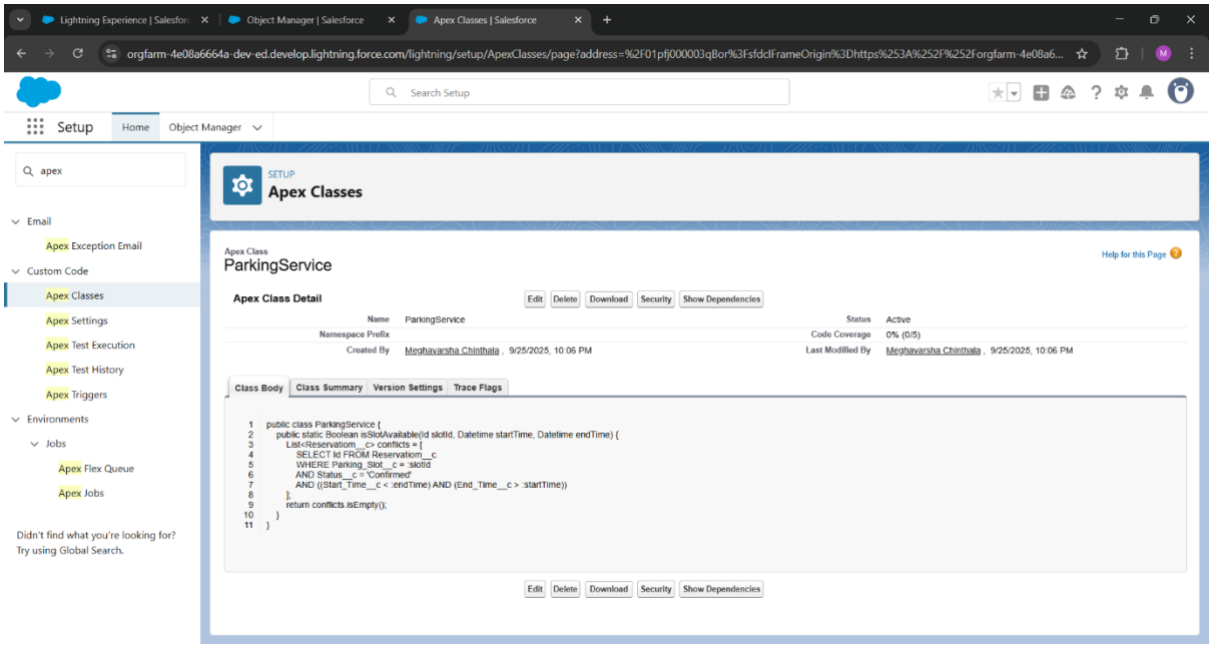
**Phase 5 – Apex Programming (Developer)**

🔷 **Step 1: Create ParkingService Class**

**Procedure:**

1. Go to **Setup → Apex Classes → New**.
2. Name it ParkingService.
3. Paste the following code:

```
public with sharing class ParkingService {


   // Method to get available slots within given time

   public static List<Parking_Slot__c> getAvailableSlots(DateTime startTime,
DateTime endTime) {

      return [SELECT Id, Name, Status__c

            FROM Parking_Slot__c

            WHERE Status__c = 'Available'

            AND Id NOT IN (

               SELECT Slot__c FROM Reservation__c

               WHERE (Start_Time__c < :endTime AND End_Time__c > :startTime)

            )];

   }

}
```




