Phase 5: Apex Programming (Developer)

Objective: Implement backend business logic, automation, and asynchronous processes using Apex to enhance system functionality for Service Request and Invoice management.

---

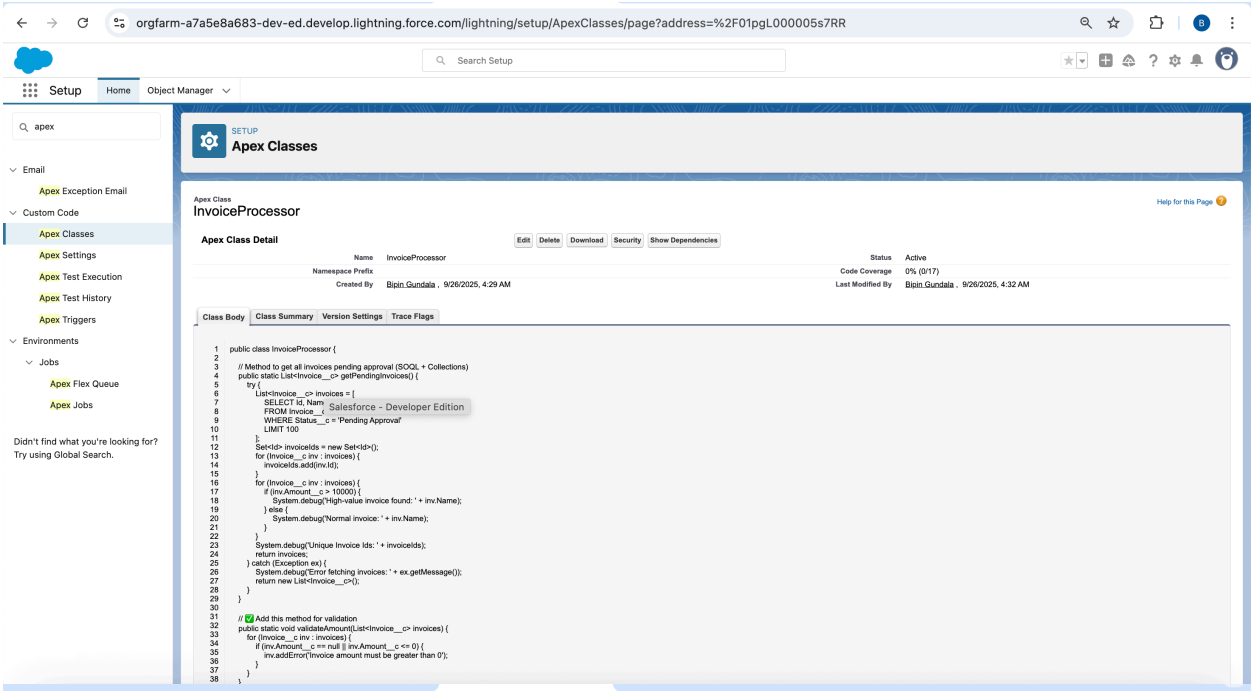
1. Classes & Objects

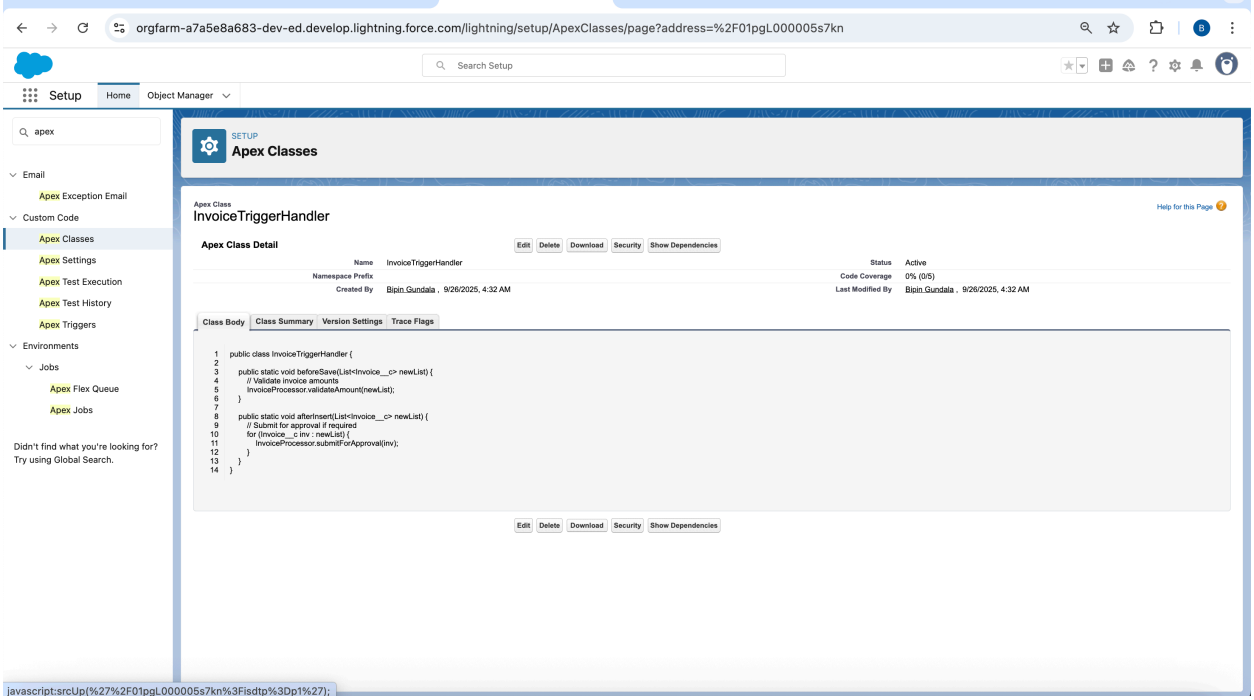
Apex Classes Implemented:

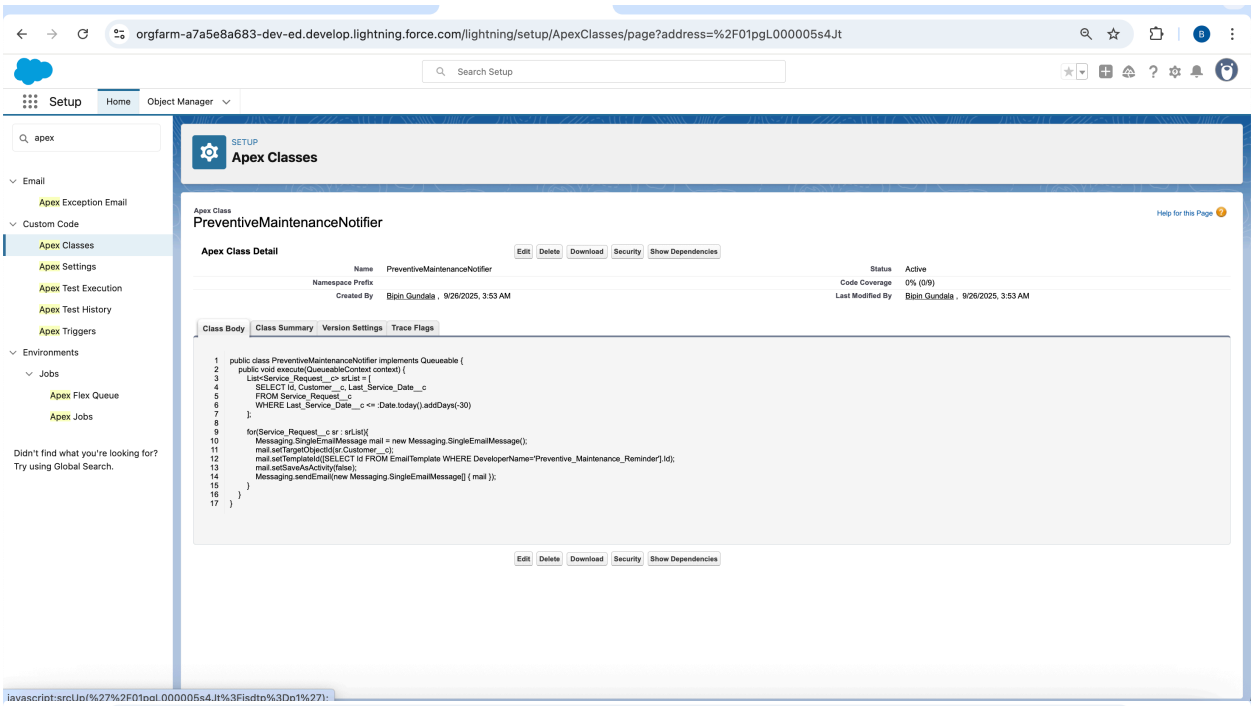
Class Name	Purpose
`ServiceRequestHandler`	Handles business logic for Service Requests, including technician assignment and sending confirmation emails.
`InvoiceProcessor`	Handles invoice processing, including fetching pending invoices, validating amounts, and identifying high-value invoices.
`ServiceRequestTriggerHandler`	Acts as the trigger handler for `Service_Request__c` object. Calls the `ServiceRequestHandler` methods for bulk processing.
`InvoiceTriggerHandler`	Acts as the trigger handler for `Invoice__c` object. Calls `InvoiceProcessor` methods for validation and approval submission.

