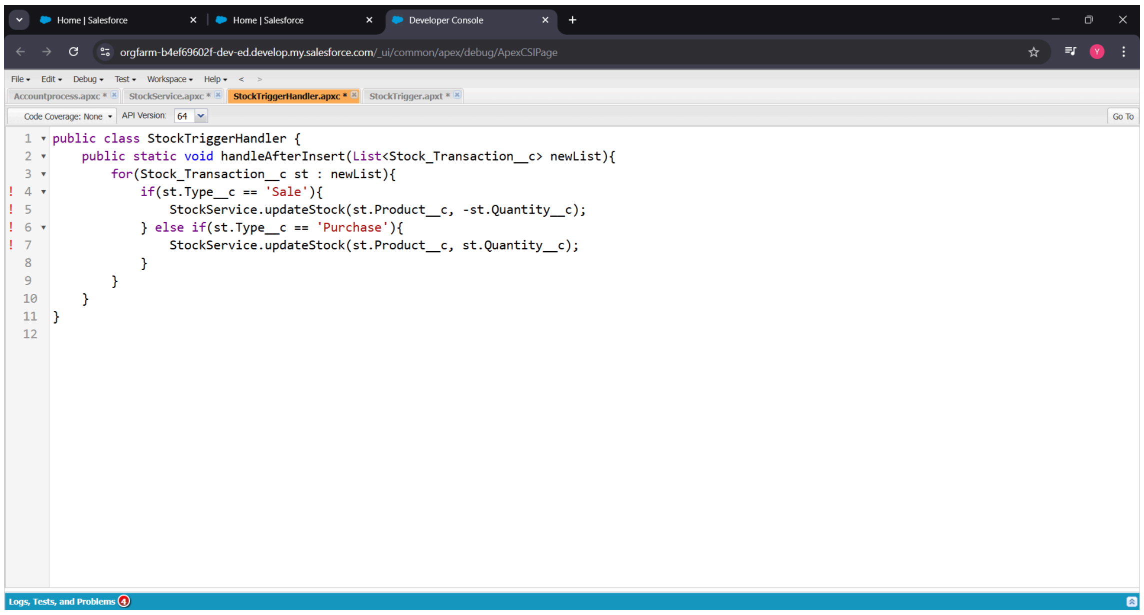
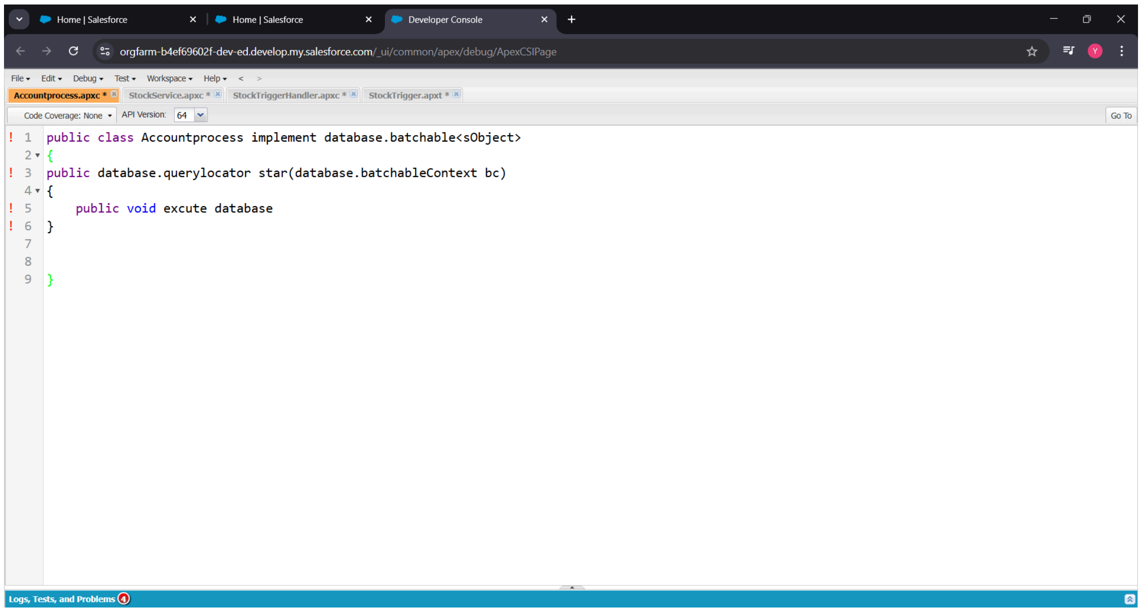
Phase 5: StockSense:-Apex Programming (Developer)

