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

**College Name : Kaamadhenu Arts And Science**

**College code : bru4p**

**Group project**

**TEAM ID : NM2025TMID21157**

**TEAM MEMBERS :**

**Team Leader : Sastika M**

**Email:sastika.cs23@Kascsathy.ac.in**

**Team Member :Kavyashri B**

**Email: Kavyashri.cs23@Kascsathy.ac.in**

**Team Member : Anandhan A**

**Email: Anandhan.cs23@Kascsathy.ac.in**

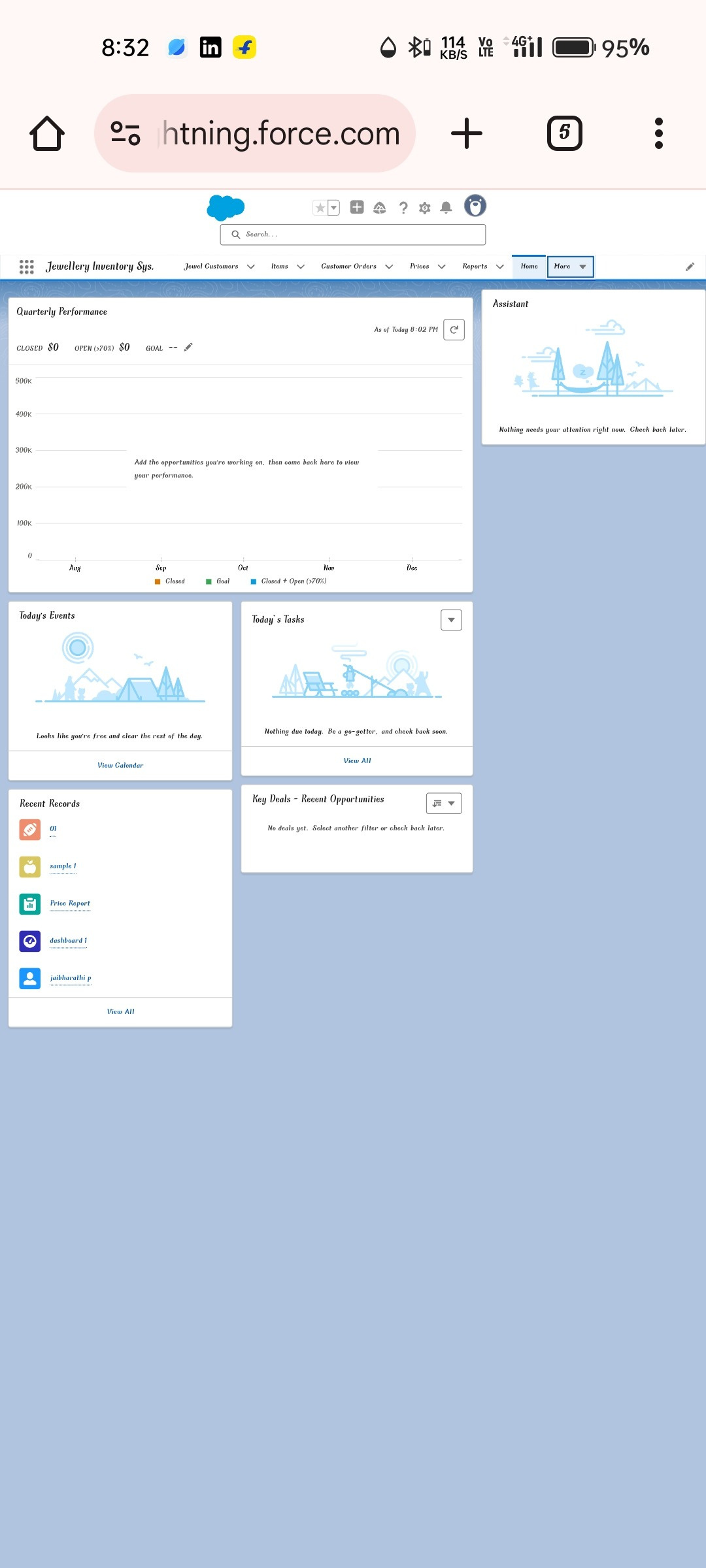
