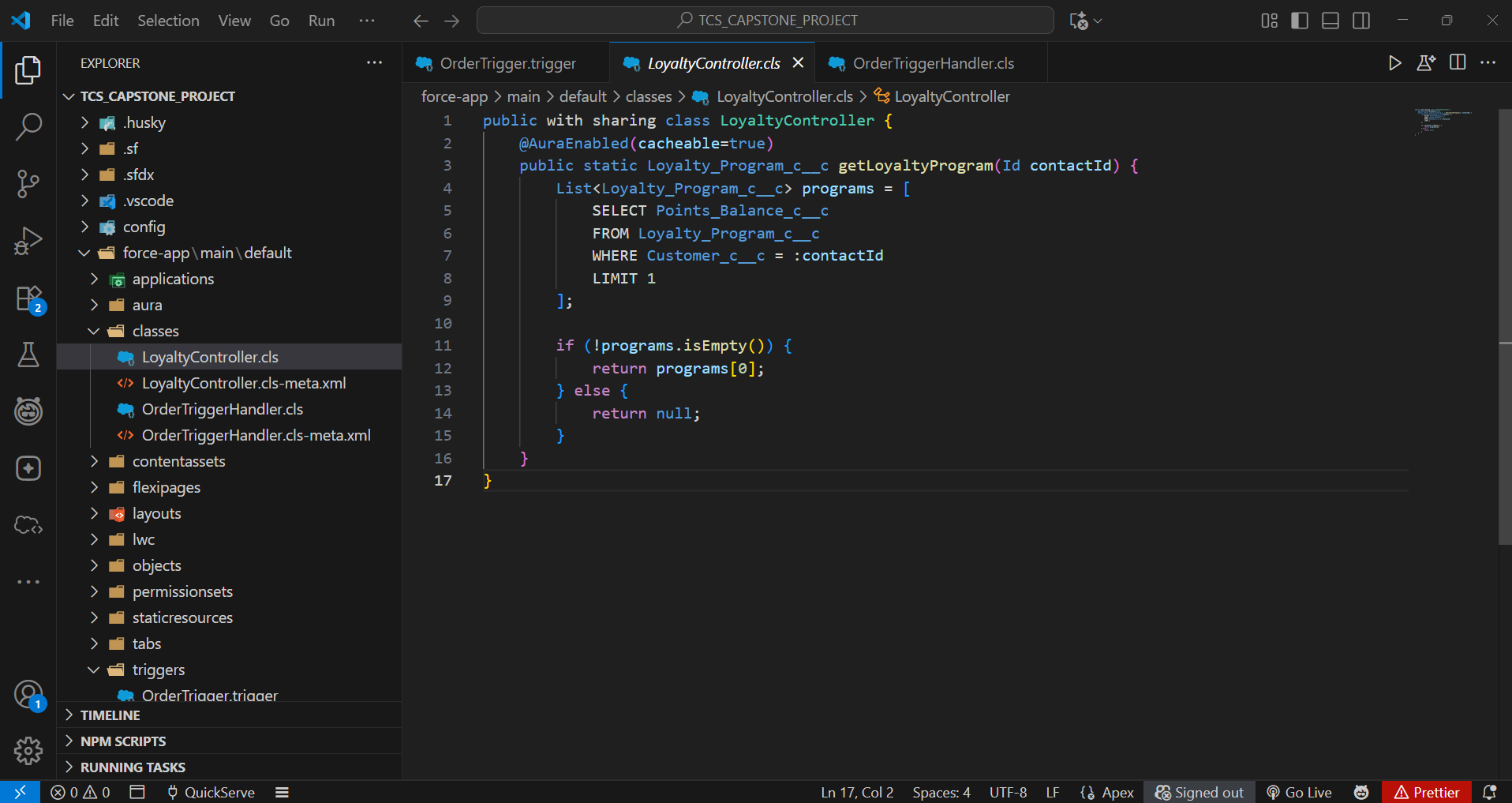
**PHASE 5 - Apex Programming (Developer)**

This phase covers the custom backend logic built with Apex to handle the core requirements of the loyalty points calculation, which cannot be achieved with declarative tools alone.