Create Apex Trigger Handler – StockTriggerHandler • Classes & Objects


Home | Salesforce
Home | Salesforce
Developer Console
orgfarm-b4ef69602f-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
Accountprocess.apxc * StockService.apxc * StockTriggerHandler.apxc * StockTrigger.apxt *
Code Coverage: None API Version: 64
Go To
Logs, Tests, and Problems 4

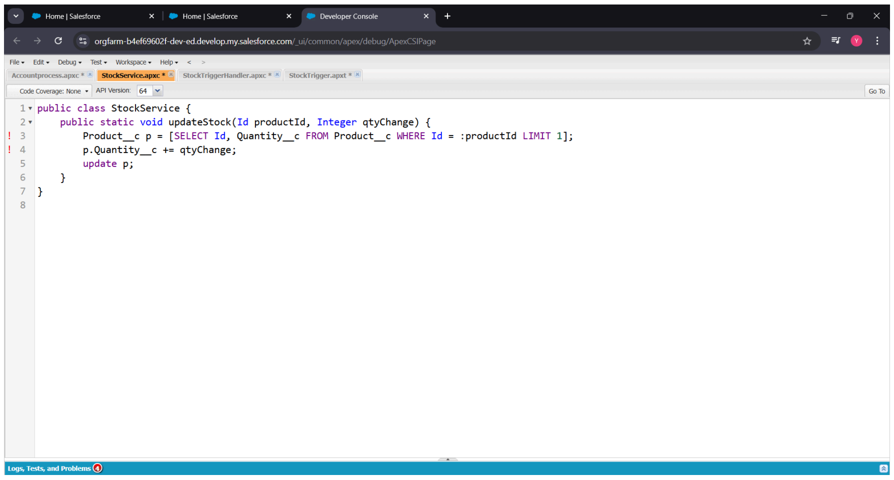
```
public class StockTriggerHandler {
    public static void handleAfterInsert(List<Stock_Transaction__c> newList){
        for(Stock_Transaction__c st : newList){
            if(st.Type__c == 'Sale'){
                StockService.updateStock(st.Product__c, -st.Quantity__c);
            } else if(st.Type__c == 'Purchase'){
                StockService.updateStock(st.Product__c, st.Quantity__c);
            }
        }
    }
}
```

Create • Asynchronous Processing– Accountprocess


Home | Salesforce
Home | Salesforce
Developer Console
orgfarm-b4ef69602f-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
Accountprocess.apxc * StockService.apxc * StockTriggerHandler.apxc * StockTrigger.apxt *
Code Coverage: None API Version: 64
Go To
Logs, Tests, and Problems 4

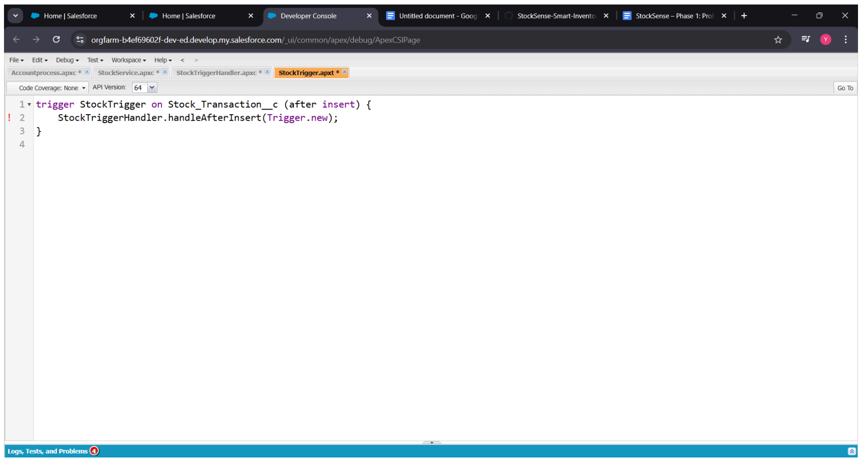
```
public class Accountprocess implement database.batchable<sObject>
{
public database.querylocator star(database.batchableContext bc)
{
    public void excute database
}


}
```

