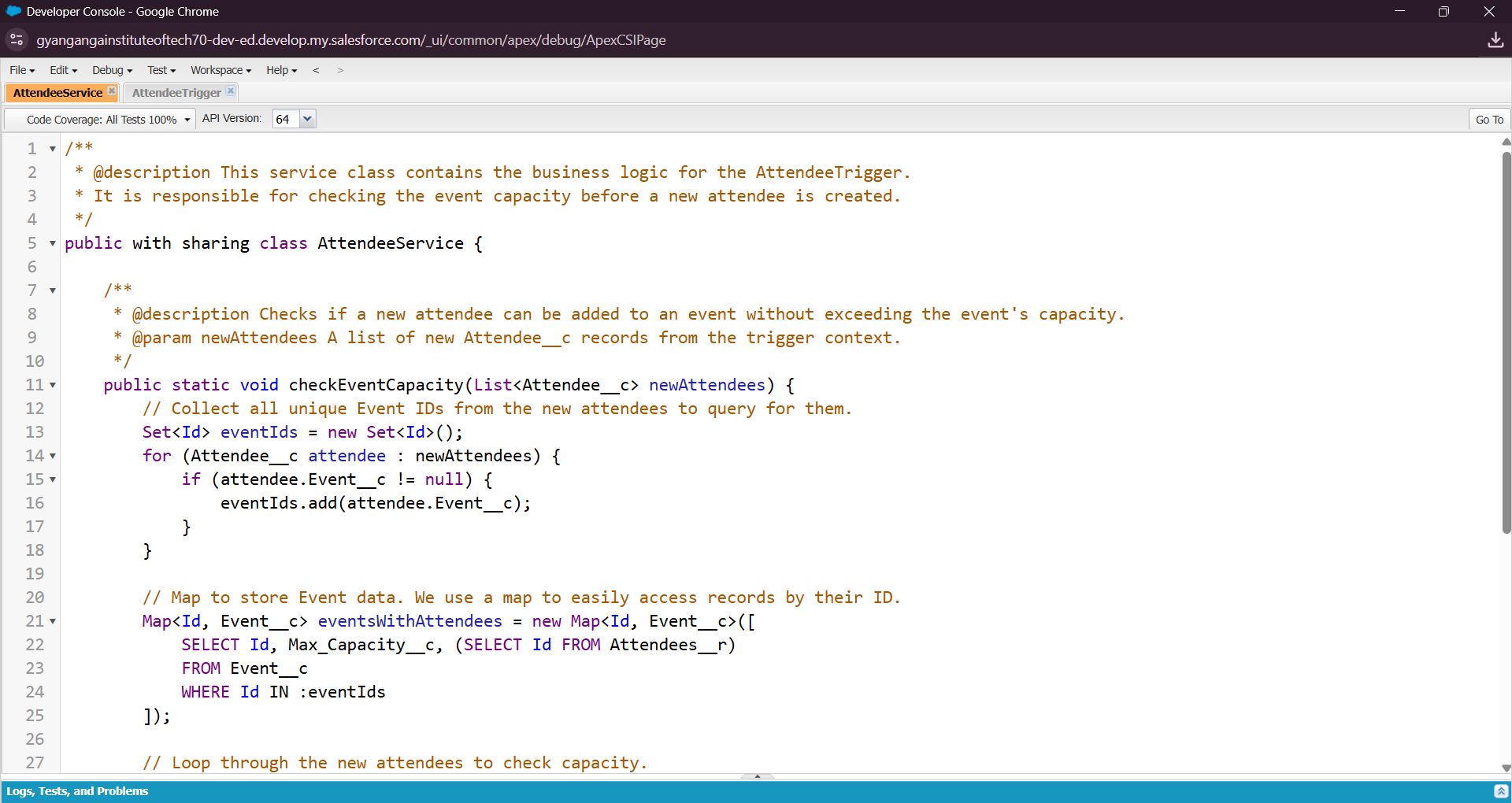
# Phase 5: Apex Programming (Developer)