**Screenshot of ParkingService Class Creation**

---

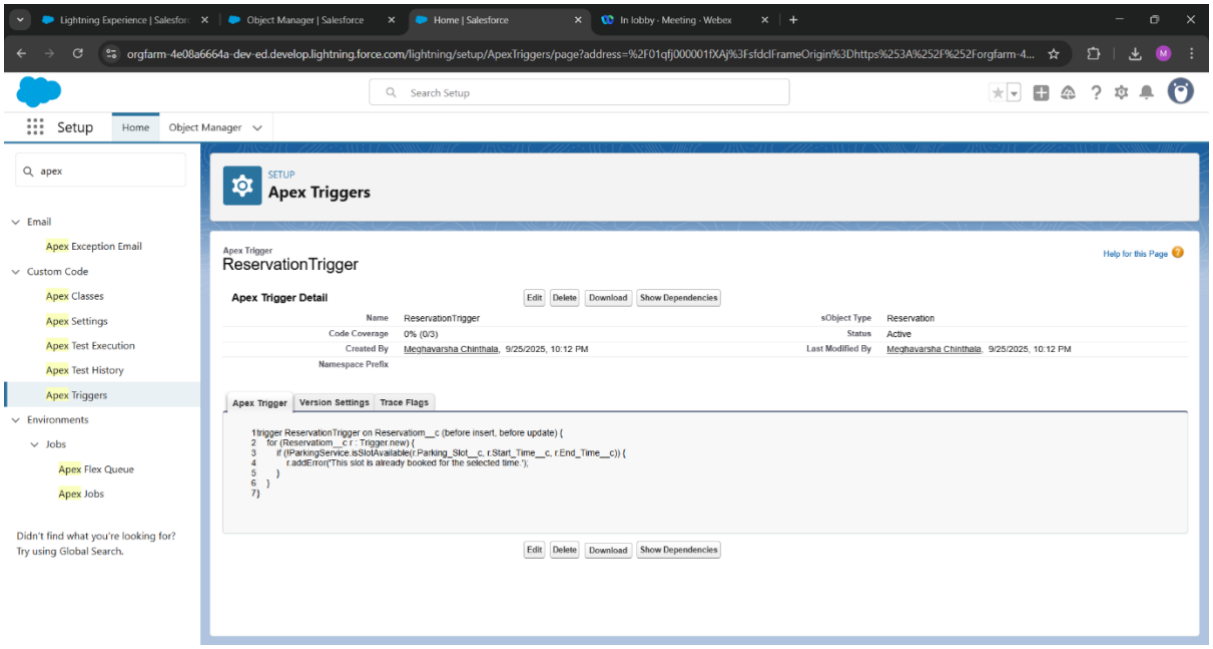
## 🔷 Step 2: Create Reservation Trigger

**Procedure:**

1. Go to **Object Manager → Reservation__c → Triggers → New**.
2. Name it ReservationTrigger.
3. Add the following code:

```
trigger ReservationTrigger on Reservation__c (before insert, before update) {
  if(Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
    ReservationHandler.validateReservation(Trigger.new);
  }
```

}




**Screenshot of Trigger Creation Screen**

---

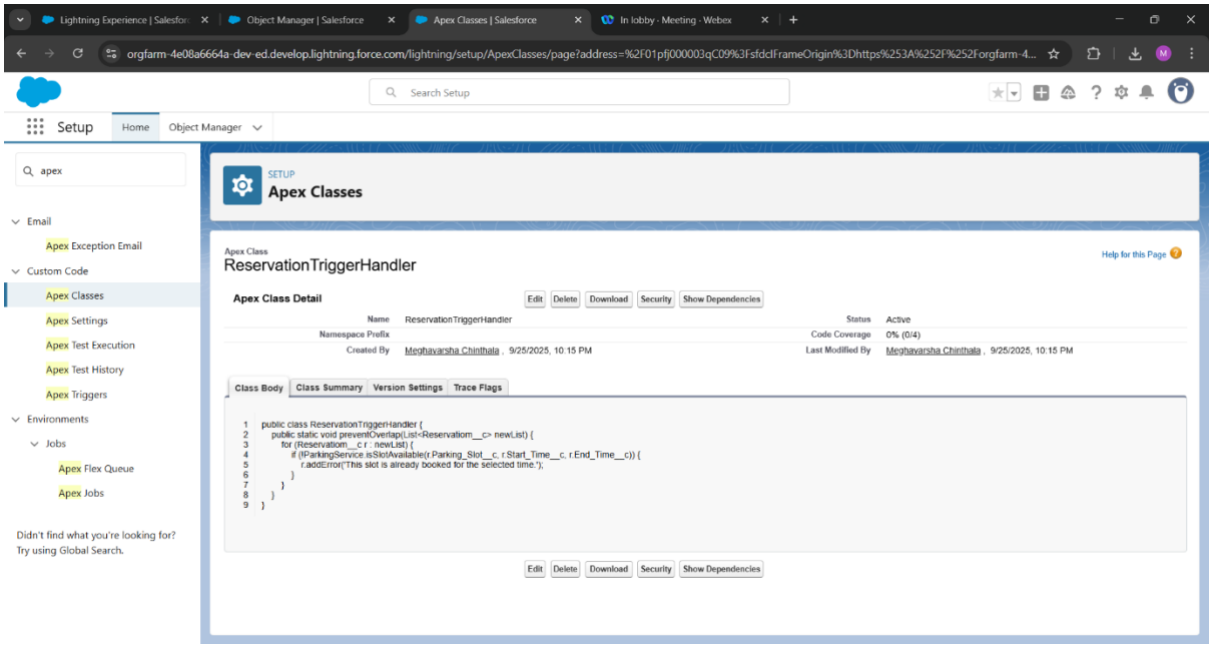
## 🔷 Step 3: Create ReservationHandler Class

