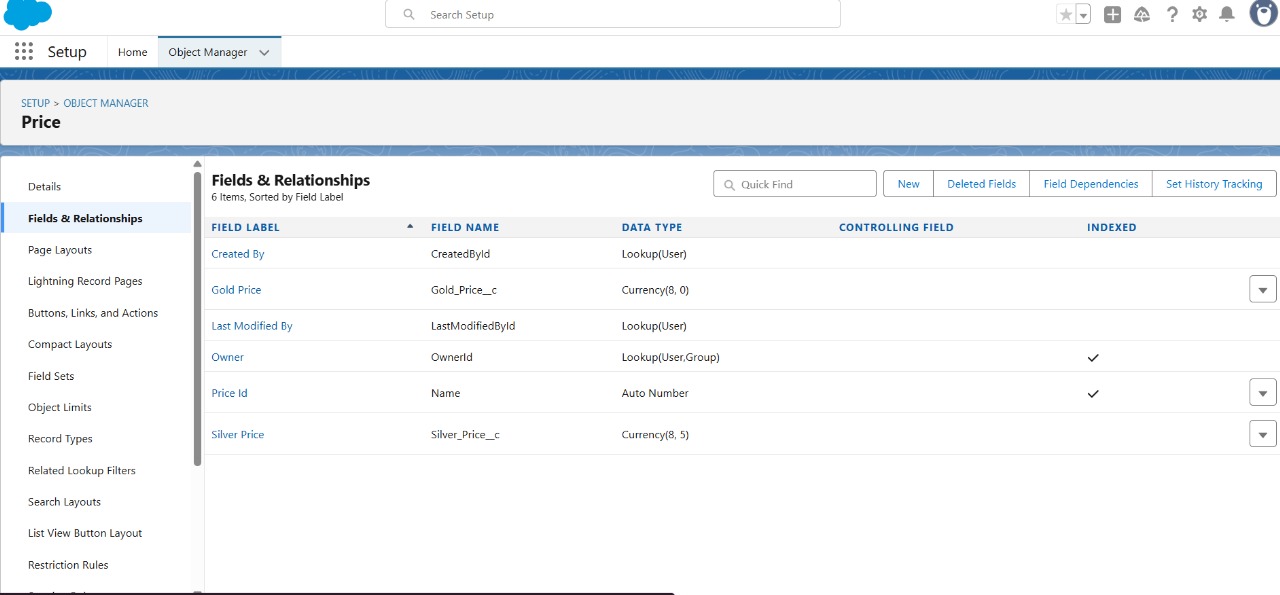
##### Types of Fields :

1. Standard Fields
2. Custom Fields









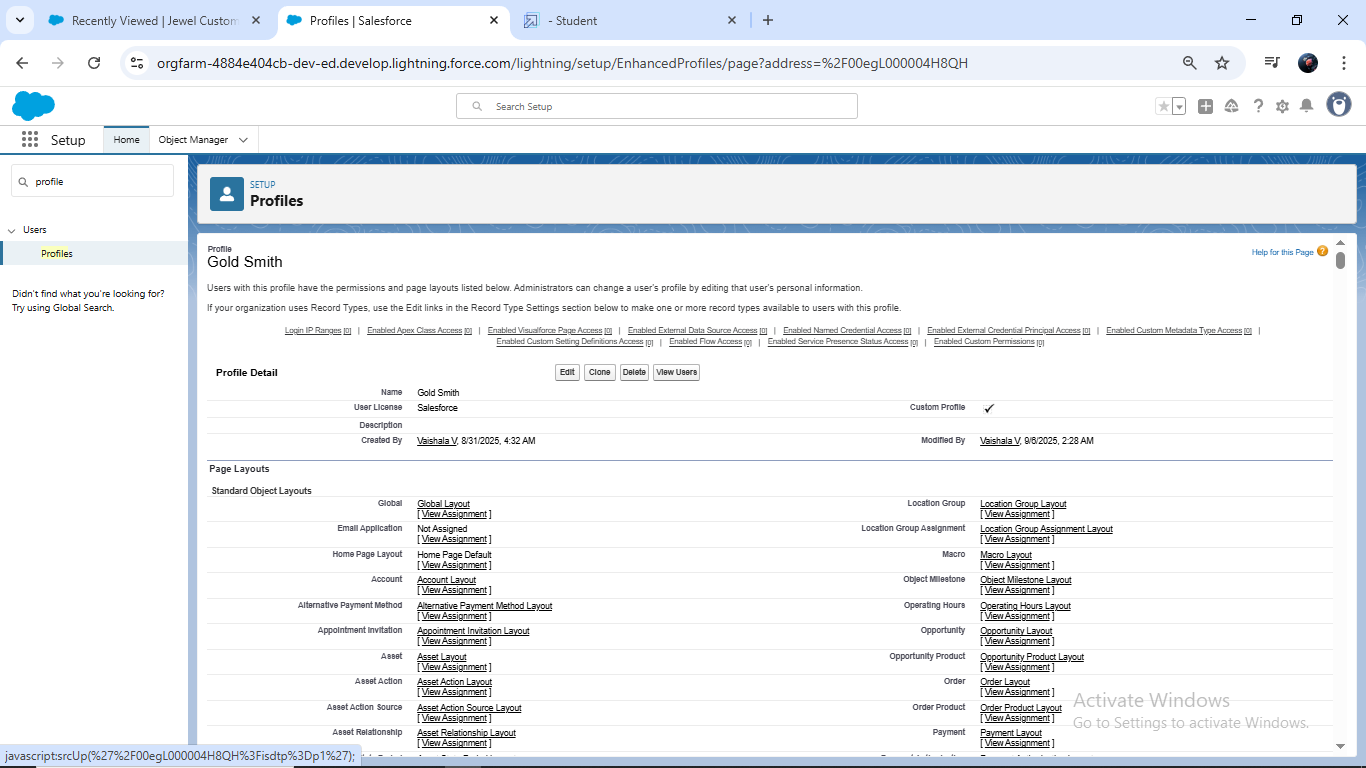


### https://lh7-rt.googleusercontent.com/docsz/AD_4nXdcVQ6IwYTw7y3XbFbuZGnrKhojK9oAdxL03Wjo4gF9qWwIUxABD0kfYUMFSygbkV5n6ptNEcUh4OsDK6UKuP2FlPtGwUlmRj8Ty02N9tSef8bLH9kDy5JwZsMqw9OCwZ5TbcQX?key=dfLmaqUsnVoHOKcSZi4NZA

### Profiles

* A profile is a group/collection of settings and permissions that define what a user can do in salesforce