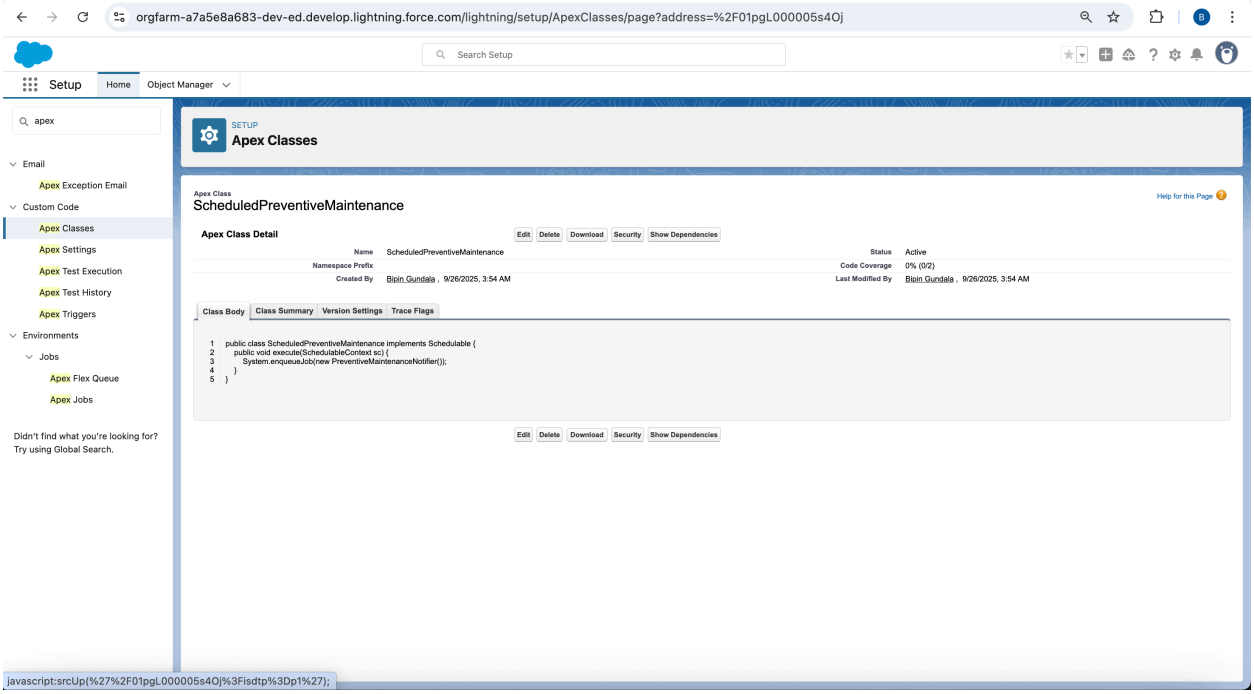
Key Features:

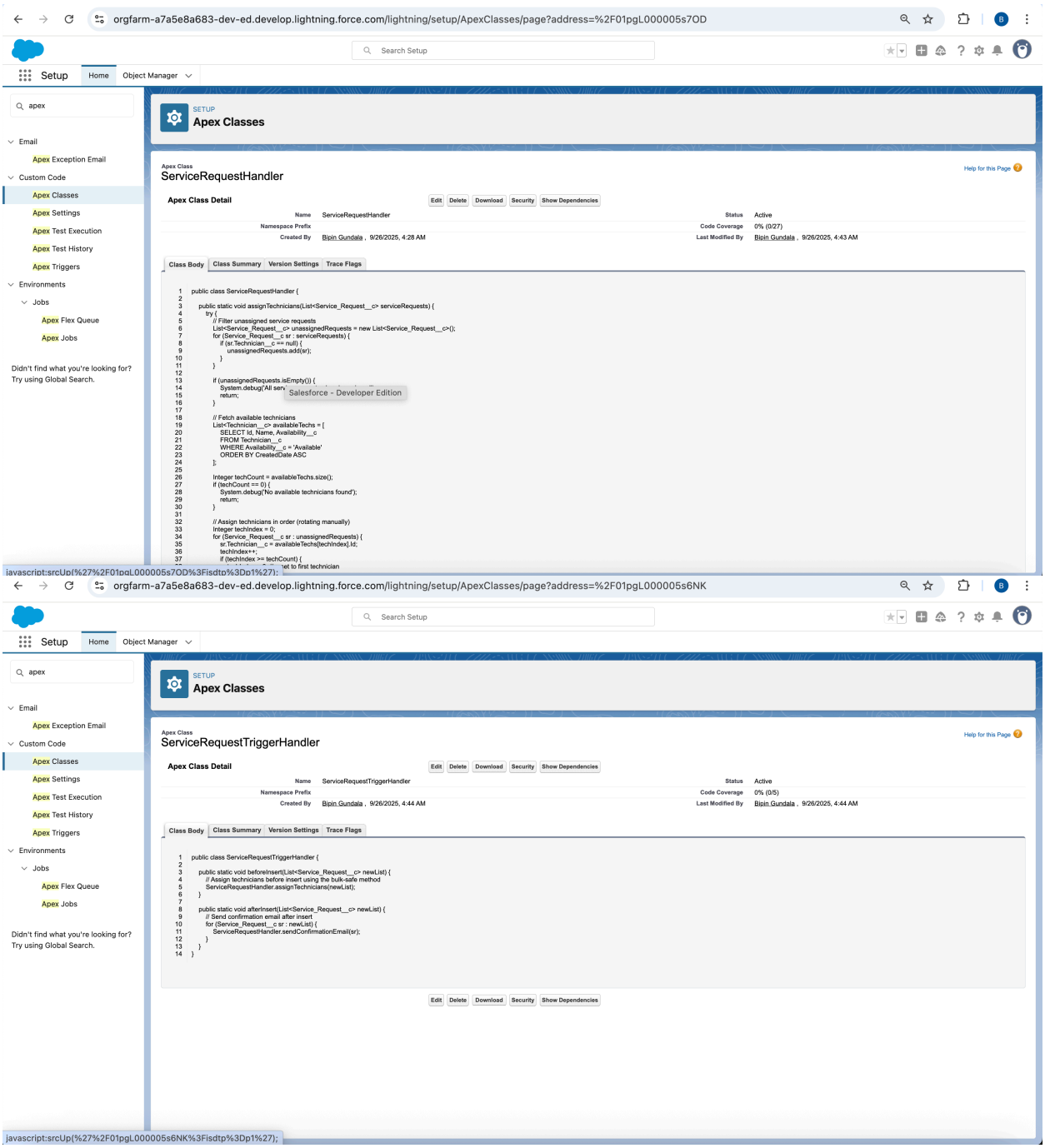
- Constructors, methods, and properties encapsulated in handler classes.
- Bulk-safe operations using `List<Records>` for triggers.
- Use of SOQL and collections (`List`, `Map`, `Set`) to handle multiple records efficiently.
- Exception handling with `try/catch` blocks for robustness.


