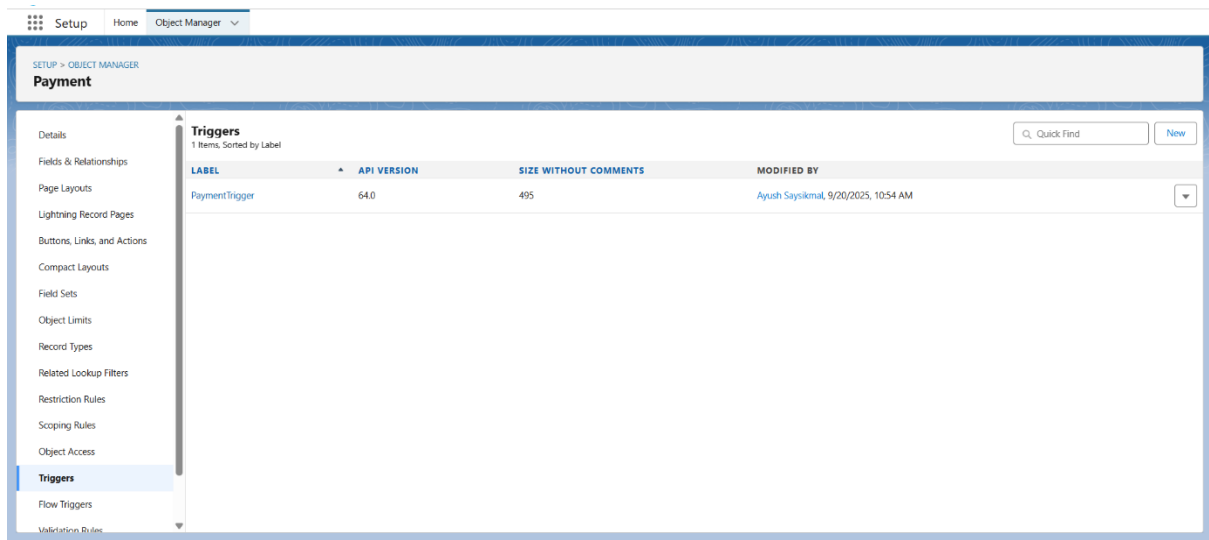
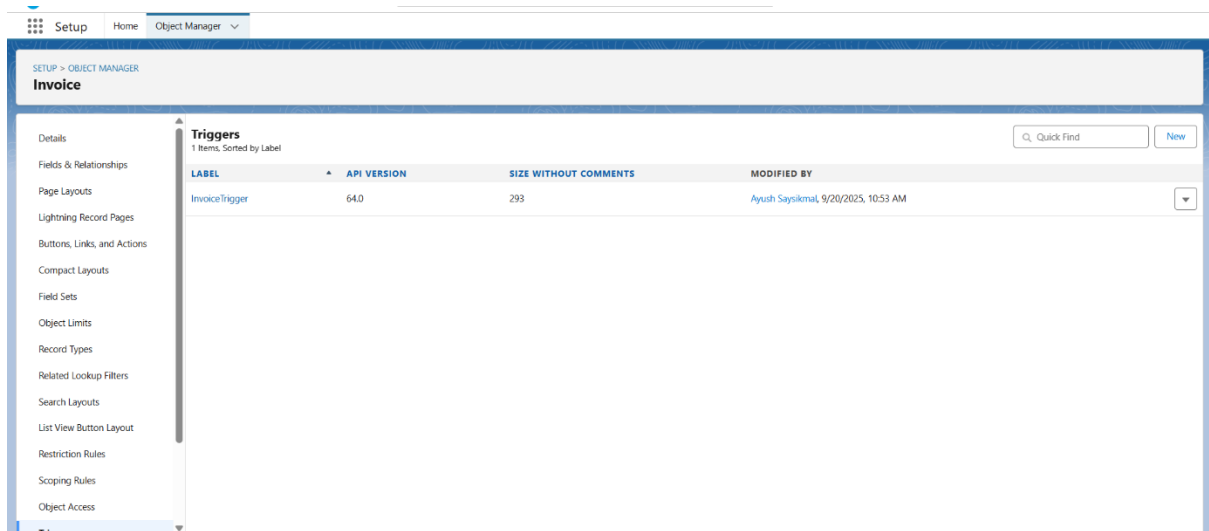