### 6.1 Gold Smith Profile



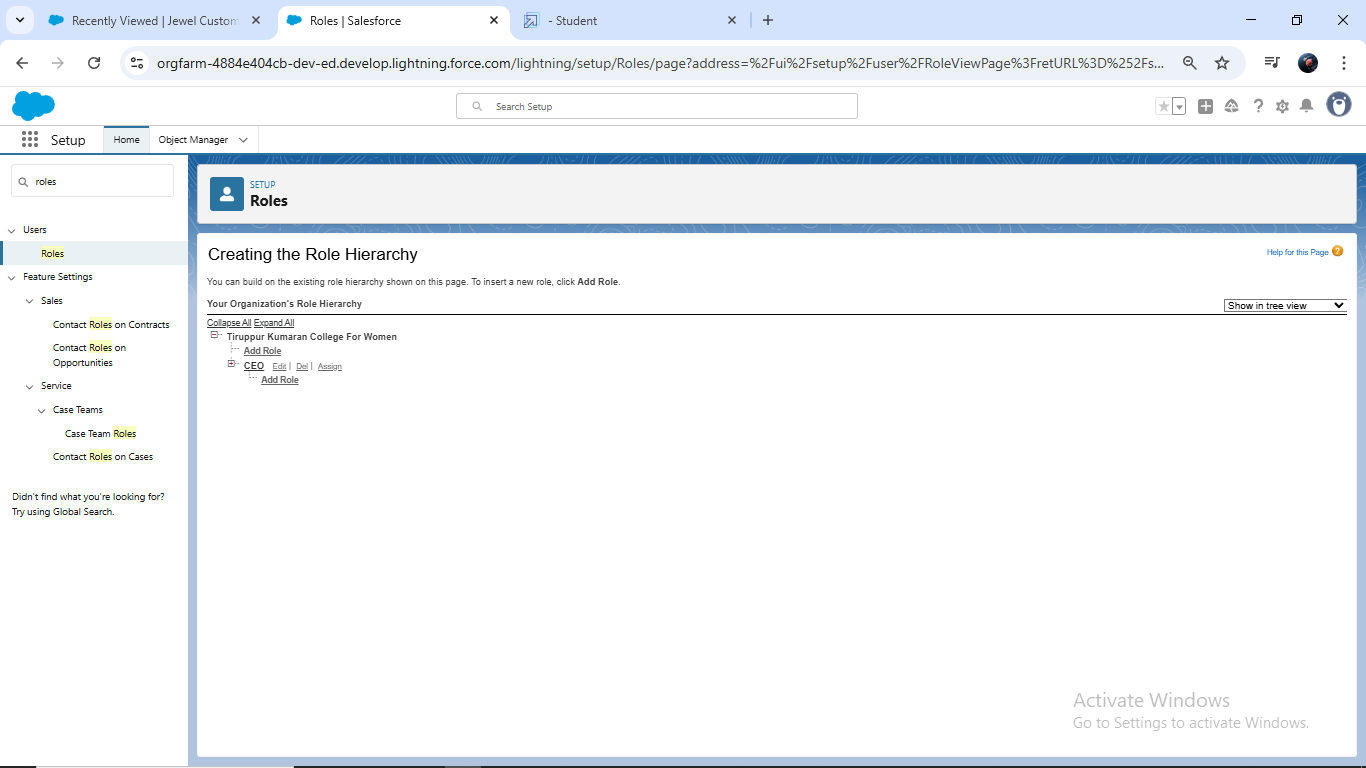
### 6.2 Worker Profile

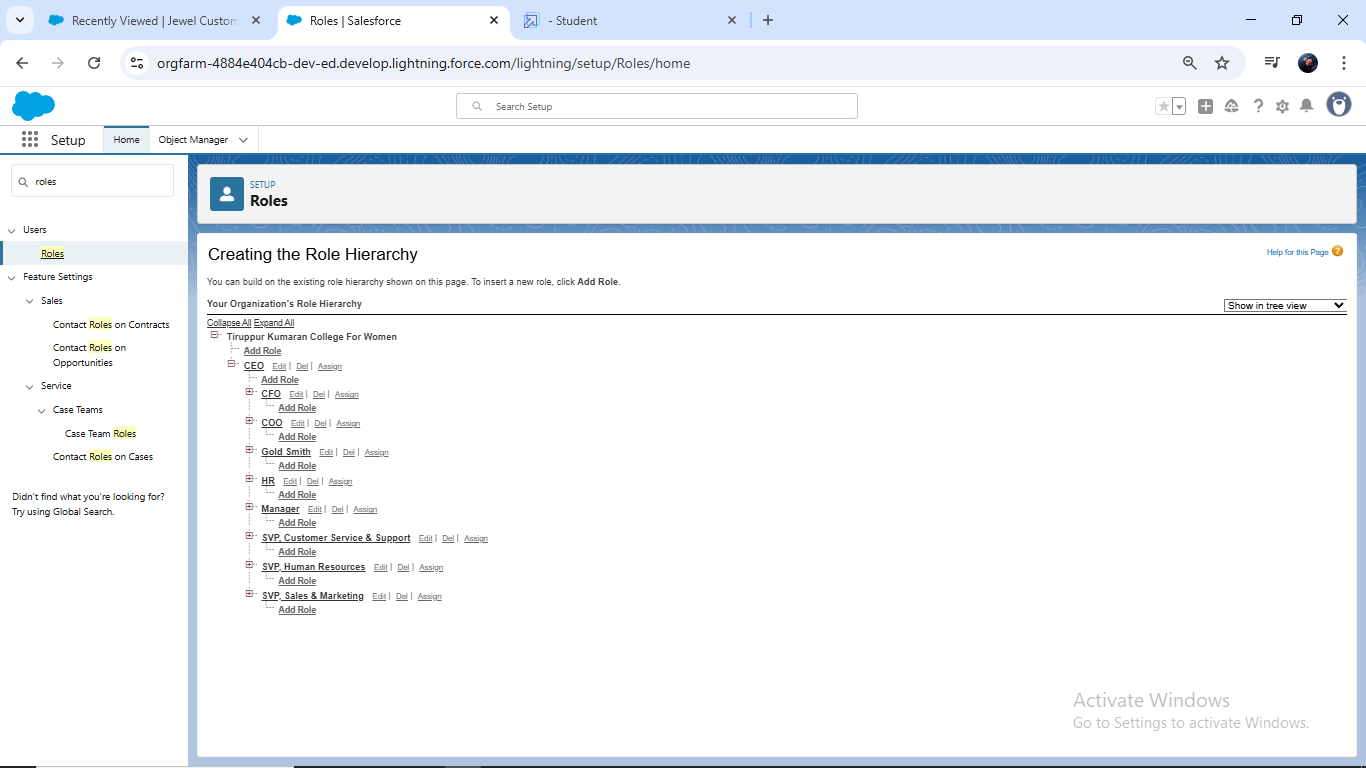


### Roles

* A role in Salesforce defines a user's visibility access at the record level.