**➡️ Classes & Objects**

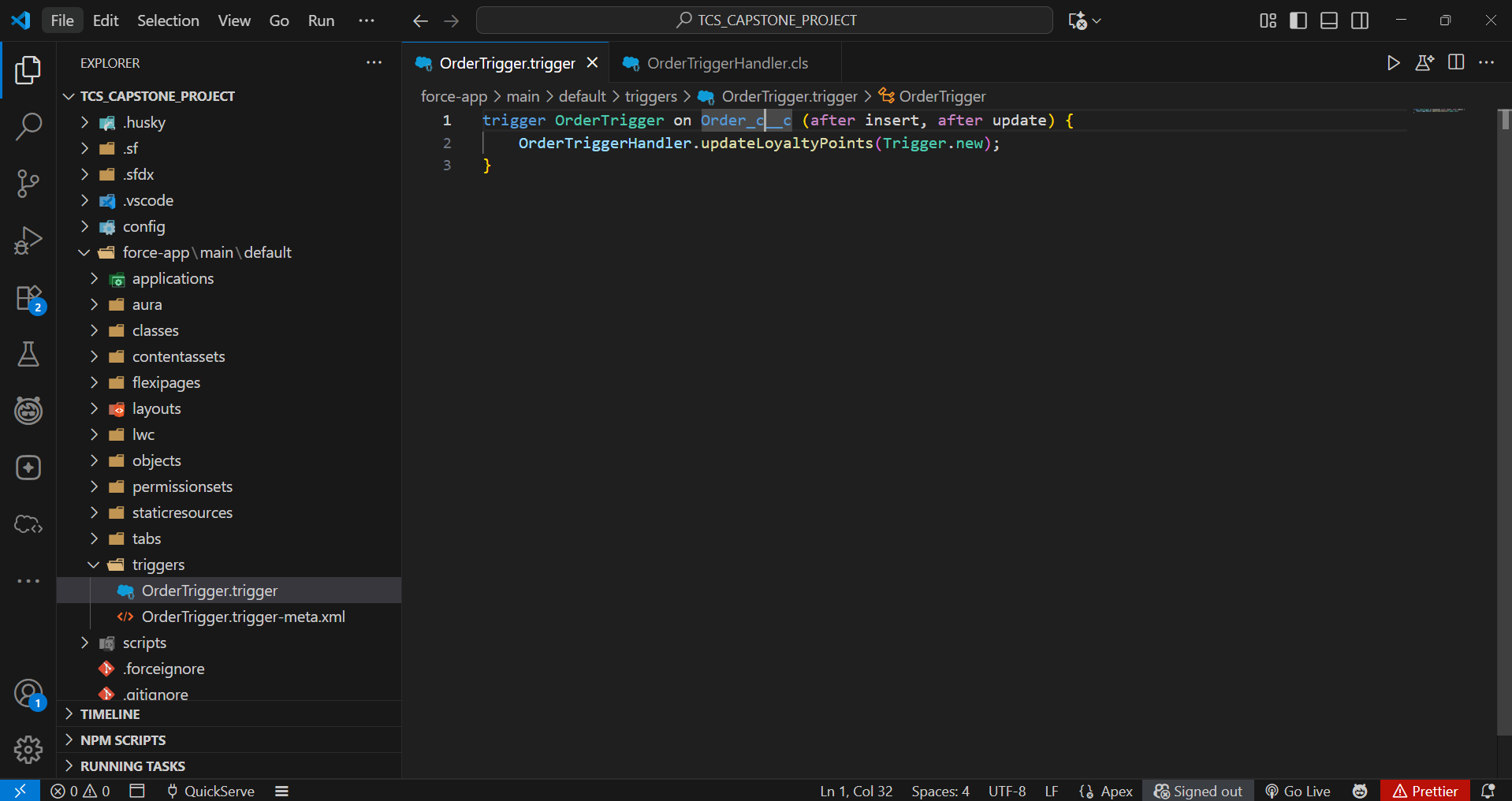
An Apex class LoyaltyController was created to serve data to the Lightning Web Component. This class interacts with our custom SObjects (Loyalty\_Program\_\_c, Order\_\_c) to query and return data.



**➡️ Apex Triggers**

The core automation is initiated by an Apex trigger.