# Phase 5: Apex Programming

## 1. Apex Trigger

- **Setup->Object Manager->Invoice/Payment->Trigger**
  - **Invoice Trigger**
  - **Payment Trigger**



## 2.Apex Class

- Setup->Apex Class->New Apex Class
  - Invoice Handler
  - Payment Handler

The screenshot shows the Salesforce Setup interface. On the left, the 'Setup' menu is open, and 'Apex Classes' is selected under 'Custom Code'. The main content area displays the 'Apex Class Detail' for 'InvoiceHandler'. The class is owned by 'Ayush Sayakmal' and was created on 9/20/2025 at 10:52 AM. It is currently 'Active' with a code coverage of 76% (13/17). The 'Class Body' tab is selected, showing the following Apex code:

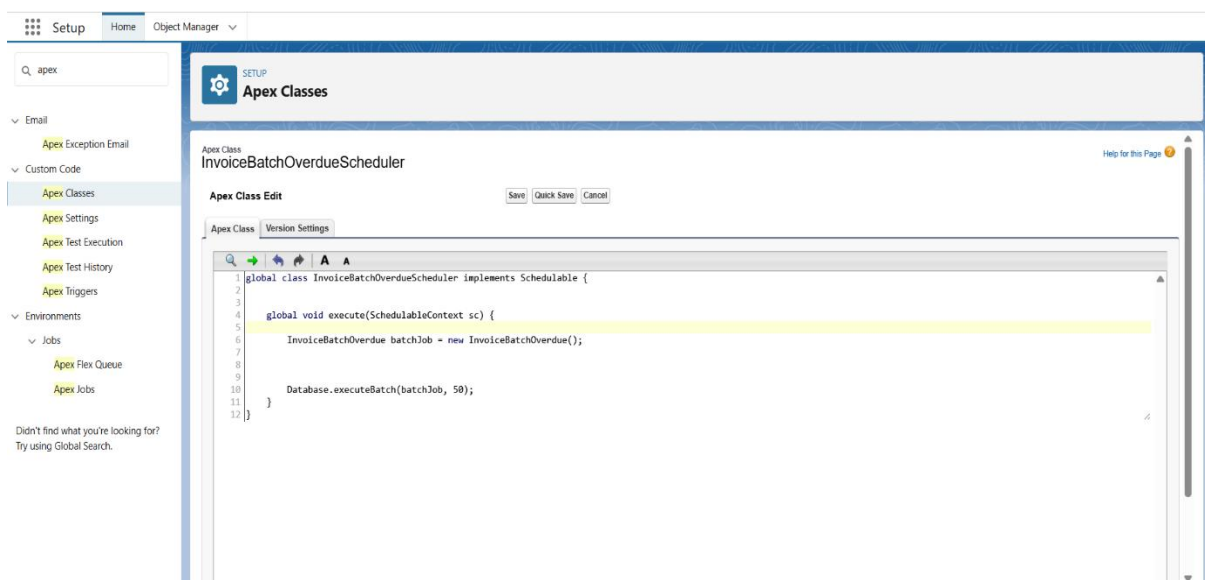
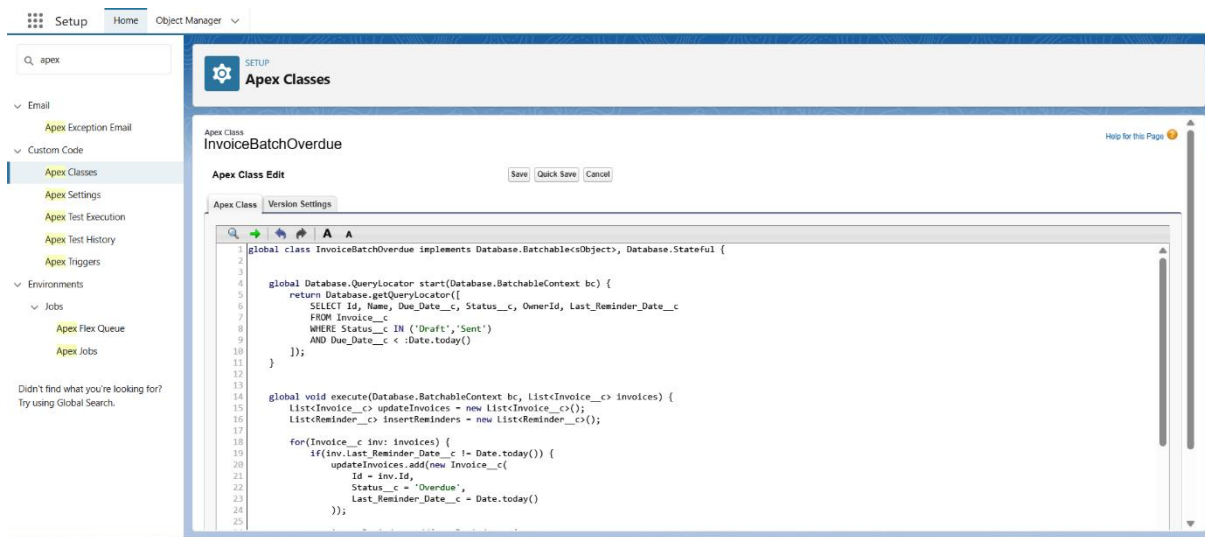
```
1 public with sharing class InvoiceHandler {
2
3     public static void beforeInsert(List<Invoice__c> newInvoices) {
4         if(newInvoices.isEmpty()) return;
5
6         Set<Id> custIds = new Set<Id>();
7         for(Invoice__c inv: newInvoices) {
8             if(inv.Customer__c != null && inv.Invoice_Date__c != null && inv.Amount__c != null) {
9                 custIds.add(inv.Customer__c);
10            }
11        }
12        if(custIds.isEmpty()) return;
13
14        Map<String, Id> existingMap = new Map<String, Id>();
15        for(Invoice__c ex: [SELECT Id, Customer__c, Invoice_Date__c, Amount__c
16                          FROM Invoice__c
17                          WHERE Customer__c IN :custIds]) {
18            String key = ex.Customer__c + 'T' + ex.Invoice_Date__c + 'T' + ex.Amount__c;
19            existingMap.put(key, ex.Id);
20        }
21
22        for(Invoice__c inv: newInvoices) {
23            String key = inv.Customer__c + 'T' + inv.Invoice_Date__c + 'T' + inv.Amount__c;
24            if(existingMap.containsKey(key)) {
25                inv.addError('Duplicate Invoice for this customer with same date and amount');
26            }
27        }
28    }
29 }
```