**Team Member : Jaibharathi P**

**Email:jaibharathip.cs23@Kascsathy.ac.in**

**1.INTRODUCTION**

**1.1 Project Overview**

The Jewel Inventory System is a comprehensive software Solution designed to streamline and manage the inventory and sales processes of a jewellery store or jewellerymanufacturer.



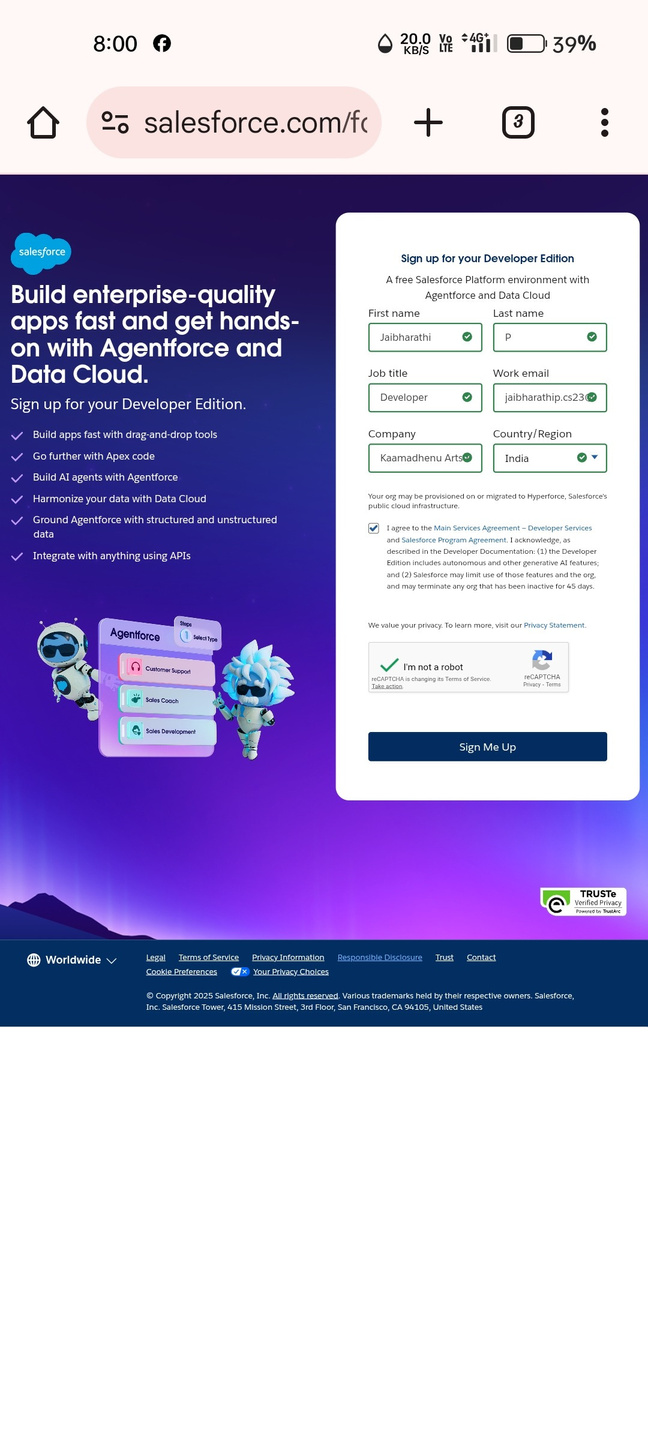
**1.2 Purpose**

The system aims to provide an efficient and user-friendly solution to track and control the inventory of various jewellery items, maintain accurate records, and facilitate seamless sales transactions.