orgfarm-a7a5e8a683-dev-ed.develop.lightning.force.com/lightning/setup/ApexClasses/page?address=%2F01pgL000005s7RR
Search Setup
Setup
Home
Object Manager
apex
Email
Apex Exception Email
Custom Code
Apex Classes
Apex Settings
Apex Test Execution
Apex Test History
Apex Triggers
Environments
Jobs
Apex Flex Queue
Apex Jobs
Didn't find what you're looking for?
Try using Global Search.
SETUP
Apex Classes
Apex Class
InvoiceProcessor
Help for this Page
Apex Class Detail
Edit
Delete
Download
Security
Show Dependencies
Name
InvoiceProcessor
Status
Active
Namespace Prefix
Code Coverage
0% (0/17)
Created By
Bipin Gundala , 9/26/2025, 4:29 AM
Last Modified By
Bipin Gundala , 9/26/2025, 4:32 AM
Class Body
Class Summary
Version Settings
Trace Flags
public class InvoiceProcessor {
// Method to get all invoices pending approval (SOQL + Collections)
public static List<Invoice__c> getPendingInvoices() {
try {
List<Invoice__c> invoices = [
SELECT Id, Nam
FROM Invoice__c
WHERE Status__c = 'Pending Approval'
LIMIT 100
];
Set<Id> invoiceIds = new Set<Id>();
for (Invoice__c inv : invoices) {
invoiceIds.add(inv.Id);
}
for (Invoice__c inv : invoices) {
if (inv.Amount__c > 10000) {
System.debug('High-value invoice found: ' + inv.Name);
} else {
System.debug('Normal invoice: ' + inv.Name);
}
}
System.debug('Unique Invoice Ids: ' + invoiceIds);
return invoices;
} catch (Exception ex) {
System.debug('Error fetching invoices: ' + ex.getMessage());
return new List<Invoice__c>();
}
}
// ✅ Add this method for validation
public static void validateAmount(List<Invoice__c> invoices) {
for (Invoice__c inv : invoices) {
if (inv.Amount__c == null || inv.Amount__c <= 0) {
inv.addError('Invoice amount must be greater than 0');
}
}
}
Salesforce - Developer Edition


orgfarm-a7a5e8a683-dev-ed.develop.lightning.force.com/lightning/setup/ApexClasses/page?address=%2F01pgL000005s7kn
Search Setup
Setup
Home
Object Manager
apex
Email
Apex Exception Email
Custom Code
Apex Classes
Apex Settings
Apex Test Execution
Apex Test History
Apex Triggers
Environments
Jobs
Apex Flex Queue
Apex Jobs
Didn't find what you're looking for?
Try using Global Search.
SETUP
Apex Classes
Apex Class
InvoiceTriggerHandler
Help for this Page
Apex Class Detail
Edit
Delete
Download
Security
Show Dependencies
Name
InvoiceTriggerHandler
Status
Active
Namespace Prefix
Code Coverage
0% (0/5)
Created By
Bipin Gundala , 9/26/2025, 4:32 AM
Last Modified By
Bipin Gundala , 9/26/2025, 4:32 AM
Class Body
Class Summary
Version Settings
Trace Flags
public class InvoiceTriggerHandler {
public static void beforeSave(List<Invoice__c> newList) {
// Validate invoice amounts
InvoiceProcessor.validateAmount(newList);
}
public static void afterInsert(List<Invoice__c> newList) {
// Submit for approval if required
for (Invoice__c inv : newList) {
InvoiceProcessor.submitForApproval(inv);
}
}
}
Edit
Delete
Download
Security
Show Dependencies
javascript:srcUp('%27%2F01pgL000005s7kn%3Fisdtp%3Dp1%27');