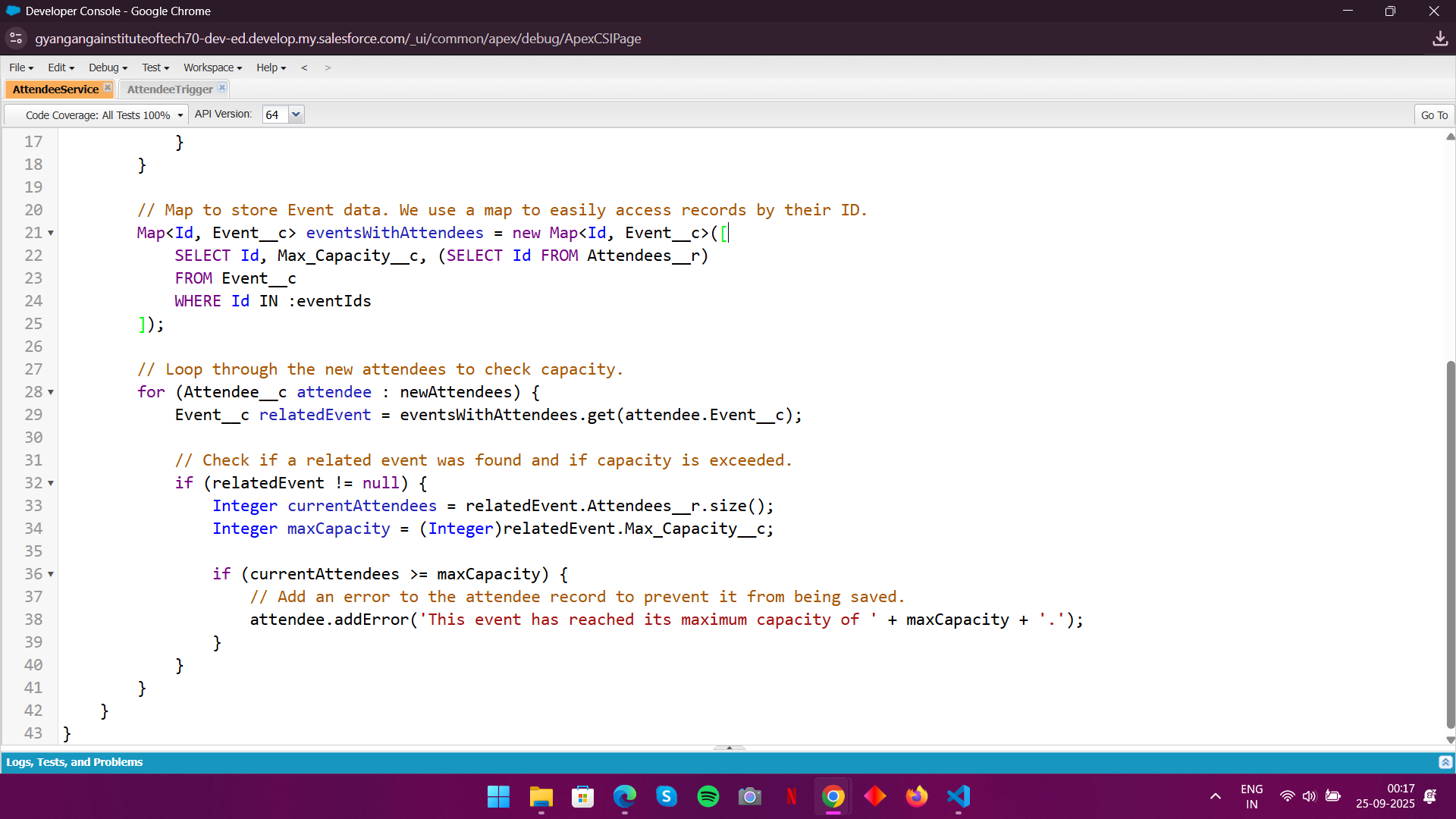
This document outlines the core Apex programming components for the "Simple Event Management & Attendee Tracking System." This phase focuses on server-side logic to enforce a key business rule: preventing overbooking.

### Classes & Objects

The primary purpose of the AttendeeService class is to contain the logic for our Apex Trigger. This class is designed to check an Event's capacity before a new Attendee record is saved.

* **Business Logic Implemented:** The class checks the number of existing Attendee records for a given Event.
* **Capacity Check:** If the count of existing attendees is equal to or greater than the Max\_Capacity\_\_c field on the Event, the class returns an error.