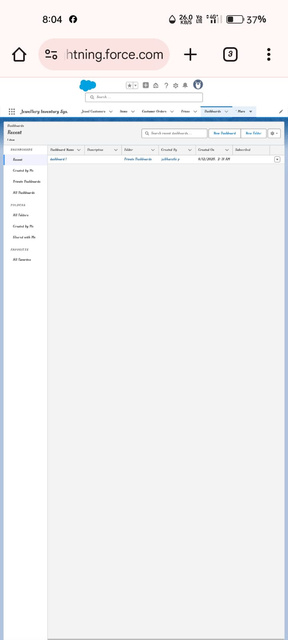
**DEVELOPMENT PHASE**

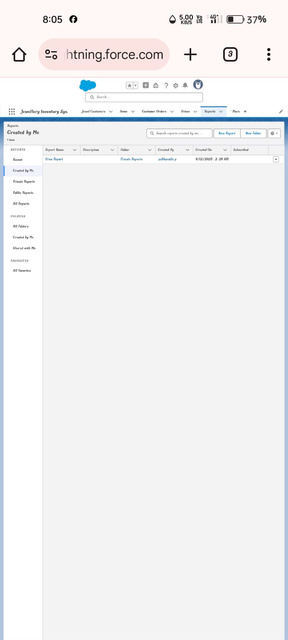
**Creating Developer Account:**

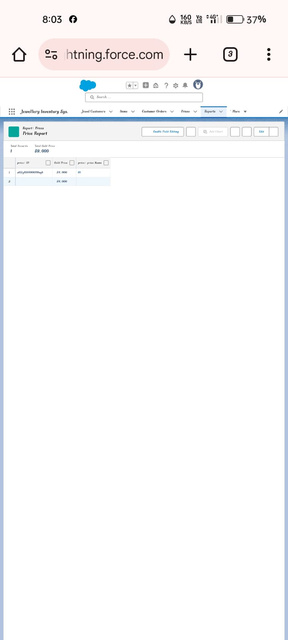
By using this URL **-** [**https://developer.salesforce.com/signup**](https://developer.salesforce.com/signup)