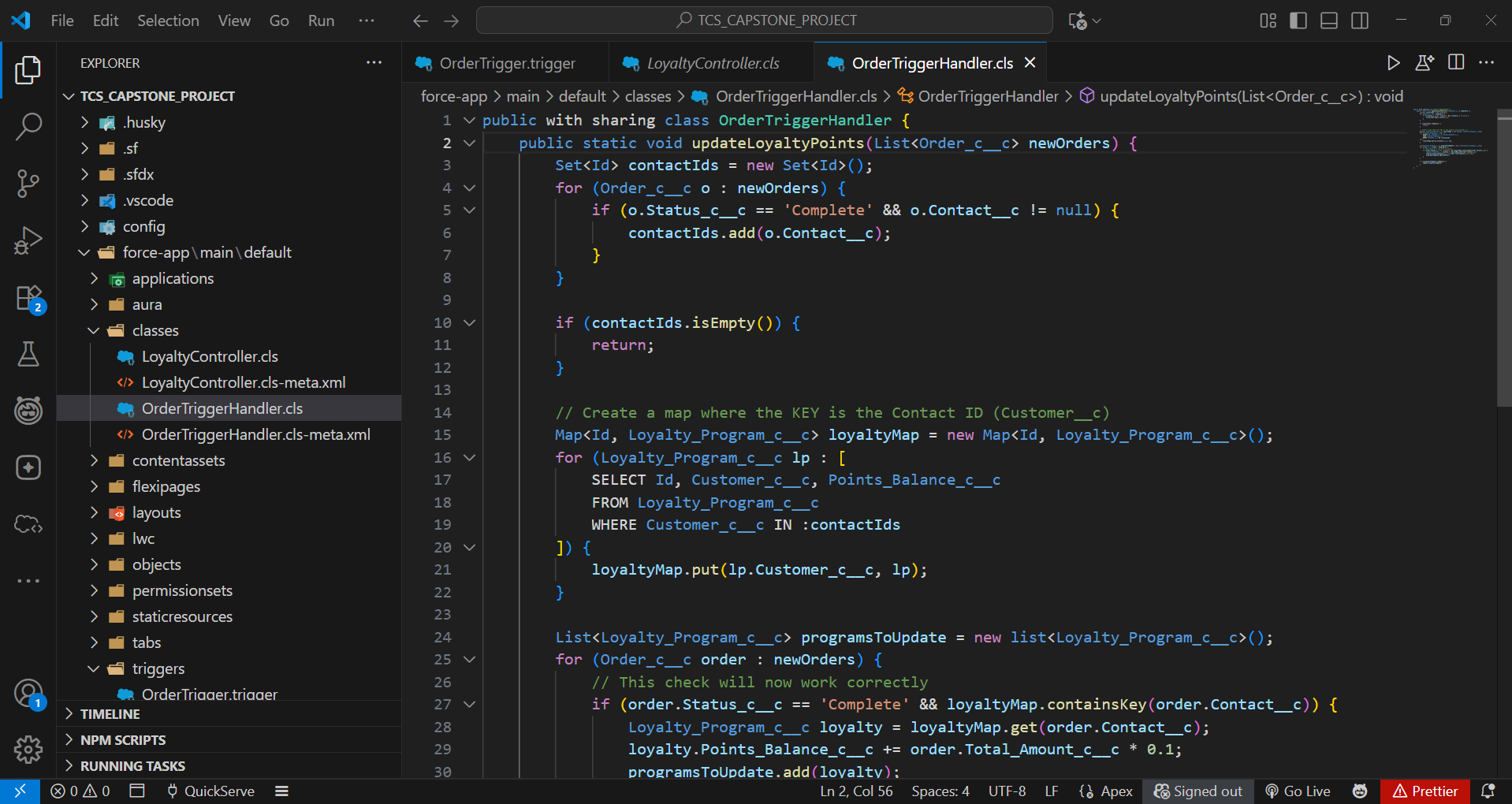
* **Trigger**: An OrderTrigger was created to run on the Order\_\_c object.
* **Events**: It is configured to fire after insert and after update to ensure the logic runs whenever an order is created or its status changes.



**➡️ Trigger Design Pattern**

A handler pattern was used to keep the trigger logic-less.

* The OrderTrigger does not contain any business logic itself.
* It calls a method in a separate handler class, OrderTriggerHandler, passing the trigger context variables (Trigger.new, Trigger.oldMap). This makes the code cleaner, more reusable, and easier to test.



**➡️ SOQL & Collections**

The Apex logic uses SOQL to query for records and Collections to manage the data efficiently.