The screenshot shows the Salesforce Setup interface with 'Apex Classes' selected. The main content area displays the 'Apex Class Edit' view for 'PaymentHandler'. The class is owned by 'Ayush Sayakmal' and was created on 9/20/2025 at 10:52 AM. It is currently 'Active' with a code coverage of 76% (13/17). The 'Class Body' tab is selected, showing the following Apex code:

```
1 public with sharing class PaymentHandler {
2
3     public static void updateInvoicePaymentStatus(Set<Id> invoiceIds) {
4         if(invoiceIds == null || invoiceIds.isEmpty()) return;
5
6         Map<Id, Decimal> paidMap = new Map<Id, Decimal>();
7         for (AggregateResult ar : [
8             SELECT Invoice__c invId, SUM(Amount_Paid__c) sumPaid
9             FROM Payment__c
10            WHERE Invoice__c IN :invoiceIds AND Payment_Status__c = 'Completed'
11            GROUP BY Invoice__c
12        ]) {
13            paidMap.put((Id)ar.get('invId'), (Decimal)ar.get('sumPaid'));
14        }
15
16        List<Invoice__c> invoicesToUpdate = new List<Invoice__c>();
17        for (Invoice__c inv : [SELECT Id, Amount__c, Status__c FROM Invoice__c WHERE Id IN :invoiceIds]) {
18            Decimal paid = paidMap.containsKey(inv.Id) ? paidMap.get(inv.Id) : 0;
19            String newStatus;
20
21            if (inv.Amount__c != null && paid >= inv.Amount__c) newStatus = 'Paid';
22            else if (paid > 0) newStatus = 'Partially Paid';
23            else newStatus = inv.Status__c;
24        }
25    }
26 }
```

