

Phase 5 – Apex Programming (Developer)

Objective:

Implement basic Apex programming in Salesforce to automate Event revenue calculation and demonstrate Apex logic for documentation purposes. This includes an **Apex Trigger** and an **Apex Class**.

5.1 Apex Class – EventRevenueHandler

Class Name: EventRevenueHandler

Purpose:

Handles calculation and update of total revenue for Events based on Ticket bookings.

Class Code:

```
public class EventRevenueHandler {

    public static void updateEventRevenue(List<Ticket__c> tickets){

        // Map to store total revenue per Event

        Map<Id, Decimal> eventRevenueMap = new Map<Id, Decimal>();

        // Sum ticket prices per Event
        for(Ticket__c t : tickets){

            if(t.Event_record__c != null && t.Status__c == 'Booked'){

                if(!eventRevenueMap.containsKey(t.Event_record__c)){

                    eventRevenueMap.put(t.Event_record__c, 0);

                }

                eventRevenueMap.put(t.Event_record__c, eventRevenueMap.get(t.Event_record__c) +
t.Price__c);

            }

        }

        // Update Event records

        List<Event_record__c> eventsToUpdate = new List<Event_record__c>();

        for(Id eventId : eventRevenueMap.keySet()){

            eventsToUpdate.add(new Event_record__c(

                Id = eventId,

                Total_Revenue__c = eventRevenueMap.get(eventId)

            ));

        }

    }

}
```

```

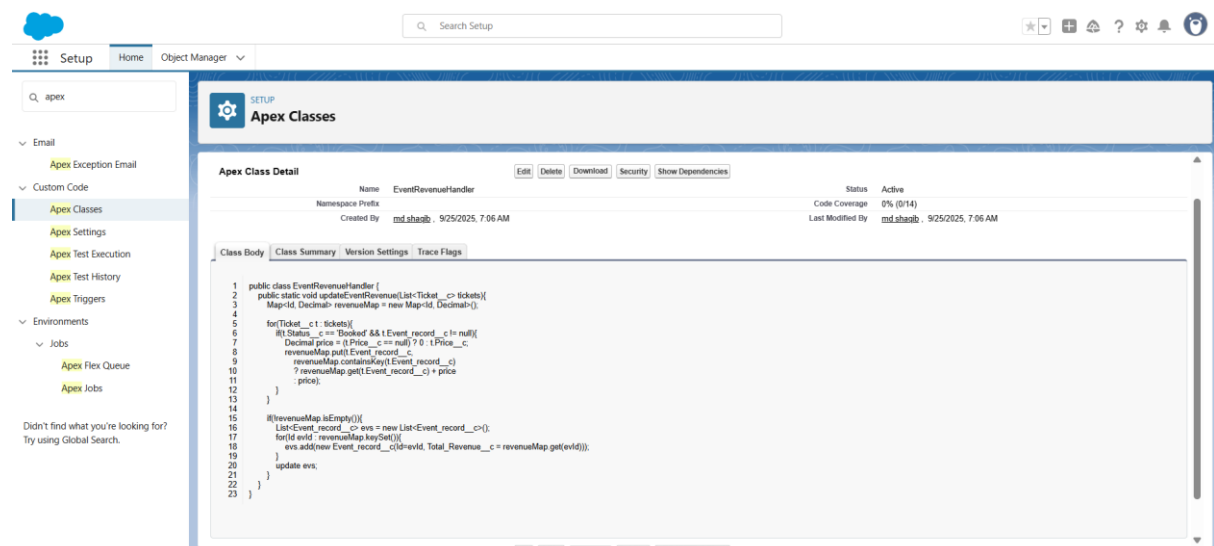
if(!eventsToUpdate.isEmpty()){
    update eventsToUpdate;
}
}
}

```

Explanation:

- Loops through Tickets
- Sums Price__c for booked tickets per Event
- Updates Total_Revenue__c field on Event

Screenshot:



5.2 Apex Trigger – TicketTrigger

Trigger Name: TicketTrigger

Object: Ticket__c

Trigger Events: after insert, after update

Purpose:

Calls the Apex class to update Event revenue whenever Tickets are booked.

Trigger Code:

trigger TicketTrigger on Ticket__c (after insert, after update) {

```

    List<Ticket__c> bookedTickets = new List<Ticket__c>();

```

```

for(Ticket__c t : Trigger.new){
Ticket__c oldT = Trigger.isInsert ? null : Trigger.oldMap.get(t.Id);

    if(t.Status__c == 'Booked' && (Trigger.isInsert || (oldT.Status__c != 'Booked'))){

        if(t.Event_record__c != null) bookedTickets.add(t);

    }

}

if(!bookedTickets.isEmpty()){

    EventRevenueHandler.updateEventRevenue(bookedTickets);

}

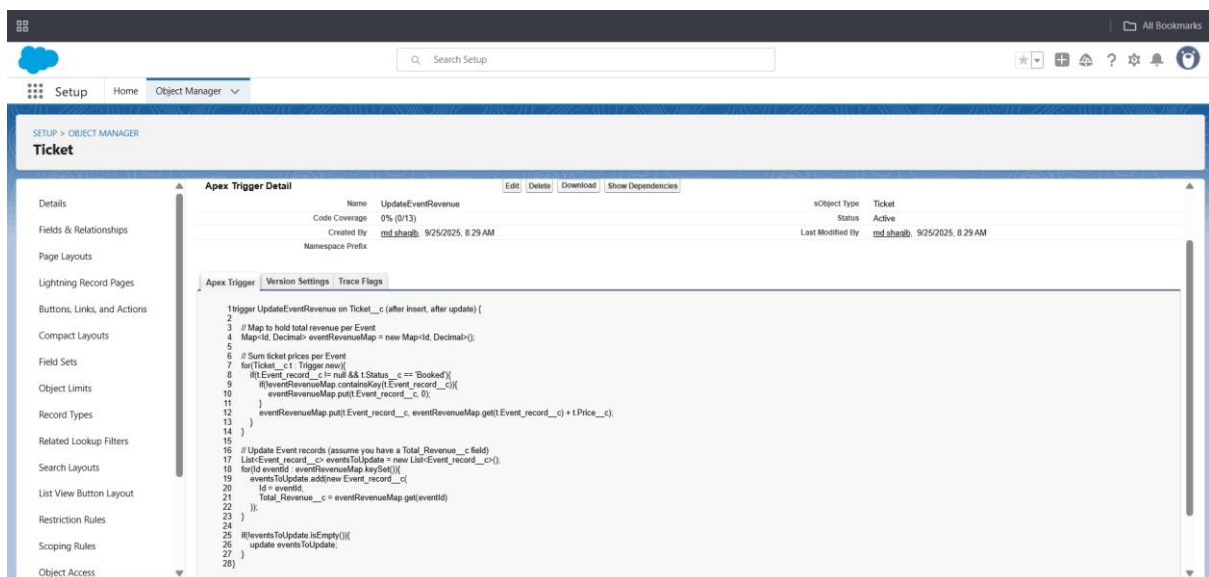
}