**Procedure:**

1. Go to **Apex Classes → New**.
2. Name it ReservationHandler.
3. Paste the following code:

```
public class ReservationHandler {


   public static void validateReservation(List<Reservation__c> newReservations)
{
      Set<Id> slotIds = new Set<Id>();
      for(Reservation__c r : newReservations){
         if(r.Slot__c != null){
            slotIds.add(r.Slot__c);
         }
```

```
        }

        // Get existing reservations for same slots
        List<Reservation__c> existing = [SELECT Id, Slot__c, Start_Time__c,
End_Time__c
                                FROM Reservation__c
                                WHERE Slot__c IN :slotIds];

        // Validation check
        for(Reservation__c r : newReservations){
            for(Reservation__c e : existing){
                if(r.Slot__c == e.Slot__c &&
                  r.Start_Time__c < e.End_Time__c &&
                  r.End_Time__c > e.Start_Time__c){
                      r.addError('This slot is already reserved during the selected time.');
                }
            }
        }
    }
}
```



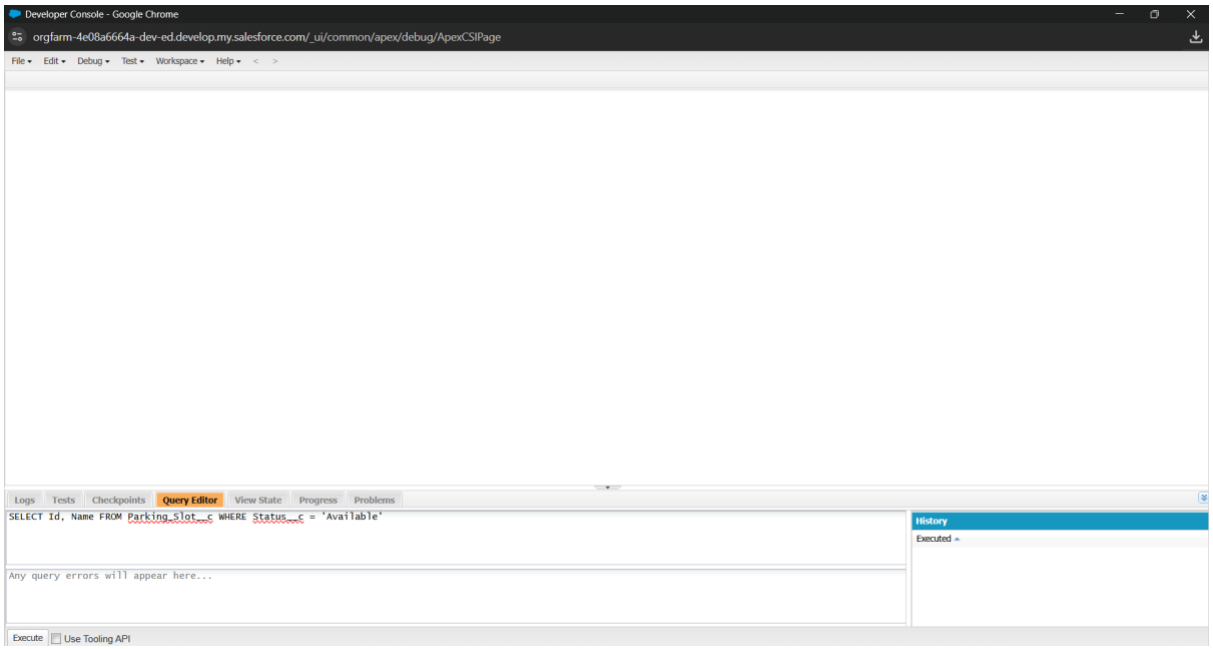

**Screenshot of Handler Class Code**

---

## 🔷 4. SOQL & SOSL Queries

**Procedure:**

- Use SOQL to query available slots.
- Use SOSL if you want to search text across objects.

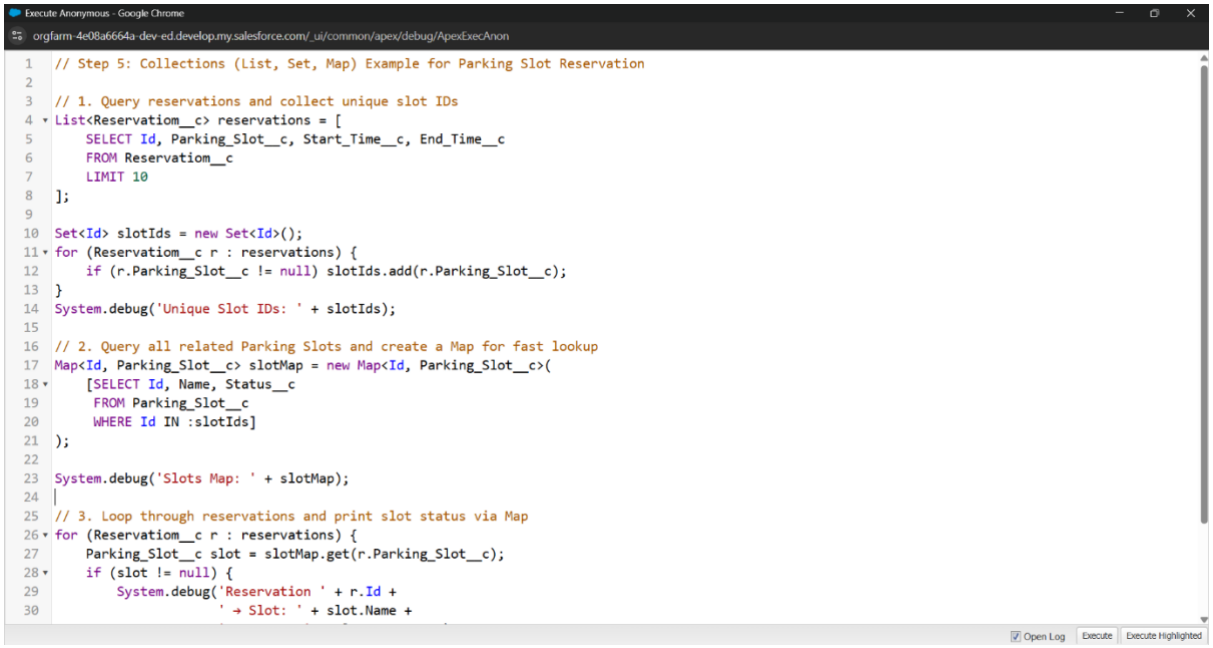



**Screenshot of Developer Console running SOQL query**

## 🔷 5. Collections (Set, Map, List)

**Procedure:**

- Use List for ordered records.
- Use Set to avoid duplicates.
- Use Map for key-value mapping.