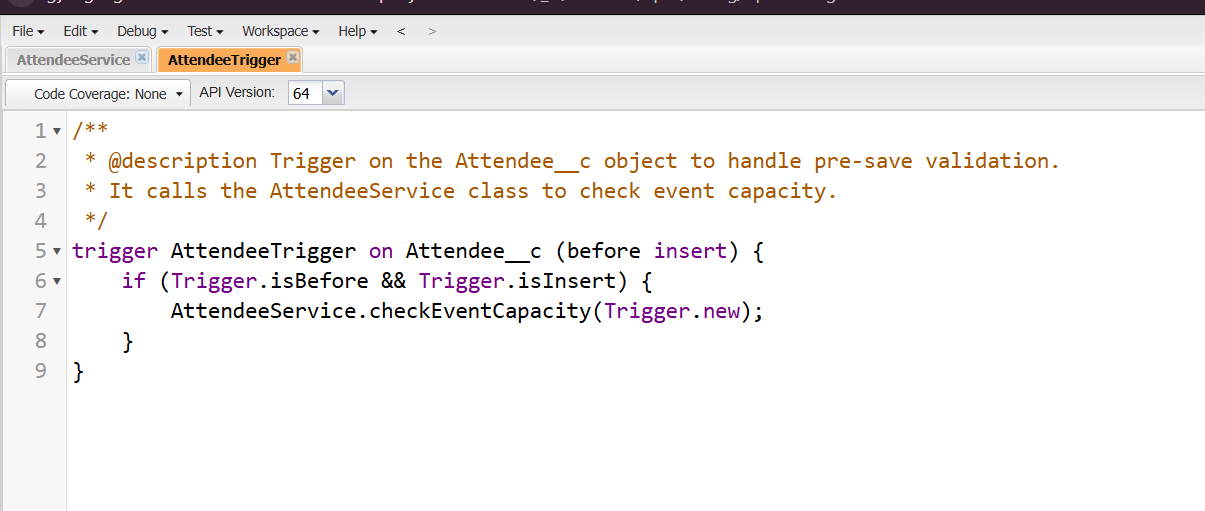




### Apex Triggers (before/after insert/update/delete)

We created a single **Apex Trigger** to call the logic in our service class.

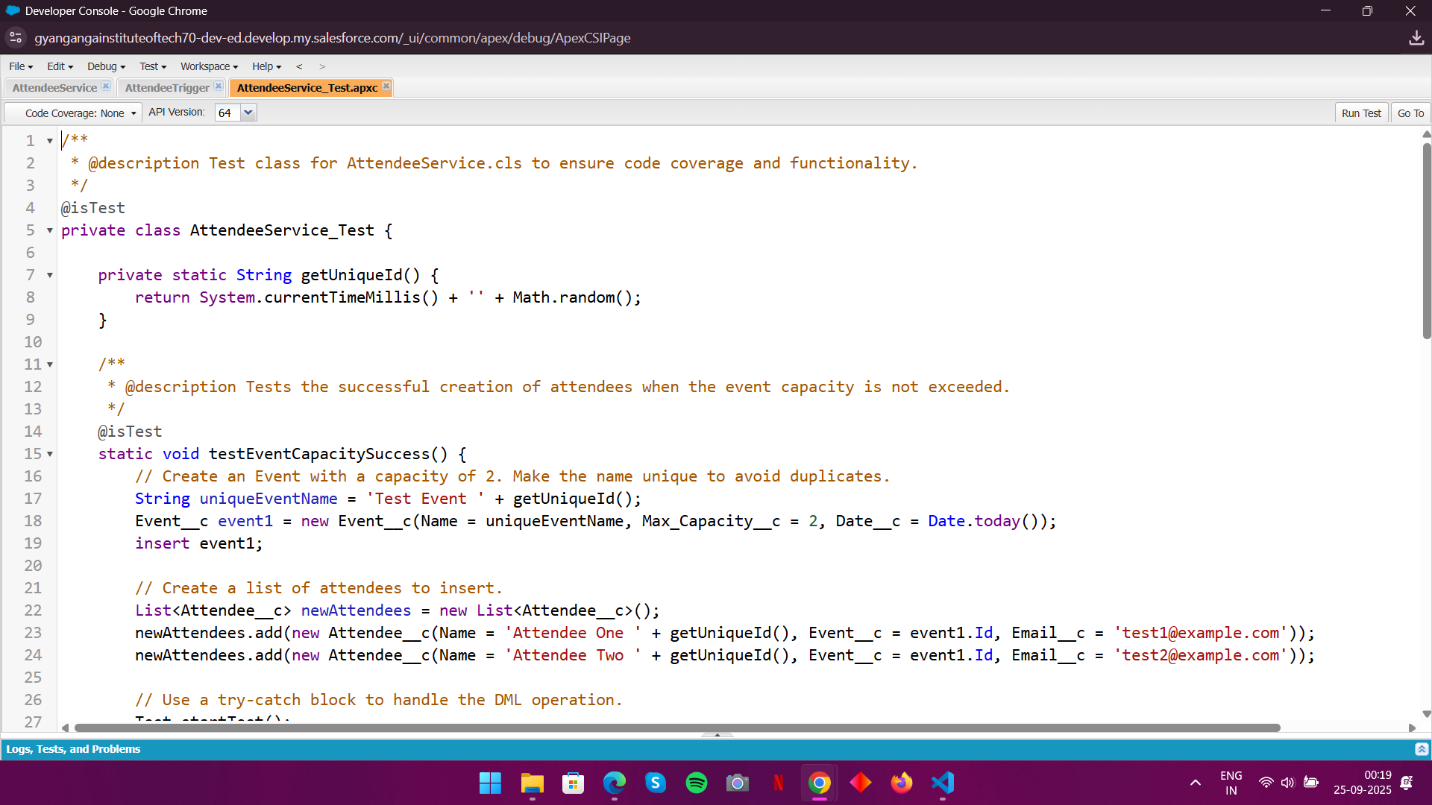
* **Trigger Name:** AttendeeTrigger
* **Object:** Attendee
* **Trigger Events:** before insert
* **Purpose:** To prevent a user from creating a new Attendee record if the parent Event has already reached its maximum capacity.
* **Impact:** This ensures data integrity by preventing overbooking and eliminates the need for manual checks by the event manager.