orgfarm-a7a5e8a683-dev-ed.develop.lightning.force.com/lightning/setup/ApexClasses/page?address=%2F01pgL000005s4Jt
Search Setup
Setup
Home
Object Manager
apex
Email
Apex Exception Email
Custom Code
Apex Classes
Apex Settings
Apex Test Execution
Apex Test History
Apex Triggers
Environments
Jobs
Apex Flex Queue
Apex Jobs
Didn't find what you're looking for?
Try using Global Search.
SETUP
Apex Classes
Apex Class
PreventiveMaintenanceNotifier
Help for this Page
Apex Class Detail
Edit
Delete
Download
Security
Show Dependencies
Name
PreventiveMaintenanceNotifier
Status
Active
Namespace Prefix
Code Coverage
0% (0/9)
Created By
Bipin Gundala , 9/26/2025, 3:53 AM
Last Modified By
Bipin Gundala , 9/26/2025, 3:53 AM
Class Body
Class Summary
Version Settings
Trace Flags
1 public class PreventiveMaintenanceNotifier implements Queueable {
2 public void execute(QueueableContext context) {
3 List<Service_Request__c> srList = [
4 SELECT Id, Customer__c, Last_Service_Date__c
5 FROM Service_Request__c
6 WHERE Last_Service_Date__c <= :Date.today().addDays(-30)
7];
8
9 for(Service_Request__c sr : srList){
10 Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
11 mail.setTargetObjectId(sr.Customer__c);
12 mail.setTemplateId([SELECT Id FROM EmailTemplate WHERE DeveloperName='Preventive_Maintenance_Reminder'].Id);
13 mail.setSaveAsActivity(false);
14 Messaging.sendEmail(new Messaging.SingleEmailMessage[] { mail });
15 }
16 }
17 }
Edit
Delete
Download
Security
Show Dependencies
javascript:srcUp(%27%2F01pgL000005s4Jt%3Fisdtp%3Dp1%27);


orgfarm-a7a5e8a683-dev-ed.develop.lightning.force.com/lightning/setup/ApexClasses/page?address=%2F01pgL000005s4Oj
Search Setup
Setup
Home
Object Manager
apex
Email
Apex Exception Email
Custom Code
Apex Classes
Apex Settings
Apex Test Execution
Apex Test History
Apex Triggers
Environments
Jobs
Apex Flex Queue
Apex Jobs
Didn't find what you're looking for?
Try using Global Search.
SETUP
Apex Classes
Apex Class
ScheduledPreventiveMaintenance
Help for this Page
Apex Class Detail
Edit
Delete
Download
Security
Show Dependencies
Name
ScheduledPreventiveMaintenance
Status
Active
Namespace Prefix
Code Coverage
0% (0/2)
Created By
Bipin Gundala , 9/26/2025, 3:54 AM
Last Modified By
Bipin Gundala , 9/26/2025, 3:54 AM
Class Body
Class Summary
Version Settings
Trace Flags
1 public class ScheduledPreventiveMaintenance implements Schedulable {
2 public void execute(SchedulableContext sc) {
3 System.enqueueJob(new PreventiveMaintenanceNotifier());
4 }
5 }
Edit
Delete
Download
Security
Show Dependencies
javascript:srcUp(%27%2F01pgL000005s4Oj%3Fisdtp%3Dp1%27);