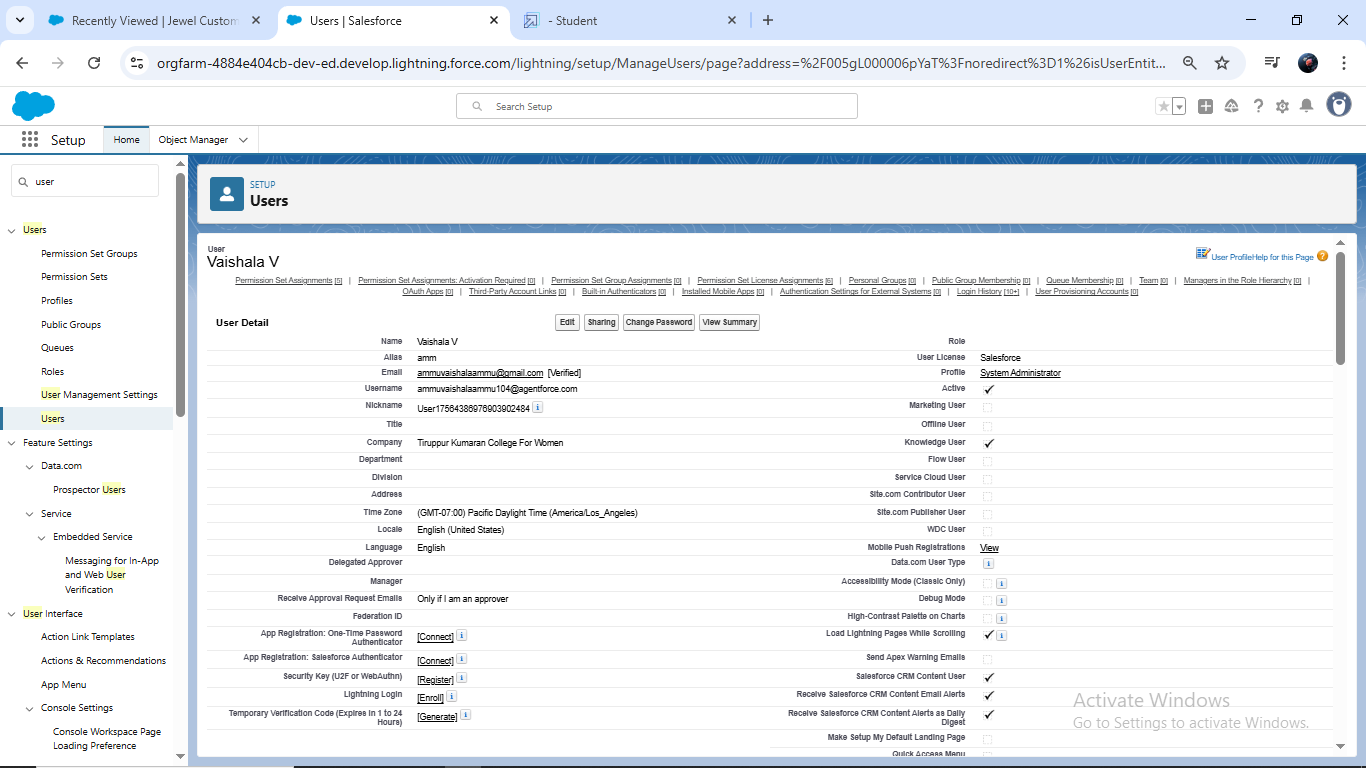
### 7.1 Creating Gold Smith Role





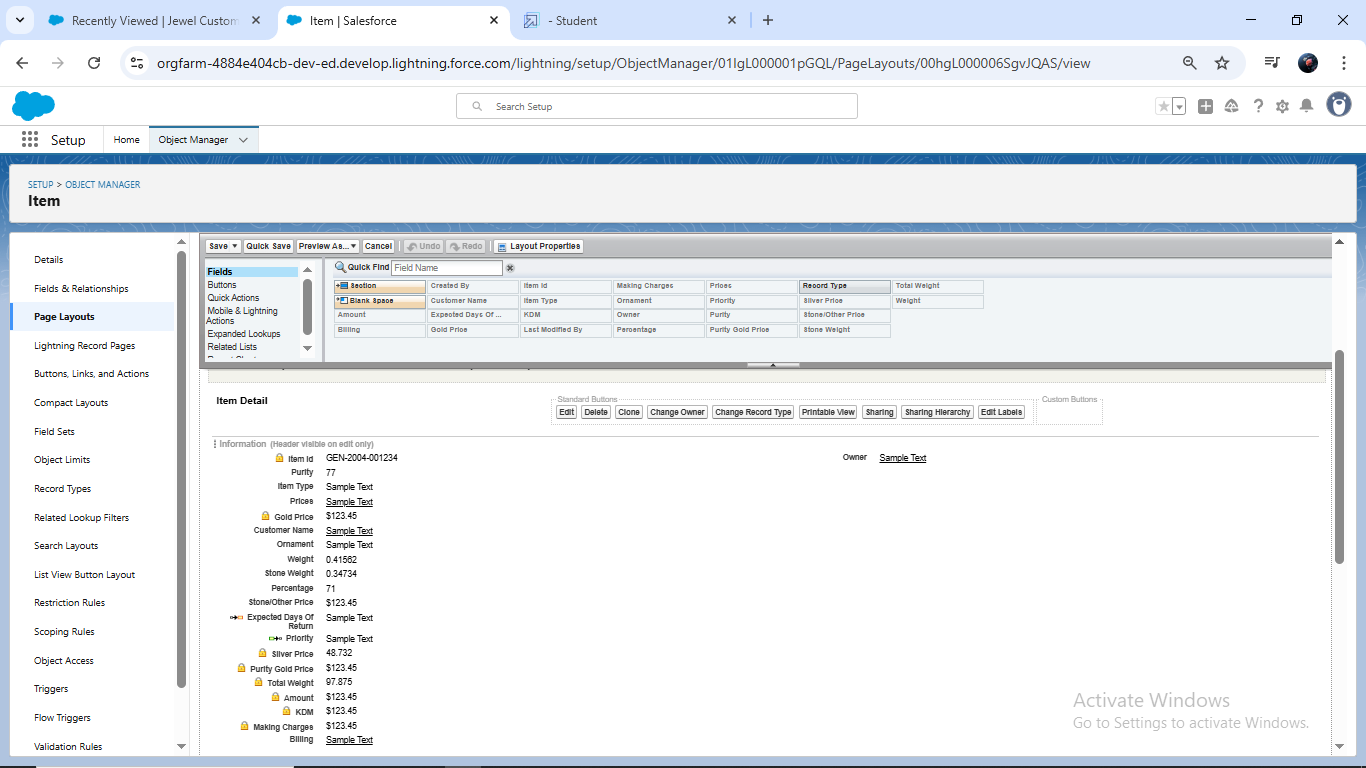
### Users

A user is anyone who logs in to Salesforce.