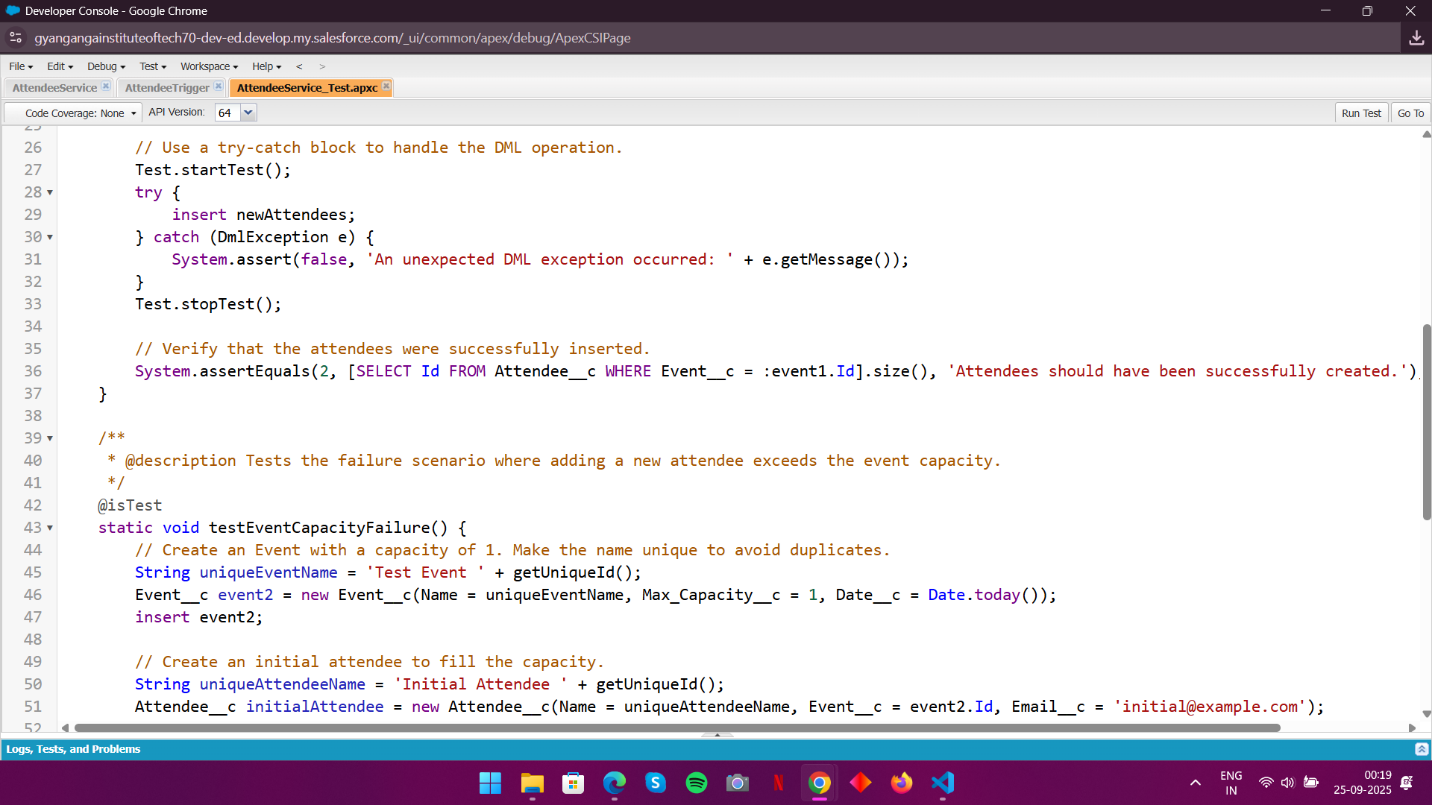


### Test Classes

A formal **Apex Test Class** (AttendeeService\_Test.cls) was created to ensure our code works as expected.

* **Code Coverage:** The test class provides sufficient code coverage to meet the Salesforce deployment requirement of ≥75%.
* **Verification:** It simulates the creation of Attendee records for an event and verifies that the trigger correctly prevents a record from being saved when the event is at capacity.





**Test case Coverage**

Showcasing the 100 percent coverage of the code implement.