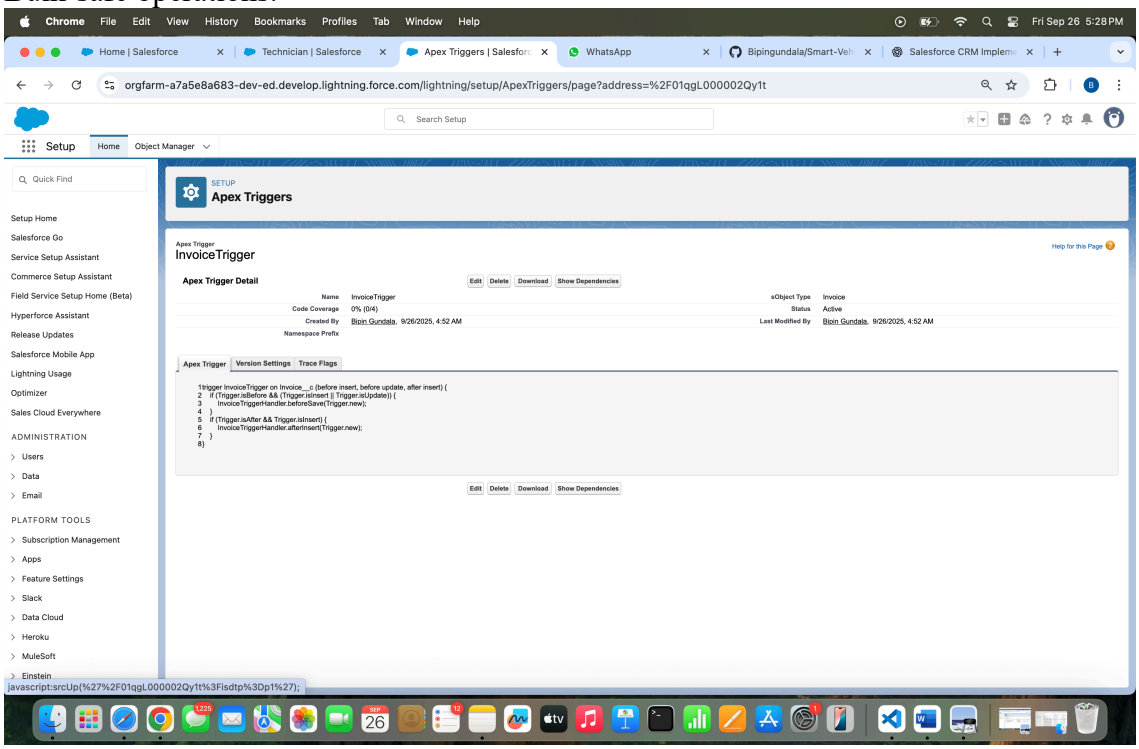


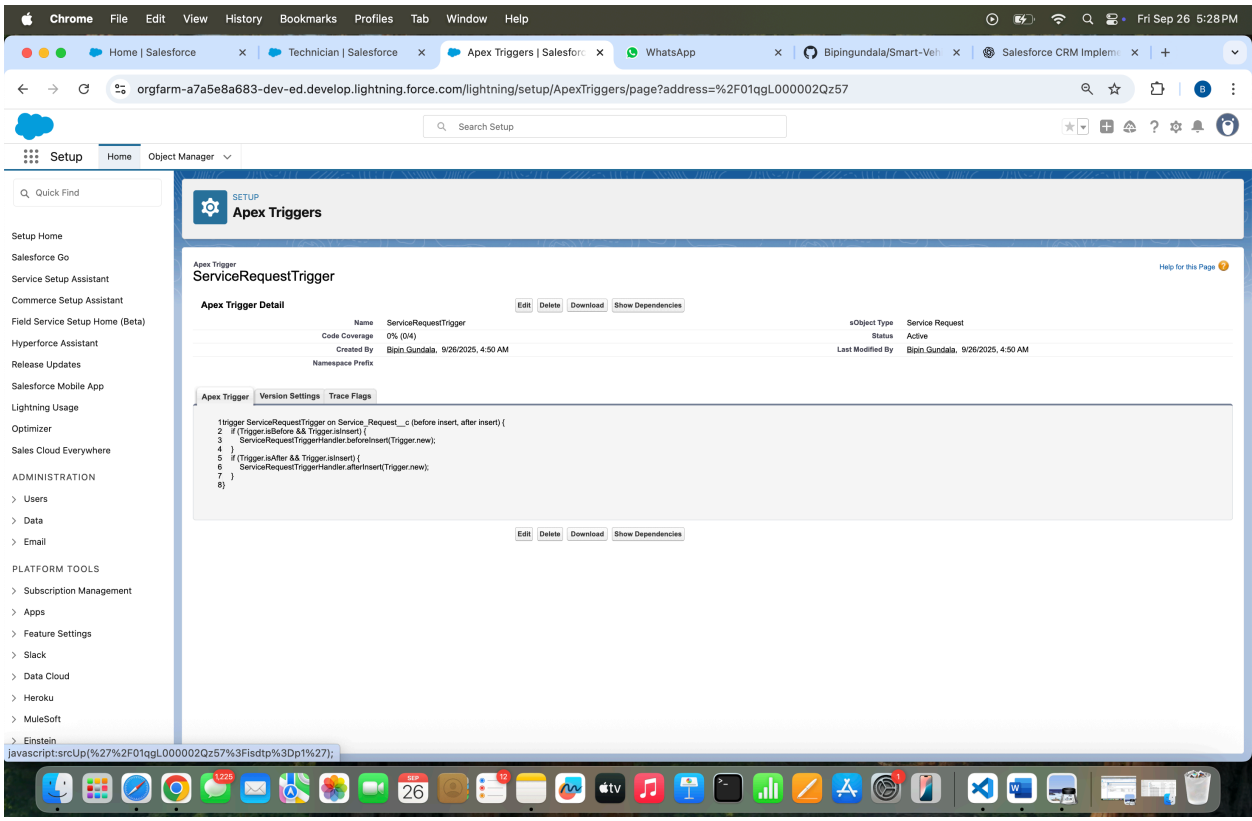
2. Apex Triggers

Triggers Implemented:

Trigger Name	Object	Event	Purpose
ServiceRequestTrigger	Service_Request__c	before insert, after insert	Assign technicians automatically (before insert) and send confirmation emails (after insert).
InvoiceTrigger	Invoice__c	before insert, before update, after insert	Validate invoice amounts (before insert/update) and submit for approval if necessary (after insert).

Trigger Design Pattern:

- **Handler Pattern**: All triggers delegate logic to Apex handler classes.
- Benefits:
  - Cleaner and modular code.
  - Easier maintenance and scalability.
  - Bulk-safe operations.
  - 

- 

---

3. SOQL & SOSL

Implemented Queries:

- **ServiceRequestHandler**:

```
List<Technician__c> availableTechs = [
    SELECT Id, Name, Availability__c
    FROM Technician__c
    WHERE Availability__c = 'Available'
    LIMIT 10
];
```

- **InvoiceProcessor**:

```
List<Invoice__c> invoices = [
    SELECT Id, Name, Amount__c, Status__c
    FROM Invoice__c
    WHERE Status__c = 'Pending Approval'
    LIMIT 100
];
```

- SOSL not used as search functionality was not required for this phase.

---

4. Collections

- **List**: Used to store records retrieved via SOQL for bulk operations.
- **Set**: Used to ensure unique record IDs, e.g., unique invoice IDs.
- **Map**: Used for fast access to records by ID, e.g., mapping technician IDs to their records.

---

5. Control Statements

- **Loops**: `for`, `while` loops used to iterate over records.
- **Conditionals**: `if/else` to handle business rules (e.g., checking technician availability, invoice amount thresholds).