Create Apex class – Stockservice



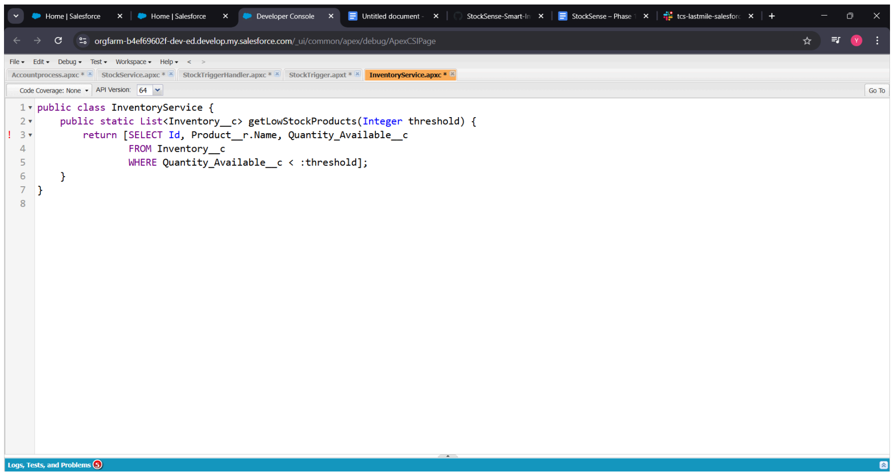
```
public class StockService {
    public static void updateStock(Id productId, Integer qtyChange) {
        Product__c p = [SELECT Id, Quantity__c FROM Product__c WHERE Id = :productId LIMIT 1];
        p.Quantity__c += qtyChange;
        update p;
    }
}
```

Create Apex Trigger - stocktrigger • Trigger Design Pattern



```
trigger StockTrigger on Stock_Transaction__c (after insert) {
    StockTriggerHandler.handleAfterInsert(Trigger.new);
}
```

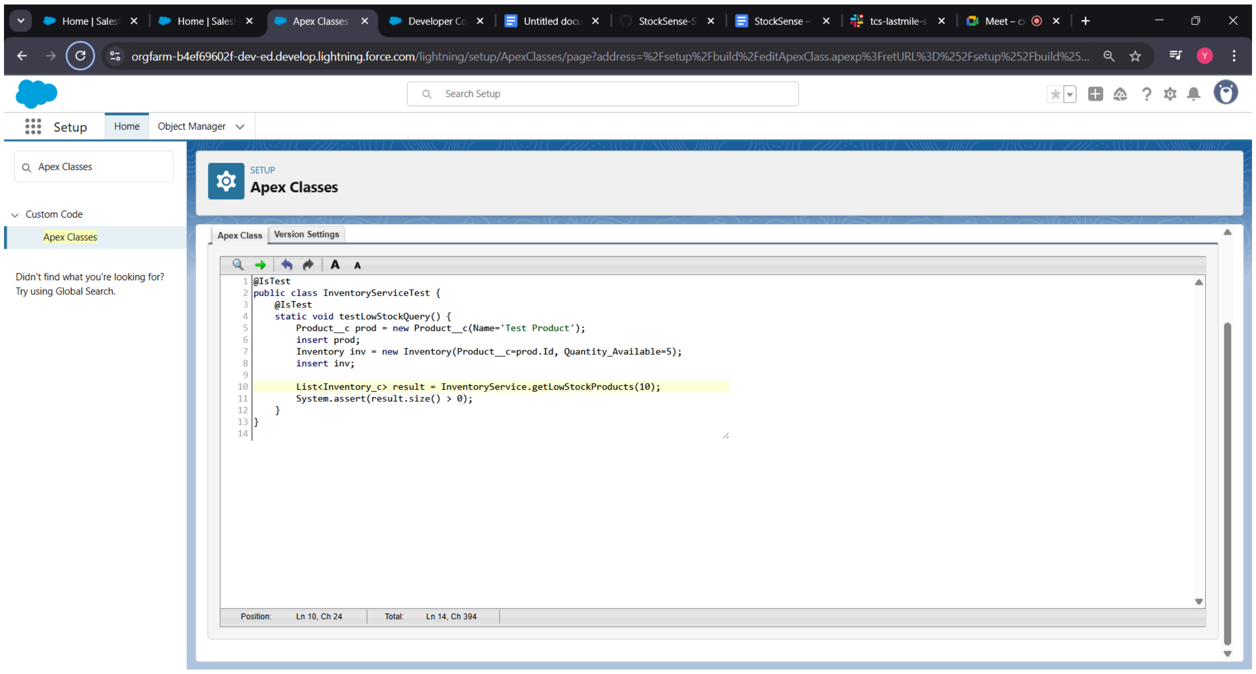
Apex Class for Reusable Queries–



```
public class InventoryService {
    public static List<Inventory__c> getLowStockProducts(Integer threshold) {
        return [SELECT Id, Product__r.Name, Quantity_Available__c
                FROM Inventory__c
                WHERE Quantity_Available__c < :threshold];
    }
}
```

Test Your Apex Class–
Click **Setup → Apex Classes → New → Test Class**.