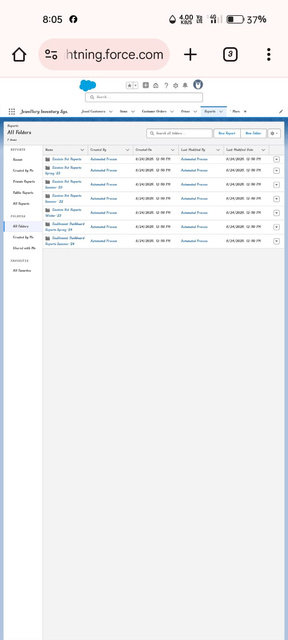


● Created objects: Jewel Customer, Item, Customer Order, Price, Billing



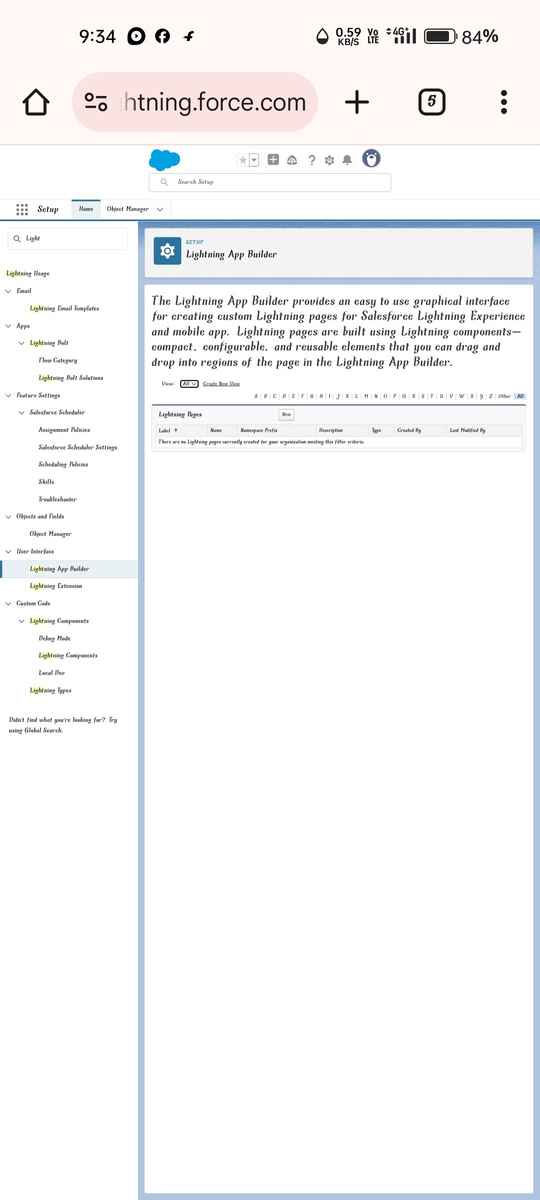






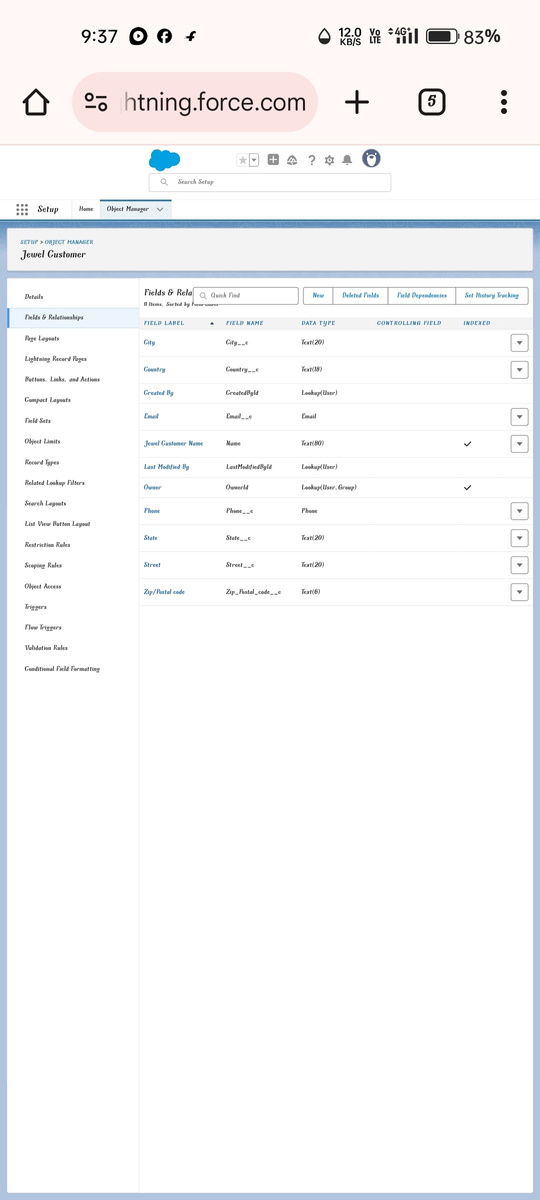
●Creating Custom tabs : Customer & Item

● Developed Lightning App with relevant tabs



Fields :

