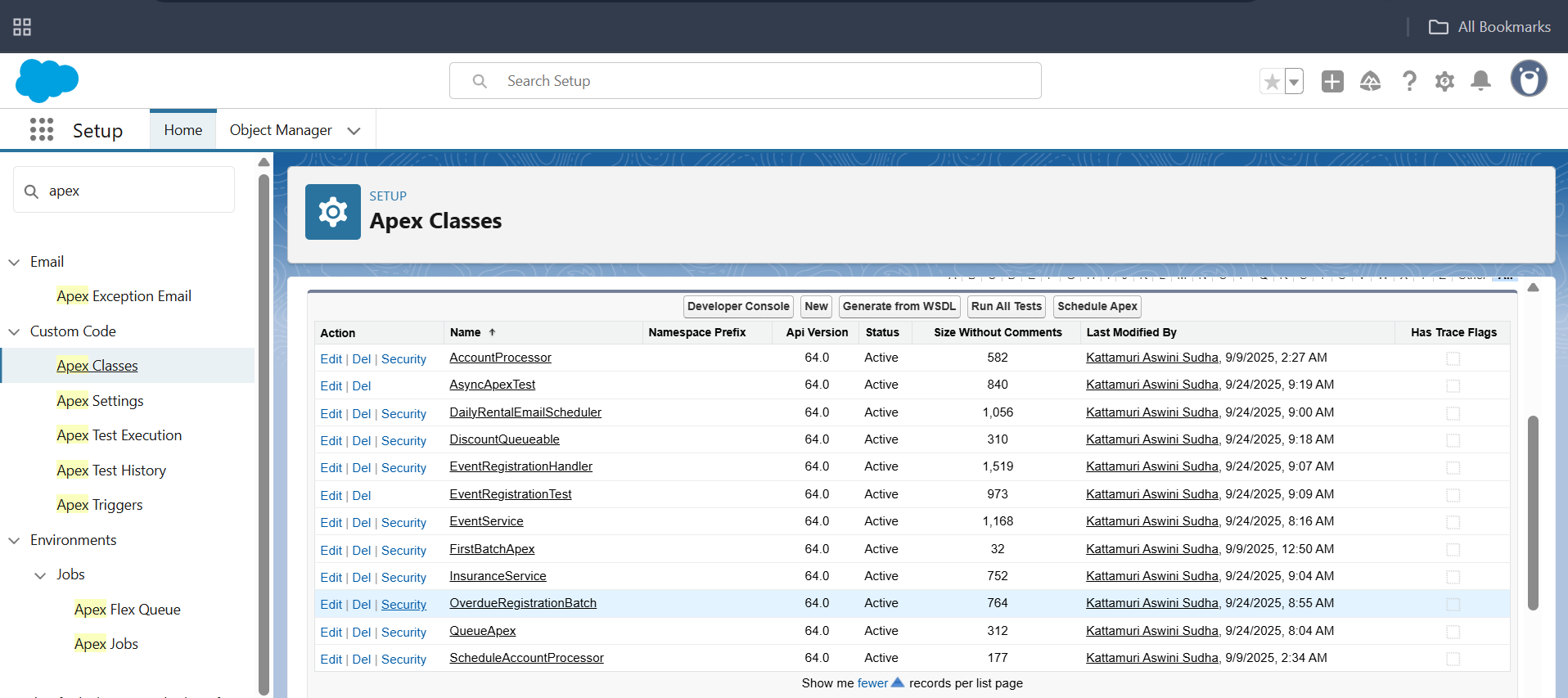
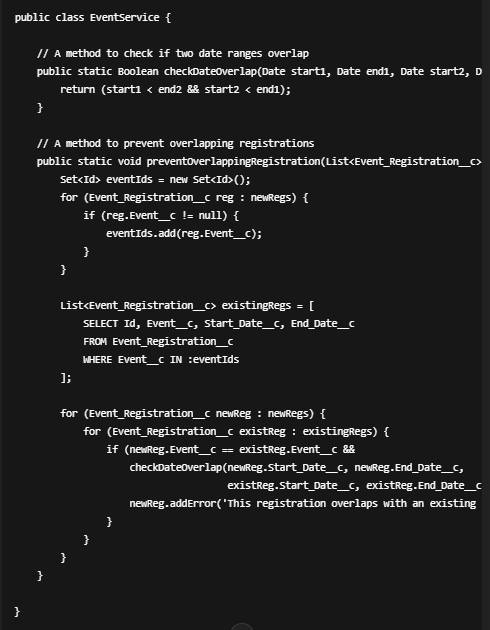
**Phase 5: Apex Programming (Developer)**

 Goal: Add advanced logic.

1. **Classes & Objects**

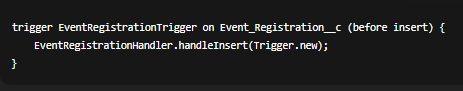
Create a EventService class for reusable logic.