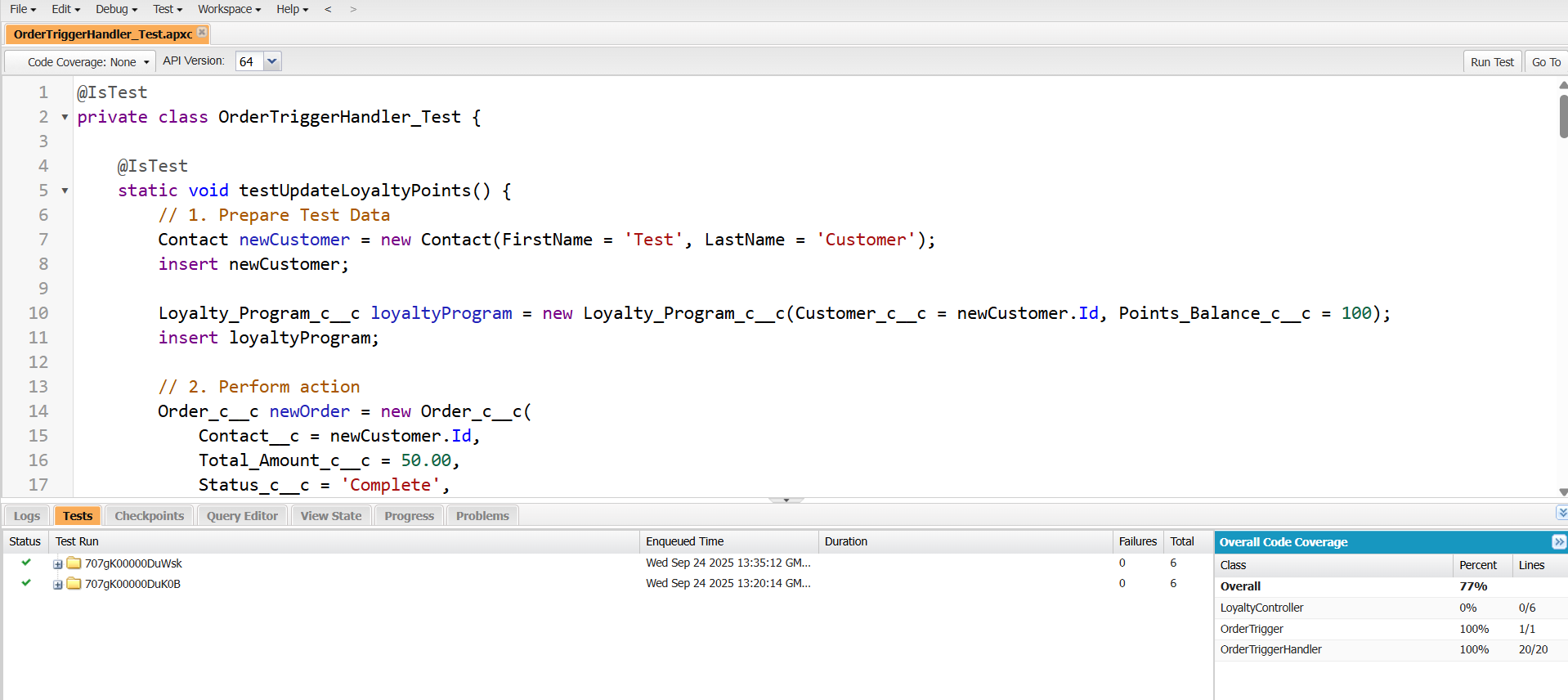
* **SOQL**: A SOQL query is used within the OrderTriggerHandler to select the Loyalty\_Program\_\_c record related to the customer on the Order\_\_c.
  + Example: SELECT Id, Points\_Balance\_\_c FROM Loyalty\_Program\_\_c WHERE Customer\_\_c = :contactId
* **Collections**:
  + A **Set** is used to collect unique Contact IDs from the orders to avoid duplicate queries.
  + A **Map** is used to hold the queried Loyalty\_Program\_\_c records, with the Contact ID as the key for easy retrieval.

**➡️ Exception Handling**

Try-Catch blocks are implemented within the Apex handler to gracefully manage potential errors, such as a query returning no records or a DML operation failing. This prevents the entire transaction from failing due to an unexpected issue.

**➡️ Test Classes**

A dedicated test class, OrderTriggerHandler\_Test, was created to validate the Apex logic.



* The test class creates all necessary data in isolation (Contacts, Orders, Loyalty Programs).
* It calls the handler method to simulate the trigger firing.
* **System.assertEquals()** is used to assert that the Points\_Balance\_\_c on the Loyalty\_Program\_\_c record was updated correctly after the order was set to 'Complete'.
* The class achieves over 75% code coverage, making the Apex trigger and handler ready for deployment.