**Screenshot of Debug Logs showing collection values**

---

## 🔷 6. Control Statements

**Procedure:**

- Use if conditions to check reservation conflicts.
- Throw error if overlap occurs.

**Code Example:**

```
if(newRes.Start_Time__c < existing.End_Time__c &&
  newRes.End_Time__c > existing.Start_Time__c){
   newRes.addError('This slot is already reserved during the selected time.');
}
```

### 🔷 7. Batch Apex

**Procedure:**

- Create a batch job that runs nightly.
- Marks expired reservations as **Completed**.

**Code Example:**

```
global class ExpiredReservationBatch implements
Database.Batchable<sObject> {

   global Database.QueryLocator start(Database.BatchableContext bc) {

      return Database.getQueryLocator(

         'SELECT Id, Status__c, End_Time__c FROM Reservation__c WHERE
End_Time__c < :System.now() AND Status__c = \'Confirmed\''

      );

   }

   global void execute(Database.BatchableContext bc, List<Reservation__c>
scope) {

      for(Reservation__c r : scope){

         r.Status__c = 'Completed';

      }

      update scope;

   }

   global void finish(Database.BatchableContext bc) {}

}
```

---

### 🔷 8. Queueable Apex

**Procedure:**

- Use for async processing like bulk notifications.

**Code Example:**

```
public class ReservationNotifier implements Queueable {
    public void execute(QueueableContext context) {
        List<Reservation__c> newReservations =
            [SELECT Id, Name FROM Reservation__c WHERE CreatedDate = TODAY];

        for(Reservation__c r : newReservations){
            System.debug('Notify employee for Reservation: ' + r.Name);
        }
    }
}

// Enqueue job
System.enqueueJob(new ReservationNotifier());
```

---

### 🔷 9. Scheduled Apex

**Procedure:**

- Send daily summary email to Manager.

**Code Example:**

```
global class DailyReservationSummary implements Schedulable {
    global void execute(SchedulableContext sc) {
        List<Reservation__c> todayReservations = [SELECT Id FROM
Reservation__c WHERE CreatedDate = TODAY];

        Messaging.SingleEmailMessage mail = new
Messaging.SingleEmailMessage();
        mail.setToAddresses(new String[] {'manager@company.com'});
        mail.setSubject('Daily Parking Reservations Summary');
```

```
        mail.setPlainTextBody('Total Reservations Today: ' +
todayReservations.size());
        Messaging.sendEmail(new Messaging.SingleEmailMessage[] {mail});
    }
}
```

---

### 🔷 10. Future Methods

**Procedure:**

- Use @future for async external API calls.

**Code Example:**

```
public class GateEntryIntegration {

    @future(callout=true)
    public static void sendEntryNotification(String reservationId) {
        System.debug('Calling external API for reservation: ' + reservationId);
        // Http callout code here
    }
}
```

---

### 🔷 11. Exception Handling

**Procedure:**

- Use try-catch blocks to handle errors gracefully.

**Code Example:**

```
try {
    insert reservation;
} catch (DmlException e) {
```

```
    System.debug('Error: ' + e.getMessage());
    reservation.addError('Reservation failed. Please try again.');
}
```

---

### 🔷 12. Test Classes

**Procedure:**

- Insert test data.
- Validate trigger logic & batch execution.

**Code Example (Simple):**

```
@isTest
public class ReservationTest {
    @isTest
    static void testValidReservation() {
        Parking_Slot__c slot = new Parking_Slot__c(Name='Test Slot',
Status__c='Available');
        insert slot;

        Reservation__c r = new Reservation__c(
            Slot__c = slot.Id,
            Start_Time__c = System.now(),
            End_Time__c = System.now().addHours(2)
        );
        insert r;

        System.assertNotEquals(null, r.Id);
    }
```

}

---

### 🔷 13. Asynchronous Processing

**Procedure:**

- Combine **Batch Apex, Queueable, and Future Methods** for smooth operations.
- Use:
  - **Batch** → clean expired reservations.
  - **Queueable** → bulk notifications.