**2.Apex Triggers**

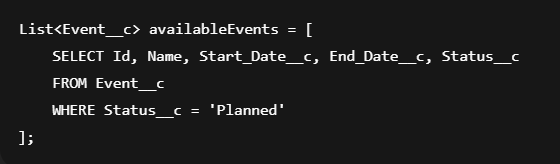
Created Apex Trigger with name **EventRegistrationTrigger** and sObject  
**Event\_Registration\_\_c**

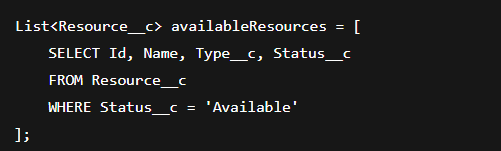
  
  
**3: Create Test Class**In **Developer Console → File → New → Apex Class**

**Name:** EventRegistrationTest  

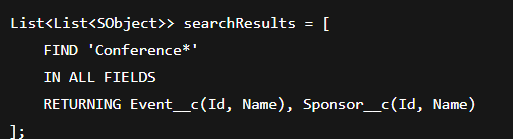

**Step 4: SOQL & SOSL in Event Management**

Open **Developer Console → Query Editor** or use **Apex Class / Execute Anonymous**.

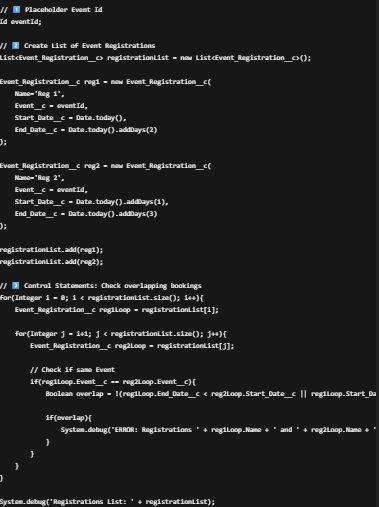
**B. Query Available Resources (SOQL)**



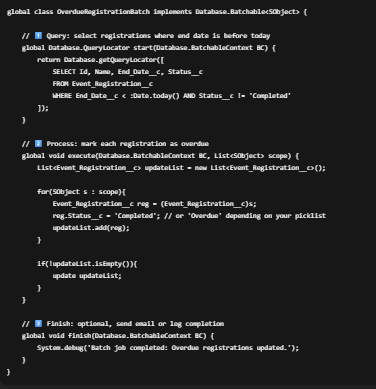
1.Open **Developer Console → Execute Anonymous**.



**Step 5:Collections: List, Set, Map**

  
  
**Step 6: Batch Apex**

1. In **Developer Console → File → New → Apex Class**
   * **Name:** EventBatch

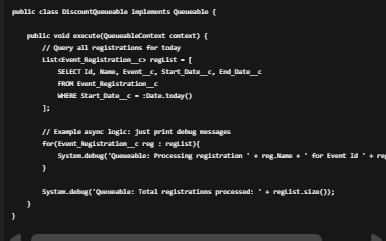


**Open Execute Anonymous Window and run**

EventBatch b = new EventBatch();

Database.executeBatch(b, 50);

**Step 7: Queueable Apex**



**Step 8: Scheduled Apex**



Open **Execute Anonymous** in Developer Console and paste:

// Schedule to run at 8 AM every day

String cronExp = '0 0 8 \* \* ?';

System.schedule('Daily Event Registrations Email', cronExp, new DailyRentalEmailScheduler());

**Step 9:Future Methods – Call External API Async**

// Get real registrations

List<Event\_Registration\_\_c> regs = [SELECT Id FROM Event\_Registration\_\_c LIMIT 5];

// Convert to CSV string

List<Id> regIds = new List<Id>();

for(Event\_Registration\_\_c r : regs){

regIds.add(r.Id);

}

String regIdsCsv = String.join(regIds, ',');

// Call Future Method

InsuranceService.callInsuranceApi(regIdsCsv);

**Run it asynchronously:**

Set<Id> regIds = new Set<Id>{'a0BXXXXXXXXXXXX', 'a0BXXXXXXXXXXXX'};

InsuranceService.callInsuranceApi(regIds);

**Step 10:Exception Handling**

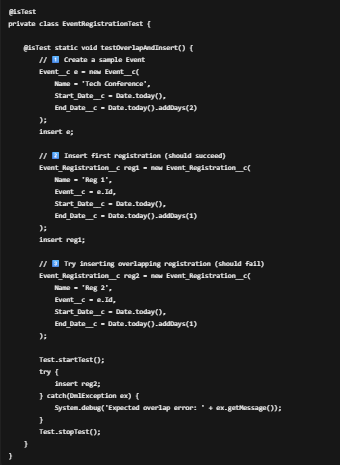
****

trigger EventRegistrationTrigger on Event\_Registration\_\_c (before insert) {

EventRegistrationHandler.checkOverlaps(Trigger.new);

}

**Step 11: Test Class for Event Registration Trigger**

****

**Step 12: Asynchronous Processing**