```

Explanation:

- Checks if Ticket is booked
- Calls EventRevenueHandler.updateEventRevenue
- Ensures only newly booked tickets trigger revenue update

Screenshot:



5.3 Test Apex – Create Event & Tickets

Purpose:

Demonstrates trigger and class working together. Creates Event and Tickets, then verifies Event revenue.

Execute Anonymous Snippet:

```
Event_record__c testEvent;
```

```
List<Event_record__c> events = [SELECT Id, Name, Total_Revenue__c FROM Event_record__c WHERE  
Name = 'Test Event' LIMIT 1];
```

```
if(events.isEmpty()){
```

```
    testEvent = new Event_record__c(
```

```
        Name = 'Test Event',
```

```
        Event_Date__c = Date.today(),
```

```
        Location__c = 'Hall A',
```

```
        Event_Type__c = 'Corporate',
```

```
        Number_of_Attendees__c = 100,
```

```
        Description__c = 'This is a test event',
```

```
        Total_Revenue__c = 0
```

```
    );
```

```
    insert testEvent;
```

```
} else {
```

```
    testEvent = events[0];
```

```
}
```

```
Ticket__c ticket1 = new Ticket__c(
```

```
    Name = 'Ticket 1',
```

```
    Event_record__c = testEvent.Id,
```

```
    Status__c = 'Booked',
```

```
    Price__c = 100,
```

```
    Attendee_Name__c = 'John Doe',
```

```
    Email__c = 'john.doe@example.com'
```

```
);
```

```
Ticket__c ticket2 = new Ticket__c(
```

```
    Name = 'Ticket 2',
```

```

Event_record__c = testEvent.Id,

Status__c = 'Booked',

Price__c = 150,

Attendee_Name__c = 'Jane Smith',

Email__c = 'jane.smith@example.com'

);

List<Ticket__c> ticketsToInsert = new List<Ticket__c>{ticket1, ticket2};

insert ticketsToInsert;

// -----

// Verify Event Revenue

// -----

testEvent = [SELECT Id, Name, Total_Revenue__c FROM Event_record__c WHERE Id = :testEvent.Id];

System.debug('Event Total Revenue after inserting tickets: ' + testEvent.Total_Revenue__c);

```

Explanation:

- Creates a test Event (if not exists)
- Creates 2 booked Tickets linked to that Event
- Trigger calls the class to calculate and update **Total_Revenue__c** automatically
- Debug log confirms revenue update

Screenshot :

Timestamp	Event	Details
20:56:25:001	USER_INFO	[EXTERNAL]005f900005s2t1md.shaqb.cs22643@agentforce.com[(GMT-07:00) Pacific Daylight Time (America/Los_Angeles)]GMT-07:00
20:56:25:001	EXECUTION_ST...	
20:56:25:001	CODE_UNIT_ST...	[EXTERNAL]execute_anonymous_apex
20:56:25:001	VARIABLE_SCO...	[7]events[1]List<Event_record__c>[true]false
20:56:25:001	VARIABLE_SCO...	[4]testEvent[Event_record__c]truefalse
20:56:25:001	VARIABLE_SCO...	[26]ticket1[Ticket__c]truefalse
20:56:25:001	VARIABLE_SCO...	[35]ticket2[Ticket__c]truefalse
20:56:25:001	VARIABLE_SCO...	[45]ticketsToInsert[List<Ticket__c>]truefalse
20:56:25:001	HEAP_ALLOCATE	[95]Bytes:3
20:56:25:001	HEAP_ALLOCATE	[100]Bytes:152
20:56:25:001	HEAP_ALLOCATE	[417]Bytes:408
20:56:25:001	HEAP_ALLOCATE	[430]Bytes:408
20:56:25:001	HEAP_ALLOCATE	[317]Bytes:6
20:56:25:001	HEAP_ALLOCATE	[EXTERNAL]Bytes:86
20:56:25:002	STATEMENT_EX...	[1]
20:56:25:002	STATEMENT_EX...	[4]
20:56:25:002	VARIABLE_ASS...	[4]testEvent[null]
20:56:25:002	STATEMENT_EX...	[7]

User	Application	Operation	Time	Status	Read	Size
md shaqb	Browser	/setup/build/editApexTrigger.apexp	9/25/2025, 8:59:31 PM	Success	Unread	1.16 KB
md shaqb	Unknown	/services/data/v44.0/tooling/executeA...	9/25/2025, 8:56:25 PM	Success		14 KB