### Page layouts

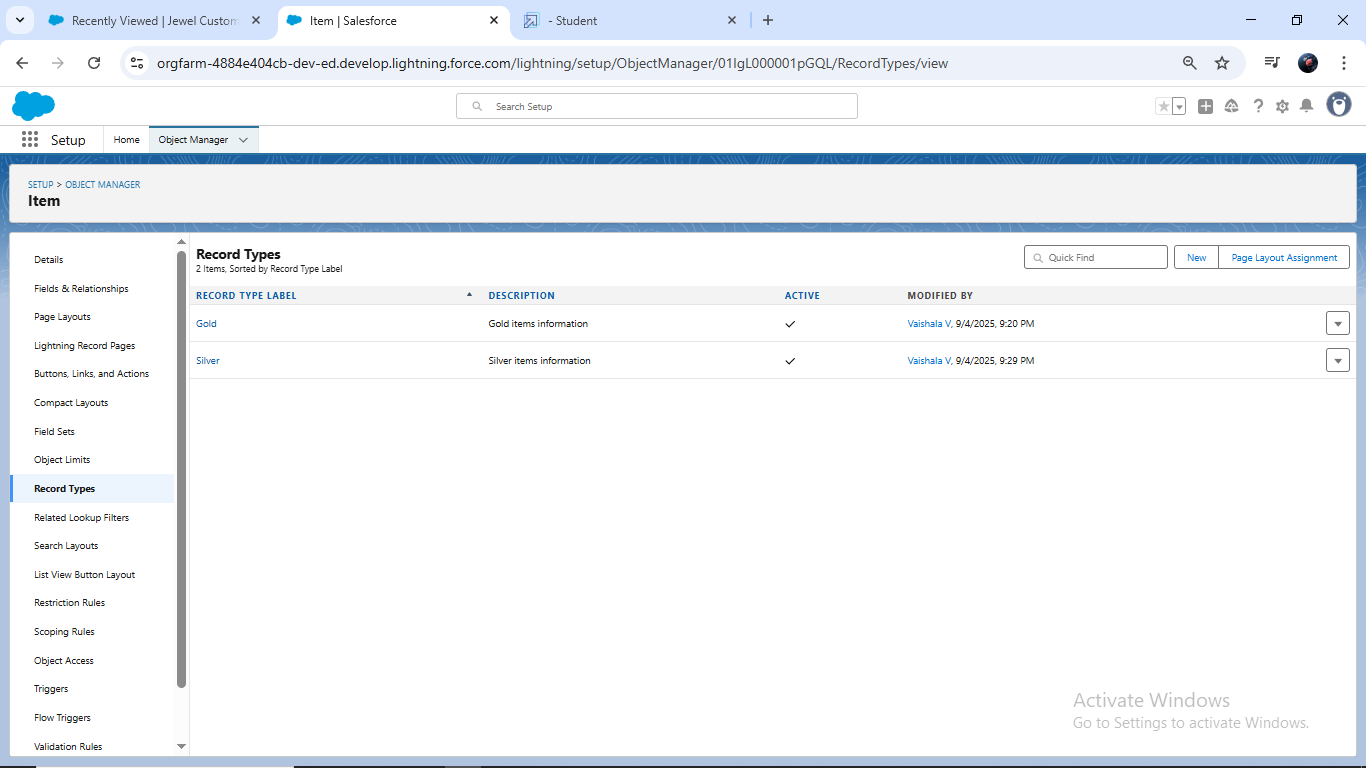
* Page Layout in Salesforce allows us to customise the design and organise detail and edit pages of records in Salesforce

