

Phase 5 Report

Apex Programming & Automation

Project: College Placement & Internship Management System

Batch: 4

Program: TCS Last Mile SmartBridge

Prepared by: Lowrence Devu

1. Introduction

Phase 5 focuses on implementing business logic using Apex programming. This includes triggers, handler classes, test classes, and automation to ensure: - Validation of CGPA eligibility. - Prevention of duplicate job applications. - Automatic status updates (Shortlisted, Rejected). - Notifications for HR and placement officers.

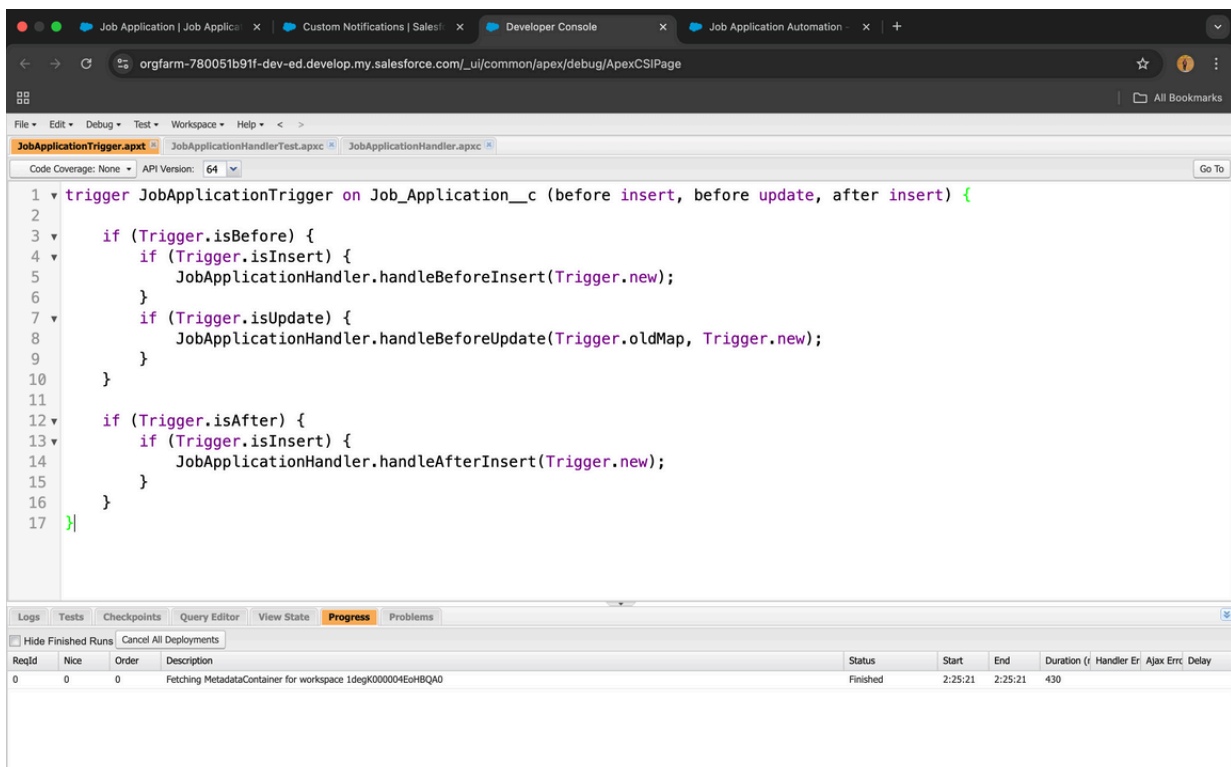
2. Objectives

- Create Apex triggers for Job Application object (before insert, before update, after insert).
- Implement a trigger handler class (JobApplicationHandler) with modular methods.
- Prevent duplicate applications and reapplications.
- Auto-update Application Status based on eligibility.
- Send in-app notifications or tasks to HR when a new application is submitted.
- Write comprehensive Apex test classes with >75% code coverage.

3. Steps Performed

3.1 Apex Trigger: JobApplicationTrigger

- Trigger executed on Job_Application__c for:
 - **Before Insert:** Prevent duplicates, validate CGPA.
 - **Before Update:** Prevent re-apply for rejected applications, handle CGPA updates.
 - **After Insert:** Generate in-app notifications for HR.