- **Exception Handling**: `try/catch` used in handler classes to prevent runtime failures.

---

6. Asynchronous Processing

Implemented Asynchronous Apex:

Apex Type	Use Case
Scheduled Apex	`PreventiveMaintenanceNotifier` class runs monthly to send preventive maintenance reminders.
Queueable Apex	Send service request notifications asynchronously for better performance.
Future Methods	Can be used for external callouts (e.g., SMS or email notifications).
Batch Apex	Prepared for large volume processing of Service Requests or Invoices (not yet implemented in current project).

Scheduled Apex Example:

```
global class ScheduledPreventiveMaintenance implements Schedulable {
    global void execute(SchedulableContext sc) {
        PreventiveMaintenanceNotifier.sendReminders();
    }
}
```

---

7. Test Classes

Purpose: Required for production deployment; ensures code correctness and achieves 75%+ code coverage.

Implemented Tests:

- `ServiceRequestHandlerTest`: Tests technician auto-assignment and confirmation email logic.
- `InvoiceProcessorTest`: Tests invoice validation and approval submission logic.
- Positive & negative scenarios included for bulk operations.

---

8. Summary of Phase 5 Implementation

- All triggers use the **handler design pattern** for modular, maintainable code.
- Business logic is implemented in **Apex classes** rather than directly in triggers.
- Bulk-safe operations, exception handling, and asynchronous processes are implemented.
- Test classes created to validate all key functionalities and ensure deployment readiness.
- SOQL queries, collections, and control statements effectively used for automation and processing.