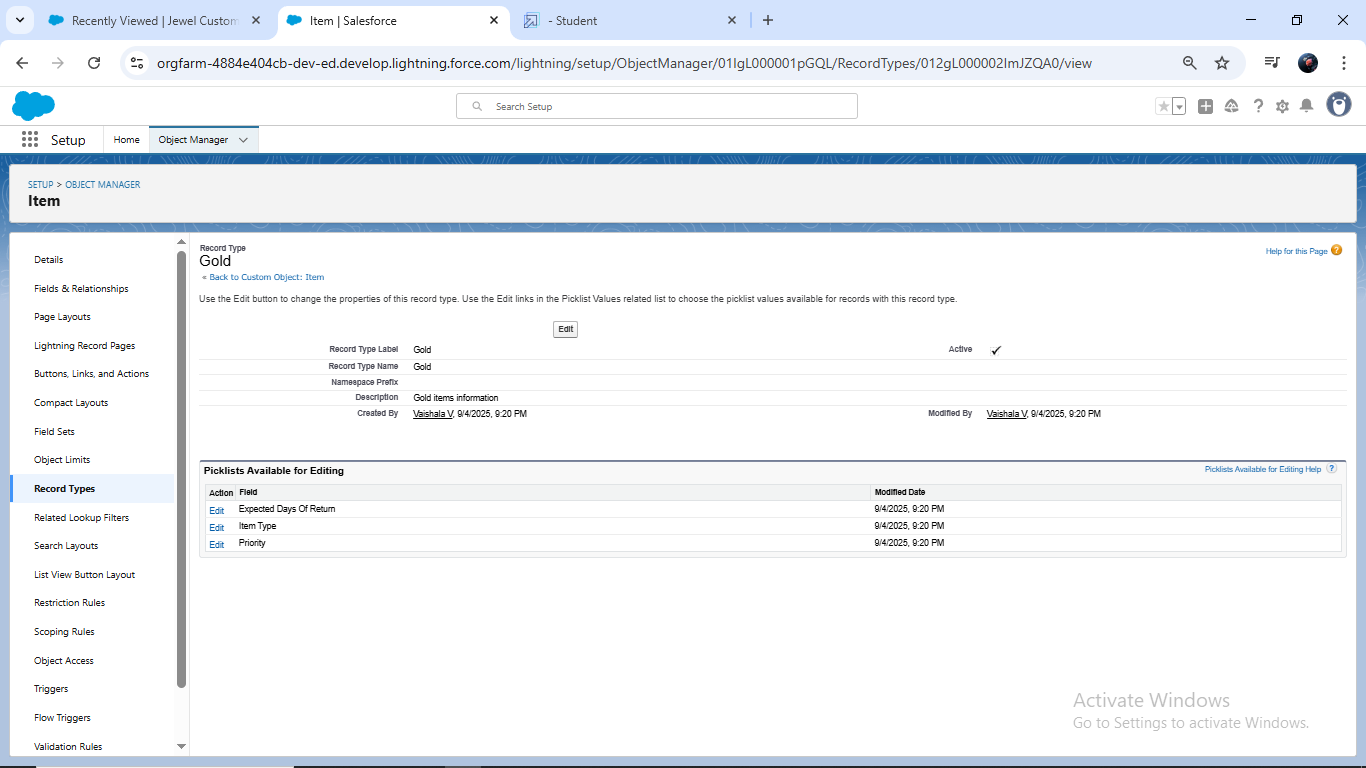


### 

### 10.Record Types

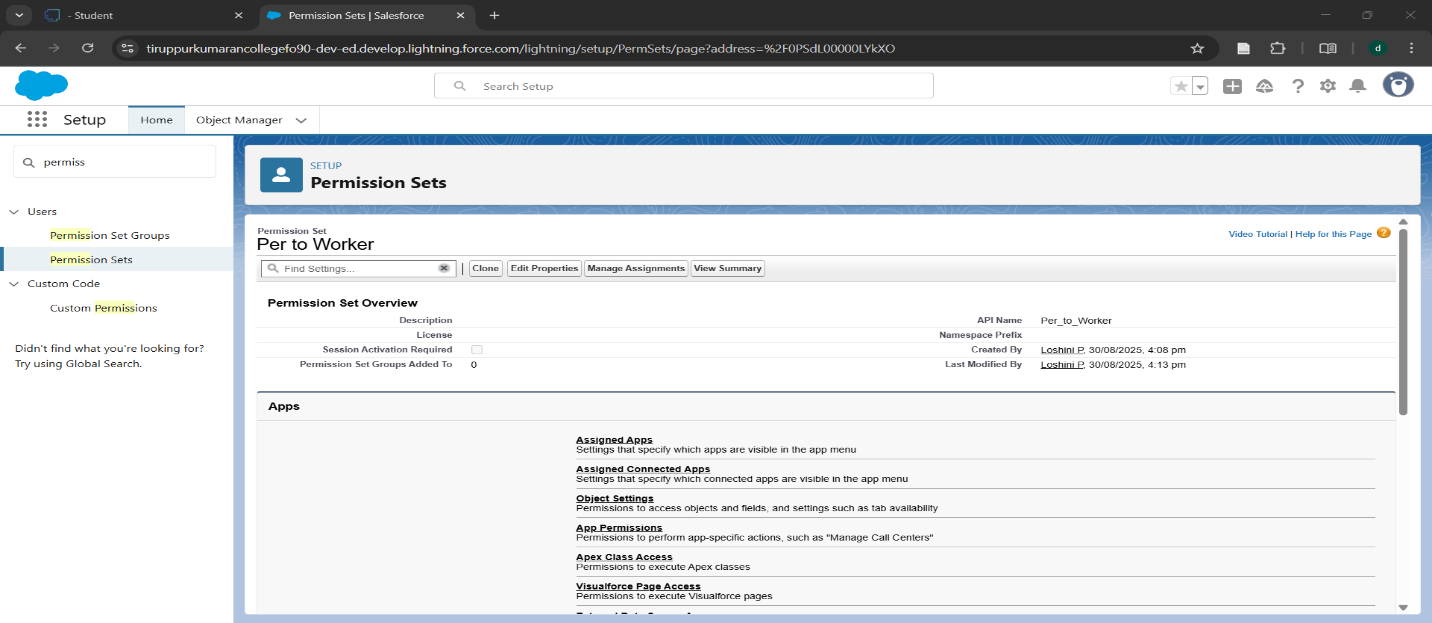
* Record Types are a way of grouping many records of one type for that object.





### Permission sets

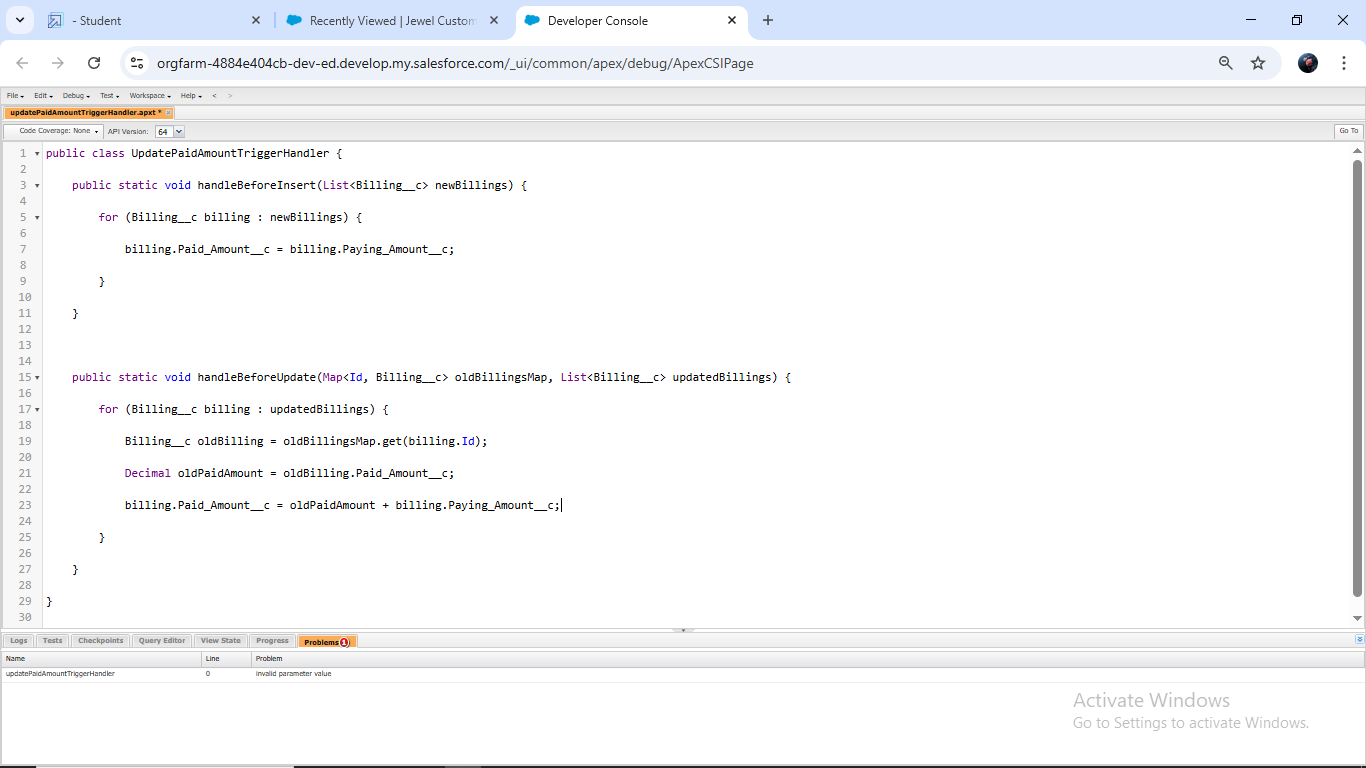
A standard permission set consists of a group of common permissions for a particular feature associated with a permission set licence