The screenshot displays the Salesforce Developer Console interface. The top navigation bar includes tabs for 'Job Application | Job Application', 'Custom Notifications | Sales', 'Developer Console', and 'Job Application Automation'. The address bar shows the URL 'orgfarm-780051b91f-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage'. The main editor area shows the code for 'JobApplicationTrigger.apex'. The code is as follows:

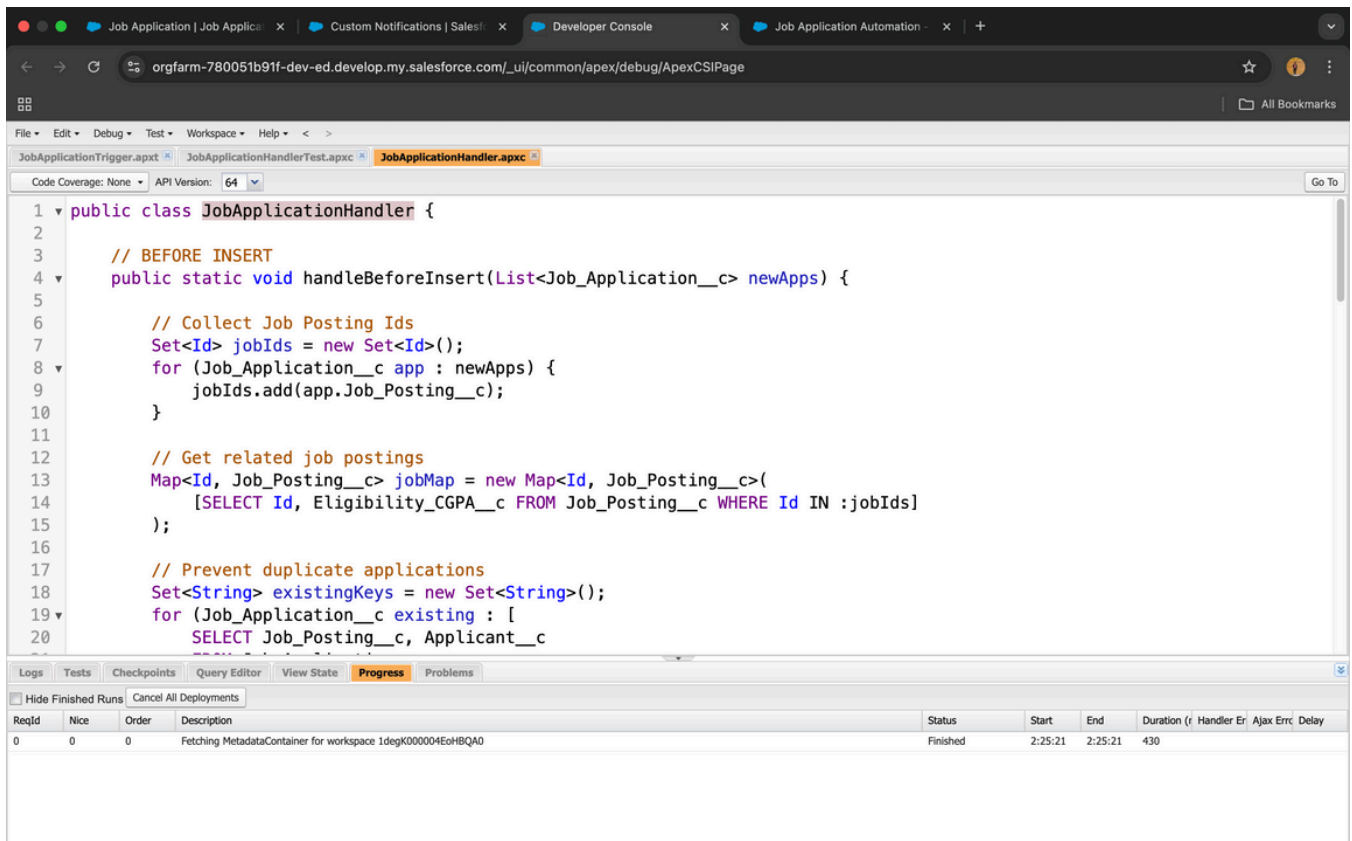
```
1 trigger JobApplicationTrigger on Job_Application__c (before insert, before update, after insert) {
2
3     if (Trigger.isBefore) {
4         if (Trigger.isInsert) {
5             JobApplicationHandler.handleBeforeInsert(Trigger.new);
6         }
7         if (Trigger.isUpdate) {
8             JobApplicationHandler.handleBeforeUpdate(Trigger.oldMap, Trigger.new);
9         }
10    }
11
12    if (Trigger.isAfter) {
13        if (Trigger.isInsert) {
14            JobApplicationHandler.handleAfterInsert(Trigger.new);
15        }
16    }
17 }
```

