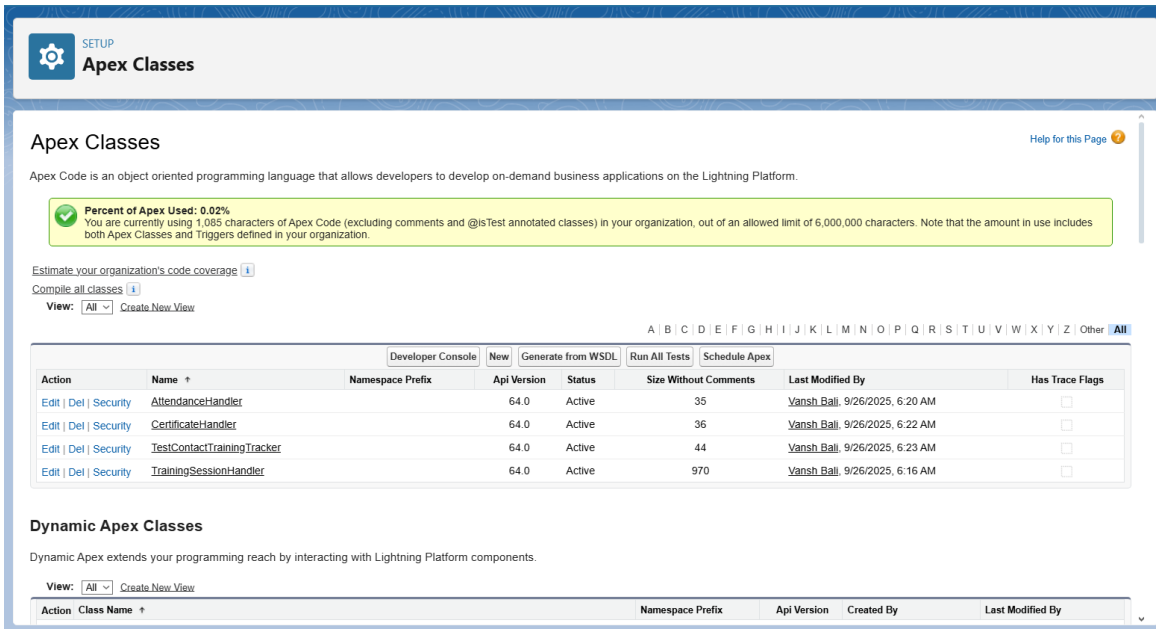
Phase 5 Report – Apex Trigger Implementation

1. Objective of Phase 5

The objective of Phase 5 is to automate and enforce business rules for the Contact Training Tracker using Apex triggers and handler classes. This implementation focuses on three key objects within the system:

- Training Session (Training_Session__c)
- Attendance (Attendance__c)
- Certificate (Certificate__c)

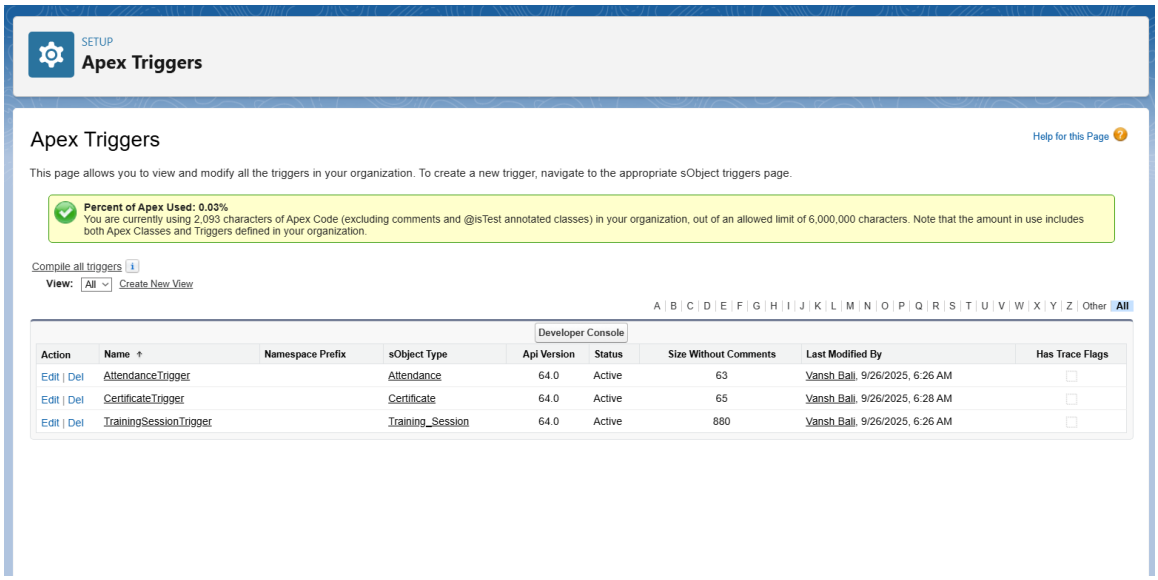
The outcome is improved data accuracy, streamlined automation (like attendance tracking and certification), and enhanced operational efficiency in managing training activities.



2. Trigger Design Pattern

A structured approach was applied for trigger development to ensure maintainability and scalability:

- Only one trigger per object to simplify management and reduce errors.
- Business logic is separated into handler classes instead of writing it directly in triggers.
- Centralized handler logic improves reusability, readability, testing, and debugging efficiency.



3. Implementation Steps

- Defined and created trigger files for each main object with contexts such as before insert, before update, after insert, and after delete.
- Developed handler classes to manage business rules, including:
  - Validation (e.g., preventing duplicate attendance records).
  - Pre-deletion logic (e.g., blocking deletion of sessions if certificates exist).
  - Post-insertion automation (e.g., auto-creating attendance records when trainees enroll).
  - Post-update automation (e.g., issuing certificates once a session is marked “Completed”).
- Created and organized metadata files for each Apex class for smooth deployment and Salesforce compliance.

4. Deployment Steps

- Organized triggers and handler classes within the Salesforce project structure using VS Code and Salesforce CLI.
- Deployed to Salesforce and verified trigger activation in Object Manager under the respective objects.
- Performed testing with sample data to validate that rules and automation execute correctly.

5. Expansion for Other Objects

The same design pattern can be extended to additional objects in the future, such as:

- Trainer (User) – automation for trainer notifications.
- Feedback (Feedback__c) – auto-creation of follow-up tasks when negative feedback is submitted.

This ensures consistent automation, scalability, and easy addition of future business logic.

6. Benefits of This Approach

- Clean and concise code with separation of concerns.
- Handler methods are reusable, reducing future development time.
- Easier and more efficient testing and debugging.
- Flexible system architecture to accommodate evolving training requirements.