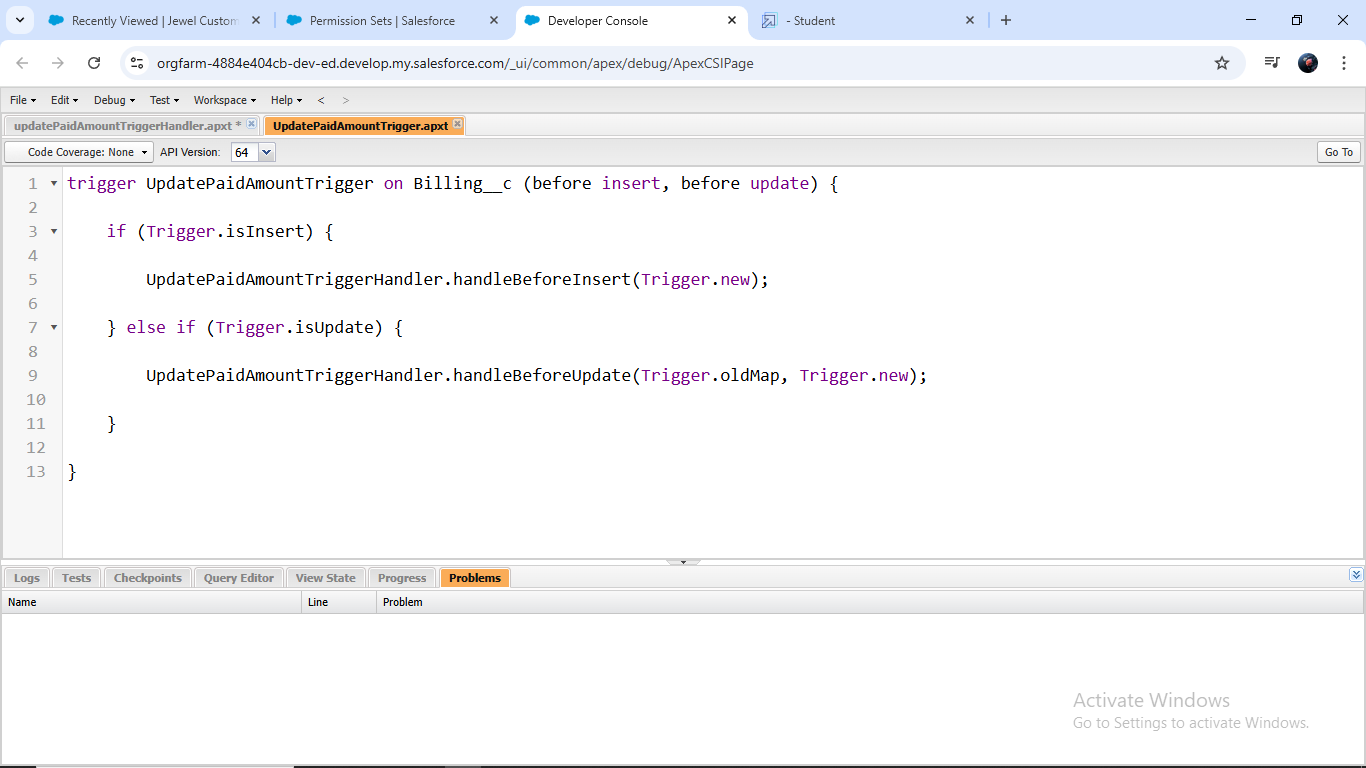
1. **Trigger**

* . Triggers are used to customise and automate actions in response to these events.