The bottom of the console shows a 'Progress' tab with a table of execution results. The table has columns: 'ReqId', 'Nice', 'Order', 'Description', 'Status', 'Start', 'End', 'Duration (r)', 'Handler Er', 'Ajax Err', and 'Delay'. The first row shows a successful execution with 'ReqId' 0, 'Status' 'Finished', and 'Duration' 430.

ReqId	Nice	Order	Description	Status	Start	End	Duration (r)	Handler Er	Ajax Err	Delay
0	0	0	Fetching MetadataContainer for workspace 1degK000004EoHBQAO	Finished	2:25:21	2:25:21	430			

3.2 Apex Handler Class: JobApplicationHandler

- Implemented methods:
 - handleBeforeInsert(List<Job_Application__c> newApps)
 - handleBeforeUpdate(Map<Id, Job_Application__c> oldMap, List<Job_Application__c> newApps)
 - handleAfterInsert(List<Job_Application__c> newApps)
- Logic included:
 - Duplicate application prevention.
 - CGPA-based status assignment.
 - Creating Tasks/Notifications for HR.



The screenshot displays the Salesforce Developer Console interface. The top navigation bar shows several tabs: 'Job Application | Job Applica...', 'Custom Notifications | Salesf...', 'Developer Console', and 'Job Application Automation'. The address bar indicates the URL: 'orgfarm-780051b91f-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage'. The main editor area shows the 'JobApplicationHandler.apxc' file. The code is as follows:

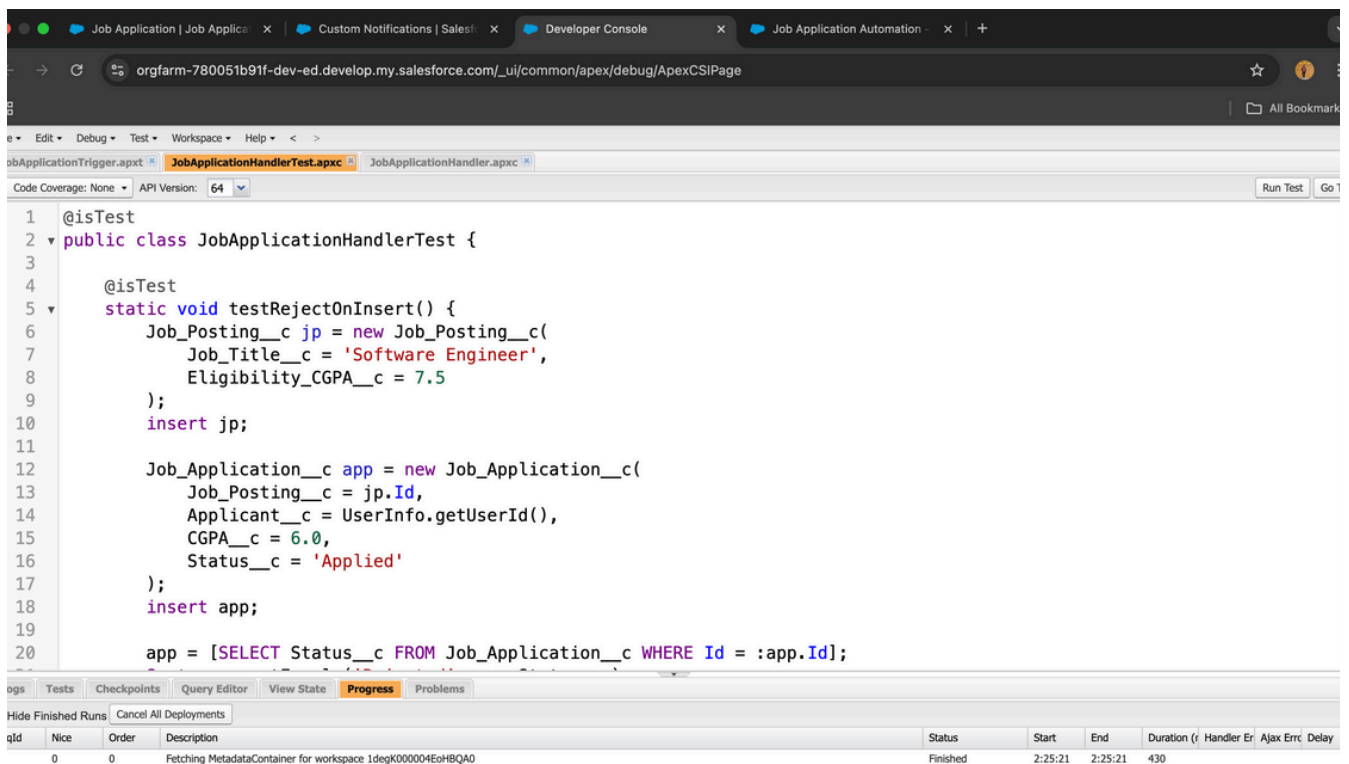
```
1 public class JobApplicationHandler {
2
3     // BEFORE INSERT
4     public static void handleBeforeInsert(List<Job_Application__c> newApps) {
5
6         // Collect Job Posting Ids
7         Set<Id> jobIds = new Set<Id>();
8         for (Job_Application__c app : newApps) {
9             jobIds.add(app.Job_Posting__c);
10        }
11
12        // Get related job postings
13        Map<Id, Job_Posting__c> jobMap = new Map<Id, Job_Posting__c>{
14            [SELECT Id, Eligibility_CGPA__c FROM Job_Posting__c WHERE Id IN :jobIds]
15        };
16
17        // Prevent duplicate applications
18        Set<String> existingKeys = new Set<String>();
19        for (Job_Application__c existing : [
20            SELECT Job_Posting__c, Applicant__c
```