● **Creating Lookup Relationship**



● **Creating a Master-Detail Relationship**

● **Creating Text Field in Jewel Customer Object**

● **Creating the Phone field in object Jewel Customer**

● **Creating the Email field in object Jewel Customer**

**Creating the number field in Item object**

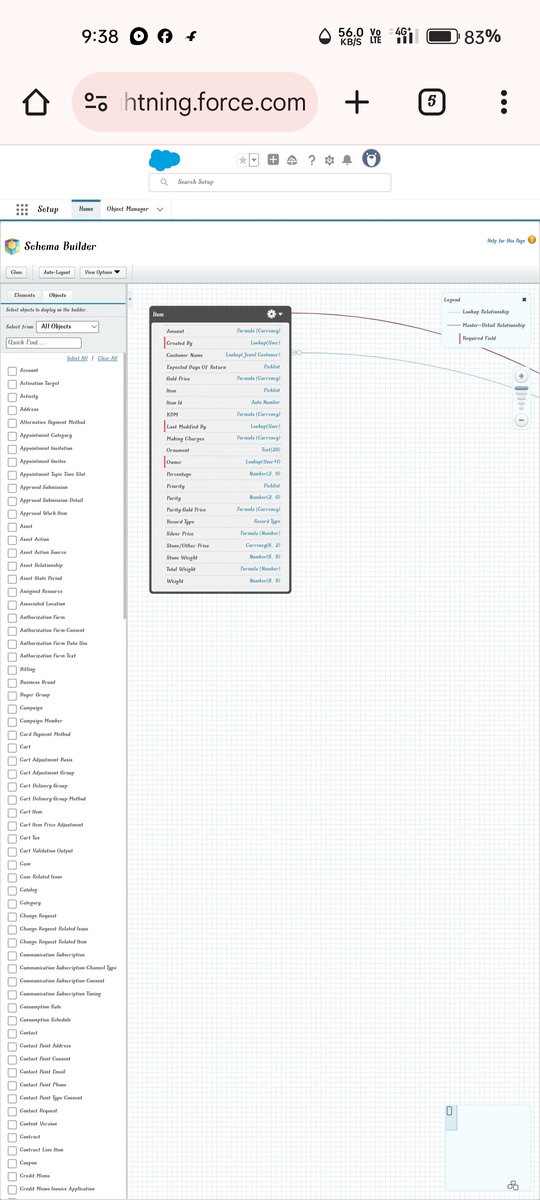
Creating Picklist Field in Item Object

**Creating Currency Field in Price Object**

**Creating Formula Field(Cross Object) in Item Object**

**Creating Remaining Fields in Objects**

**Schema Builder**



**Creating the Field Dependencies**

**Creating the validation rule**

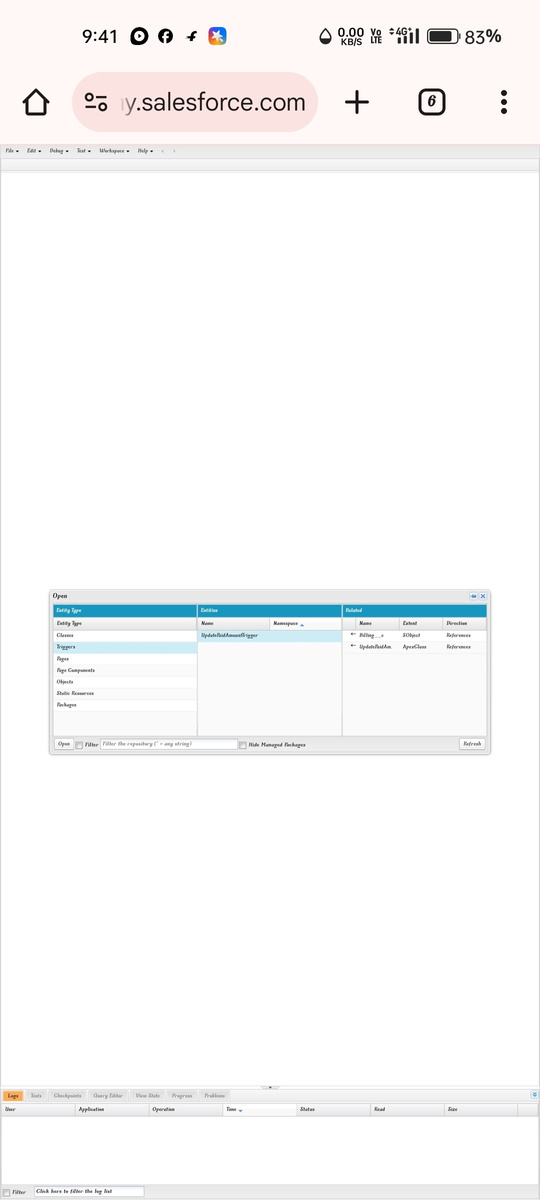
● Approval Process creation

For Tenant Leaving:

For Check for Vacant:

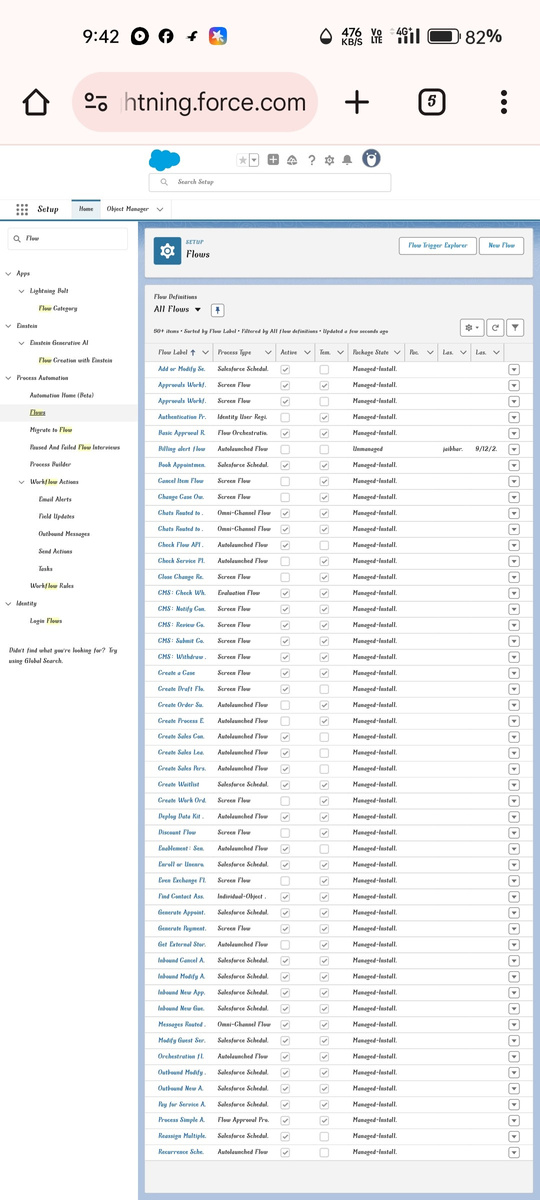
● Apex Trigger

Create an Apex Trigger



Create an Apex Handler class

● FLOWS



● Schedule class:

Create an Apex Class

Schedule Apex class

**FUNCTIONAL AND PERFORMANCE TESTING**

**Performance Testing**

● Trigger validation by entering duplicate tenant-property records

● Validation Rule checking

● Test flows on payment update

● Approval process validated through email alerts and status updates

**RESULTS**

**Output Screenshots**

● Tabs for Property, Tenant, Lease, Payment

● Email alerts

● Request for approve the leave

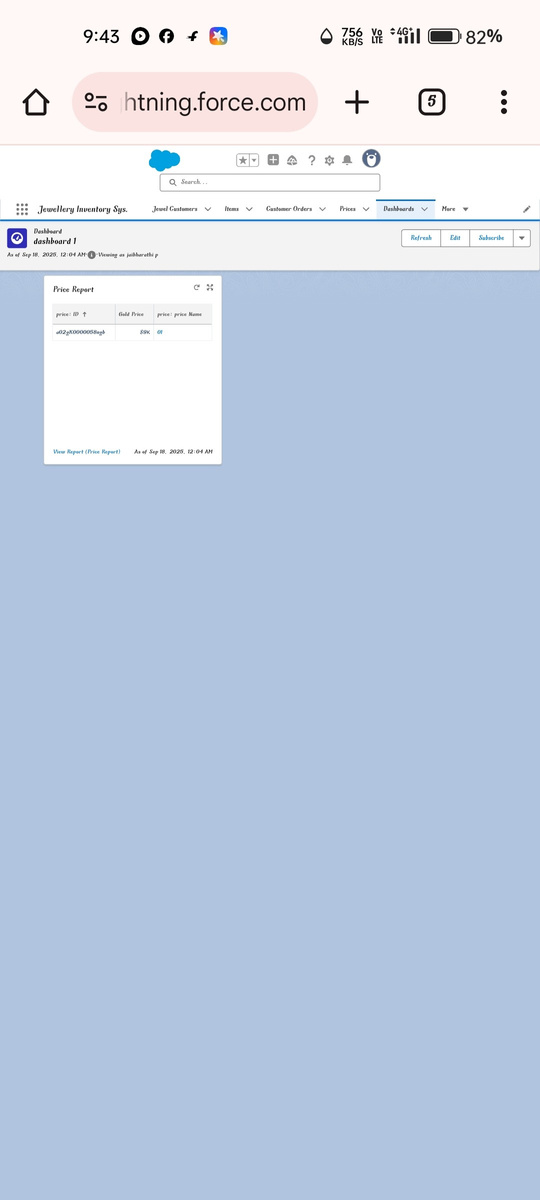
● Leave approved

● Leave rejected

● Flow runs

● Trigger error messages

● Approval process notifications



**ADVANTAGES & DISADVANTAGES** .

**CONCLUSION**

The Lease Management System successfully streamlines the operations of leasing through a structured, automated Salesforce application. It improves efficiency, communication, and data accuracy for both admins and tenants.

**APPENDIX**

● **Source Code:** Provided in Apex Classes and Triggers

**Test.apxt:**

trigger test on Tenant\_\_c (before insert) { if

(trigger.isInsert && trigger.isBefore){

testHandler.preventInsert(trigger.new);

} }

**testHandler.apxc:**

public class

testHandler {

public static void

preventInsert(List<

Tenant\_\_c> newlist)

{ Set<Id>

existingPropertyIds

= new Set<Id>()

for (Tenant\_\_c existingTenant : [SELECT Id, Property\_\_c FROM Tenant\_\_c WHERE Property\_\_c != null]) {

existingPropertyIds.add(existingTenant.Property\_\_c;

} for (Tenant\_\_c newTenant :

newlist) {

if (newTenant.Property\_\_c != null &&

existingPropertyIds.contains(newTenant.Property\_\_c)) { newTenant.addError('A tenant can have only one property');

}

}

}

}

**MothlyEmailScheduler.apxc:**

global class MonthlyEmailScheduler implements Schedulable {

global void execute(SchedulableContext sc) { Integer

currentDay = Date.today().day(); if (currentDay == 1) {

sendMonthlyEmails();

}

} public static void

sendMonthlyEmails() { List<Tenant\_\_c>

tenants = [SELECT Id, Email\_\_c FROM

Tenant\_\_c]; for (Tenant\_\_c tenant :

tenants) {

String recipientEmail = tenant.Email\_\_c;

String emailContent = 'I trust this email finds you well. I am writing to remind you that the monthly rent is due Your timely payment ensures the smooth functioning of our rental arrangement and helps maintain a positive living environment for all.';

String emailSubject = 'Reminder: Monthly Rent Payment Due';

Messaging.SingleEmailMessage email = new

Messaging.SingleEmailMessage(); email.setToAddresses(new

String[]{recipientEmail}); email.setSubject(emailSubject);

email.setPlainTextBody(emailContent);

Messaging.sendEmail(new Messaging.SingleEmailMessage[]{email}); }

}

}