### 12.1 Create a Trigger Handler class

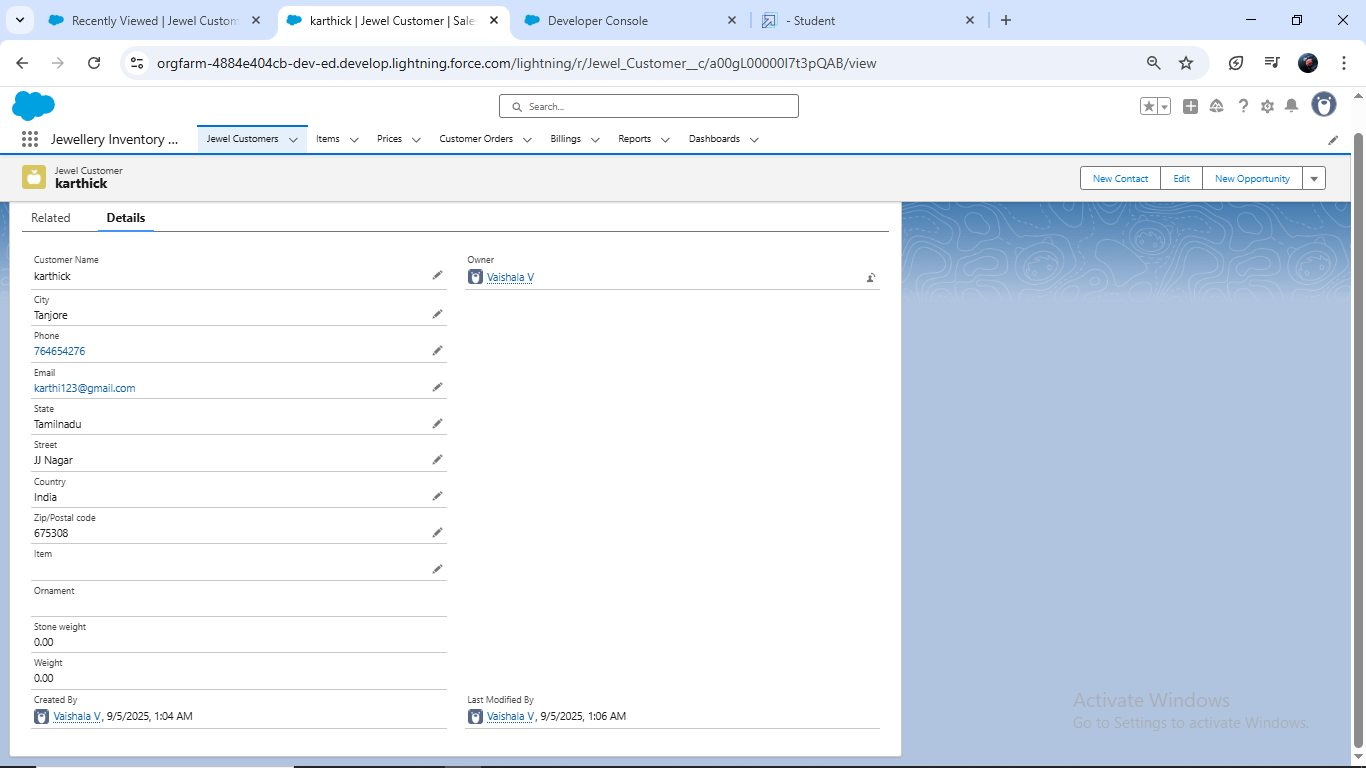


### Create the trigger



### User Adoption

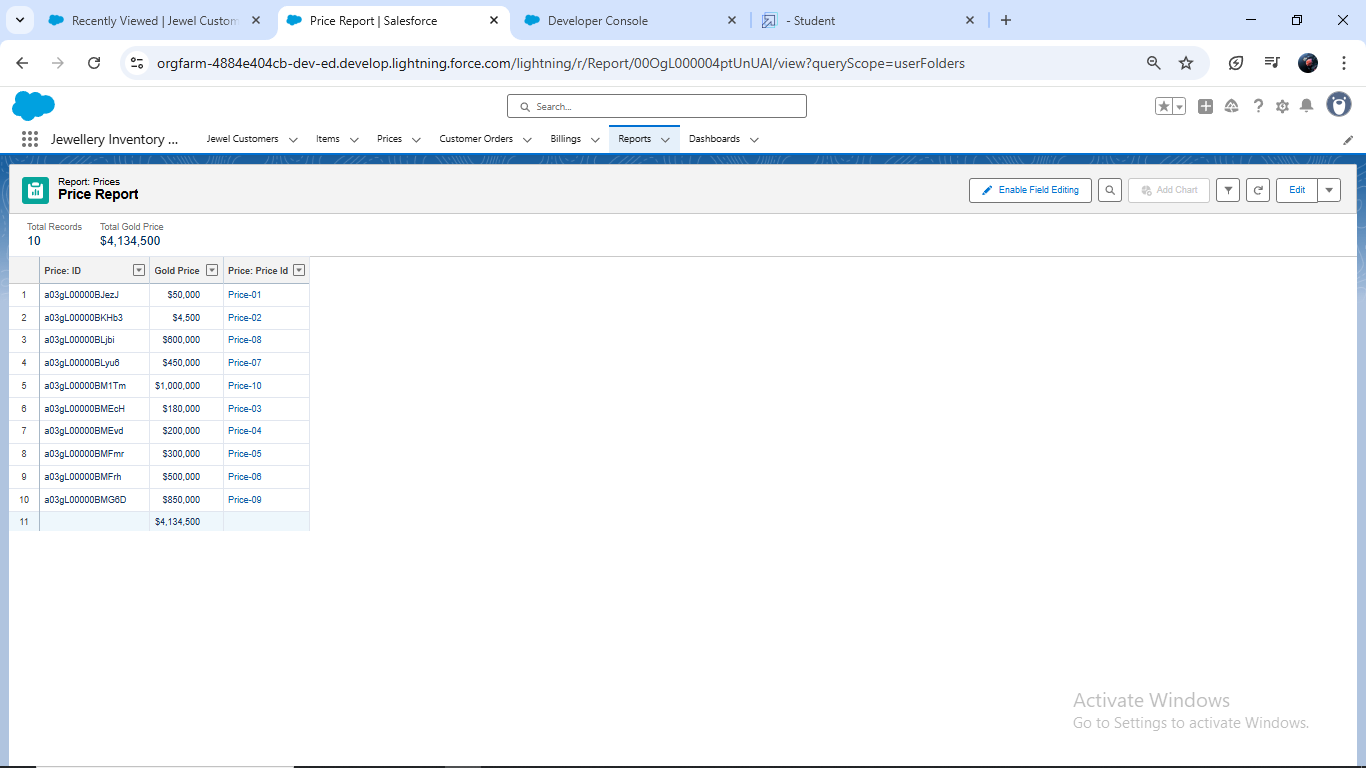
* . In this unit, you will learn about users and how you add users to your Salesforce org.



### Reports

* Reports give you access to your Salesforce data.