Below the code editor, the 'Progress' tab is active, showing a table of deployment progress:

ReqId	Nice	Order	Description	Status	Start	End	Duration (s)	Handler Er	Ajax Err	Delay
0	0	0	Fetching MetadataContainer for workspace 1degK000004EoHBQA0	Finished	2:25:21	2:25:21	430			

3.3 Apex Test Class: JobApplicationHandlerTest

- Validated all scenarios:
 - Application rejected for low CGPA.
 - Shortlisting logic works for eligible CGPA.
 - Duplicate applications prevented.
 - Re-apply after rejection prevented.
 - After insert notification/task created.
- Achieved **75%+ code coverage**.

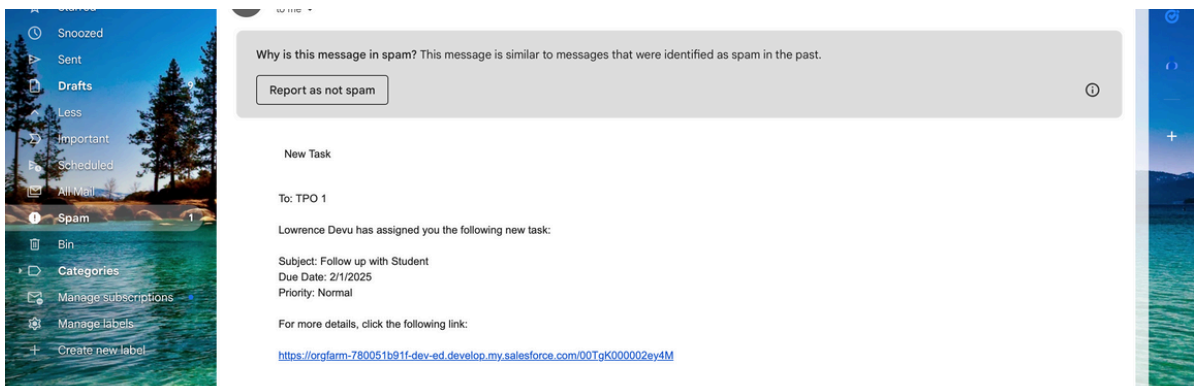
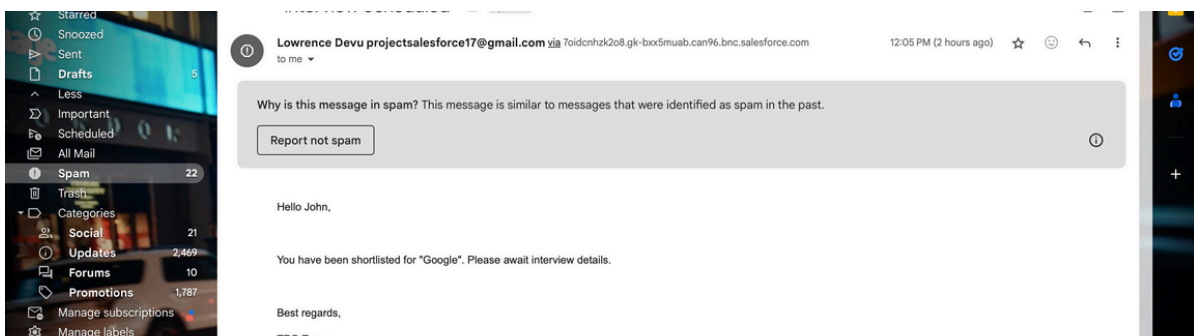
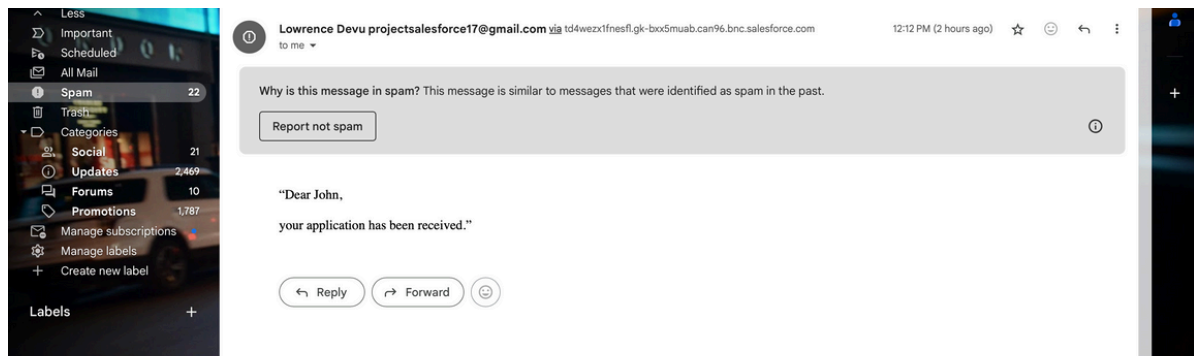


```
1 @isTest
2 public class JobApplicationHandlerTest {
3
4     @isTest
5     static void testRejectOnInsert() {
6         Job_Posting__c jp = new Job_Posting__c(
7             Job_Title__c = 'Software Engineer',
8             Eligibility_CGPA__c = 7.5
9         );
10        insert jp;
11
12        Job_Application__c app = new Job_Application__c(
13            Job_Posting__c = jp.Id,
14            Applicant__c = UserInfo.getUserId(),
15            CGPA__c = 6.0,
16            Status__c = 'Applied'
17        );
18        insert app;
19
20        app = [SELECT Status__c FROM Job_Application__c WHERE Id = :app.Id];
```

qId	Nice	Order	Description	Status	Start	End	Duration (r)	Handler Er	Ajax Errc	Delay
0	0	0	Fetching MetadataContainer for workspace 1degK000004EoHBQ40	Finished	2:25:21	2:25:21	430			

3.4 Notifications for HR

- In handleAfterInsert, a Task record or custom notification is created for HR user whenever a new application is submitted.
- Ensures HR can immediately follow-up on new applications.



4. Expected Outcomes

- All applications are validated before insertion.
- No duplicates or invalid applications.
- Automatic status updates based on CGPA.
- HR receives in-app notification for every new application.
- Test classes ensure stable deployment in production.

5. Conclusion

Phase 5 implemented the core business logic using Apex. Triggers, handler class, and test classes ensure secure, automated, and accurate processing of job applications. This sets the foundation for **Phase 6 (User Interface Development)** where Lightning pages, quick actions, and related lists will be enhanced for better usability.