### 3. Batch Apex

- **Setup->Apex Classes->New Class**
  - **Invoice Batch Overdue**
  - **Invoice Batch Overdue Scheduler**



## 4. Scheduled Apex

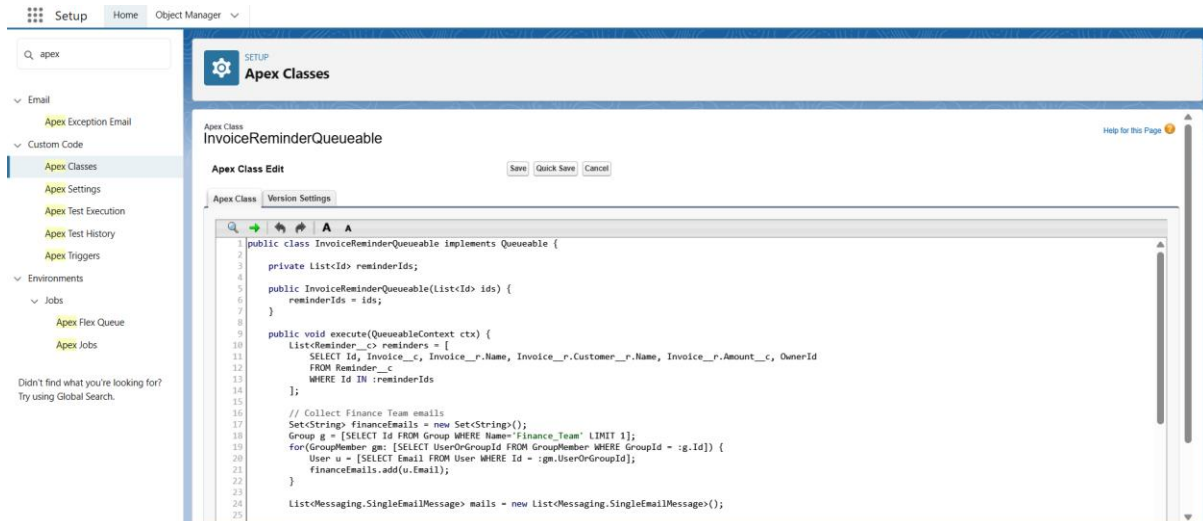
- **Setup->Scheduled Jobs->New Job**
  - **Invoice Age Monthly**
  - **Invoice Overdue Nightly**

The screenshot shows the Salesforce 'Schedule Apex' configuration page. The left sidebar contains navigation links for Home, Administer, and various system settings. The main content area is titled 'Schedule Apex' and includes a description: 'Schedule an Apex class that implements the Schedulable interface to be automatically executed on a specified interval.' Below this, there are buttons for 'Reschedule Job', 'Pause Job', 'Cancel', and 'Delete'. The 'Job Name' is 'InvoiceAgingMonthly' and the 'Apex Class' is 'InvoiceAgingScheduler'. Under 'Schedule Using', 'Schedule Builder' is selected. The 'Schedule Apex Execution' section shows 'Frequency' set to 'Monthly' (selected over 'Weekly'), 'Start' date as '9/20/2025', 'End' date as '9/21/2025', and 'Preferred Start Time' as '2:00 AM'. There are also options for 'On day 1 of every month' and 'On the 1st Sunday of every month'. A note at the bottom states: 'Exact start time will depend on job queue activity.'

The screenshot shows the Salesforce 'Schedule Apex' configuration page for a different job. The left sidebar is identical to the previous screenshot. The main content area is titled 'Schedule Apex' with the same description. The 'Job Name' is 'InvoiceOverdueNightly' and the 'Apex Class' is 'InvoiceBatchOverdueScheduler'. Under 'Schedule Using', 'Cron Expression' is selected. The 'Cron Expression' field contains '0 0 2 \* \* ?'. A note at the bottom states: 'Exact start time will depend on job queue activity.'

## 5. Queueable Apex

- Setup->Apex classes->New Classes
  - Invoice Reminder Queueable



## 6. Test Classes

- Setup->Apex Classes->New Classes
  - Invoice Test Class