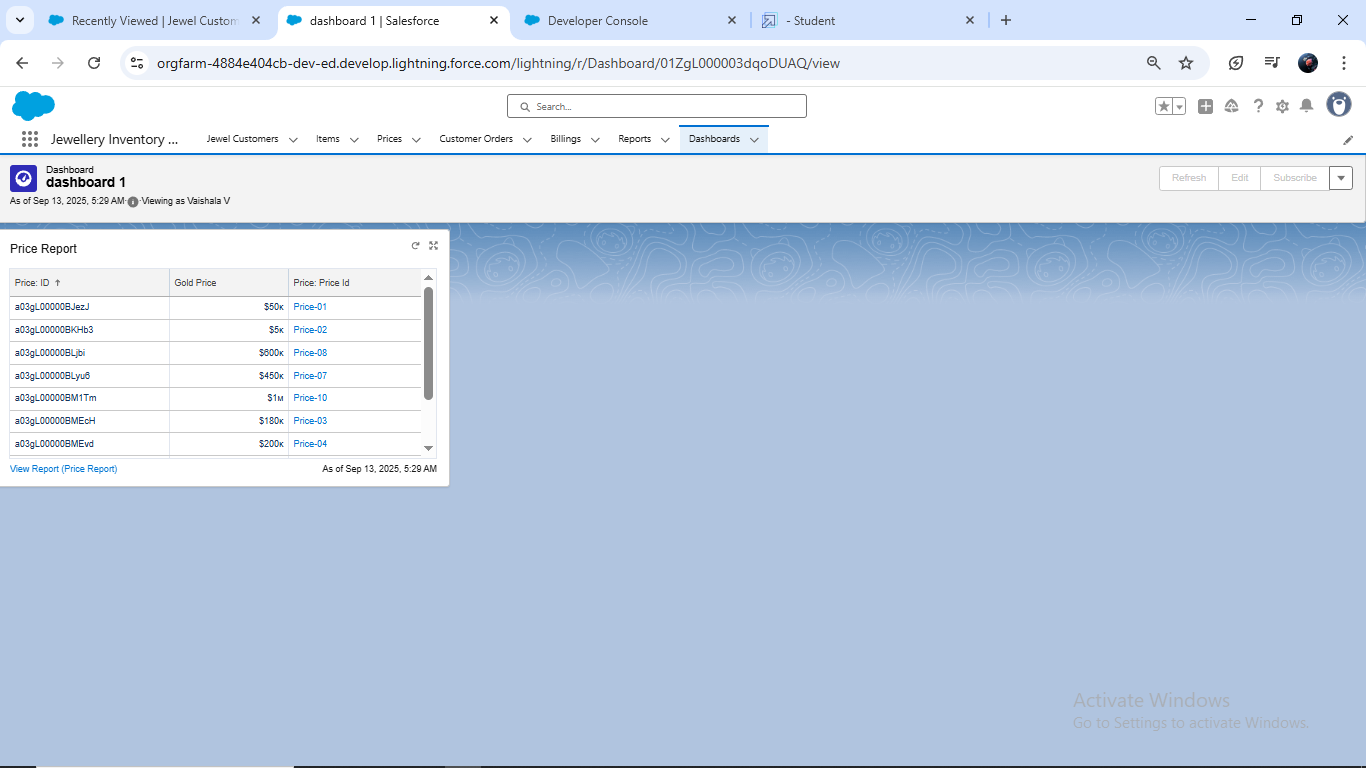
### 14.1 Create Report



### Dashboards

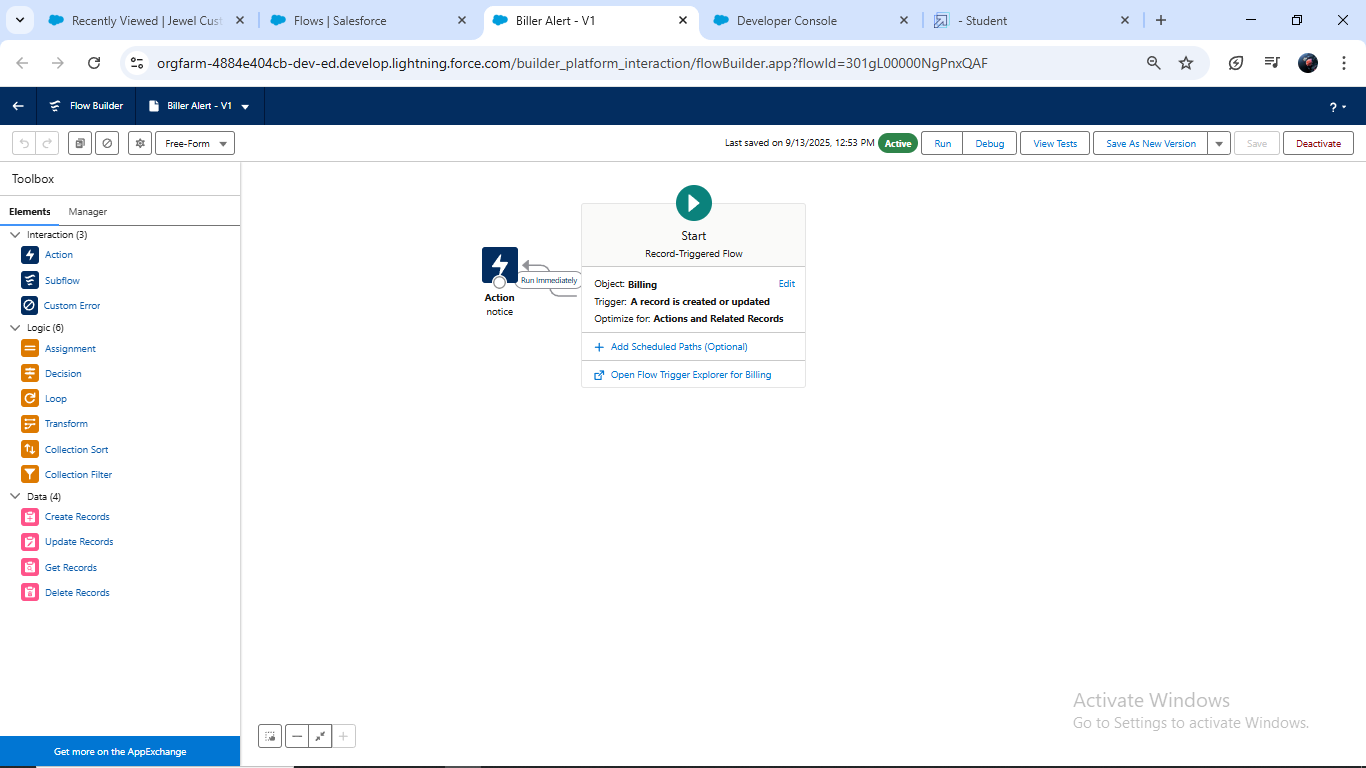
Dashboards help as visually understand changing business conditions so you can make decisions based on the real-time data you’ve gathered with reports.

### 15.1 Create Dashboard

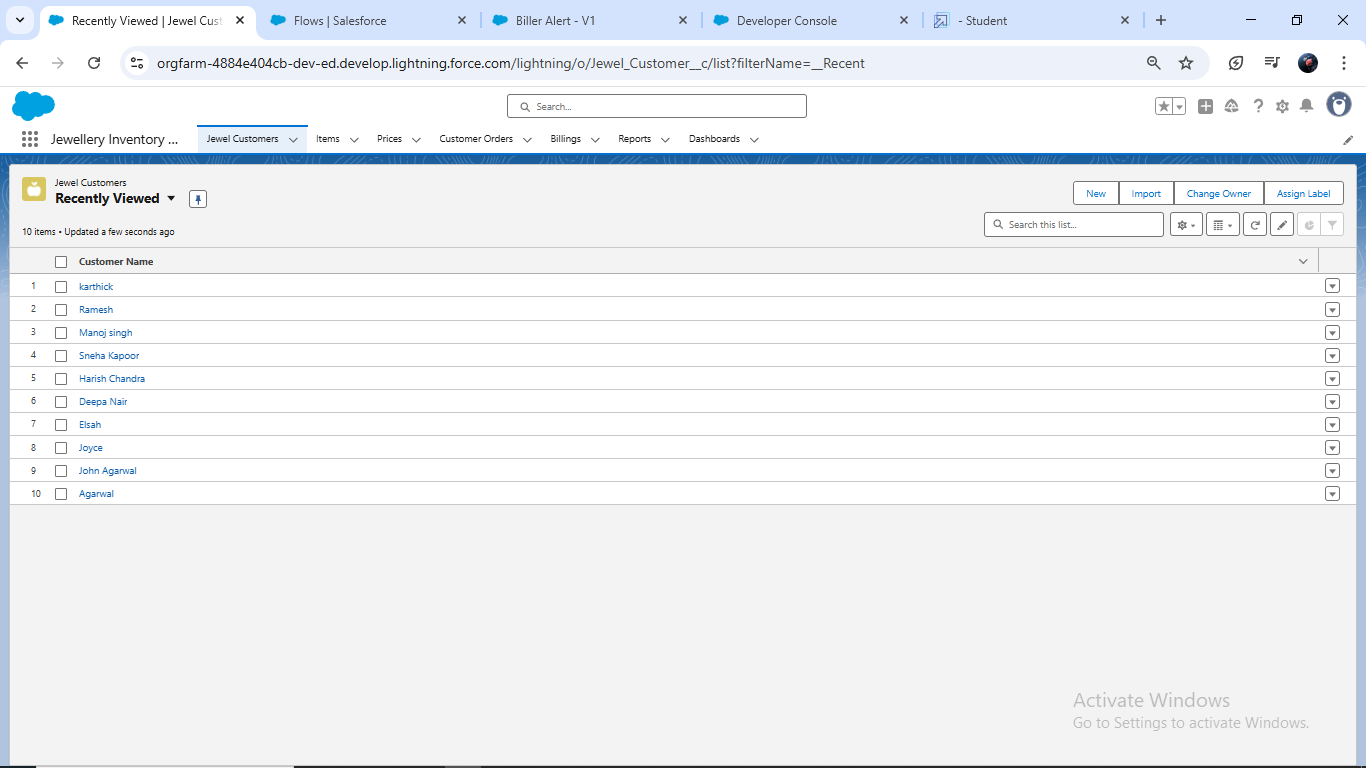


### Flows

* . Flows are built using a visual interface and can be created without any coding knowledge



1. Result



## Conclusion

The CRM Application for Jewel Management will act as a complete solution for jewelry businesses, helping them manage their customers, inventory, and sales efficiently. By integrating modern technologies and CRM practices, this system will enhance business growth, customer satisfaction, and operational excellence.

**APPENDIX**

* **Source Code:** Provided in Apex Classes

**Trigger handler:**

**UpdatePaidAmountTriggerHandler**

**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**

**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);**

**}**

**}**