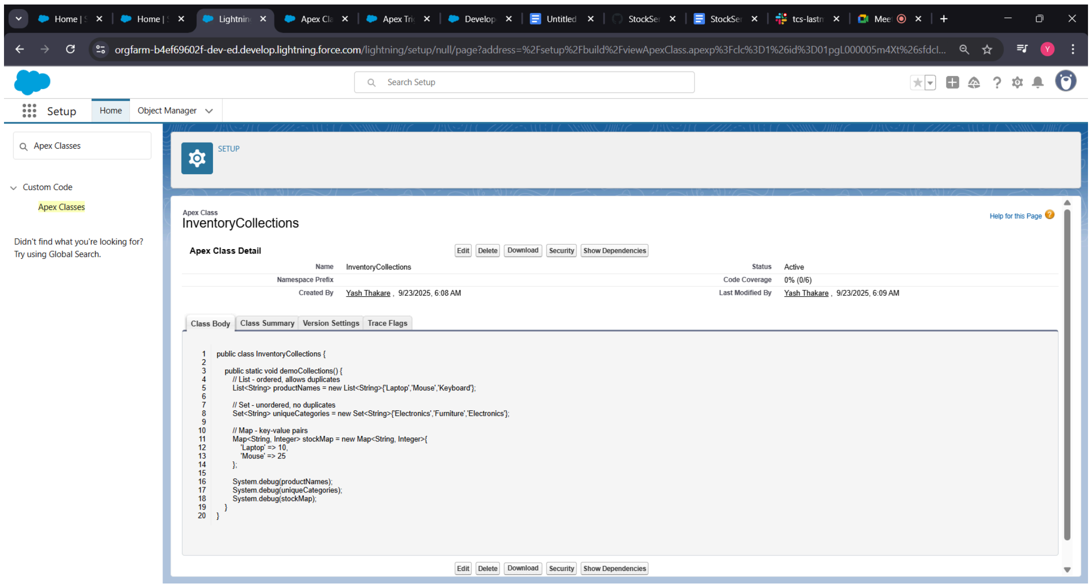
Write something like:



```
@IsTest
public class InventoryServiceTest {
    @IsTest
    static void testLowStockQuery() {
        Product__c prod = new Product__c(Name='Test Product');
        insert prod;
        Inventory inv = new Inventory(Product__c=prod.Id, Quantity_Available=5);
        insert inv;

        List<Inventory__c> result = InventoryService.getLowStockProducts(10);
        System.assert(result.size() > 0);
    }
}
```

Collections: List, Set, Map:-**Go to Setup → Apex Classes → New.**

Create a class for inventory operations:



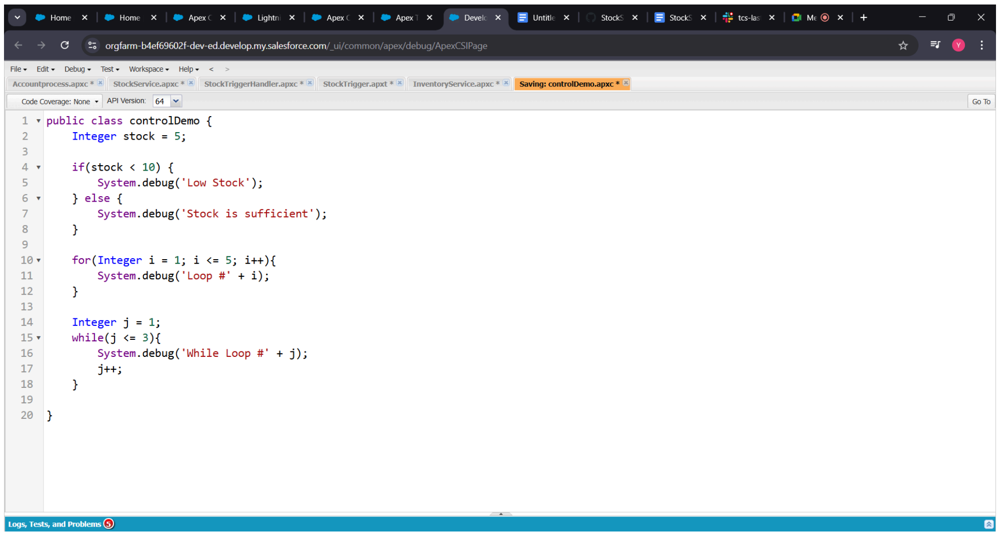
Apex Class
InventoryCollections

Apex Class Detail

Name	InventoryCollections	Status	Active
Namespace Prefix		Code Coverage	0% (0/6)
Created By	Yash Thakare , 9/23/2025, 6:08 AM	Last Modified By	Yash Thakare , 9/23/2025, 6:09 AM

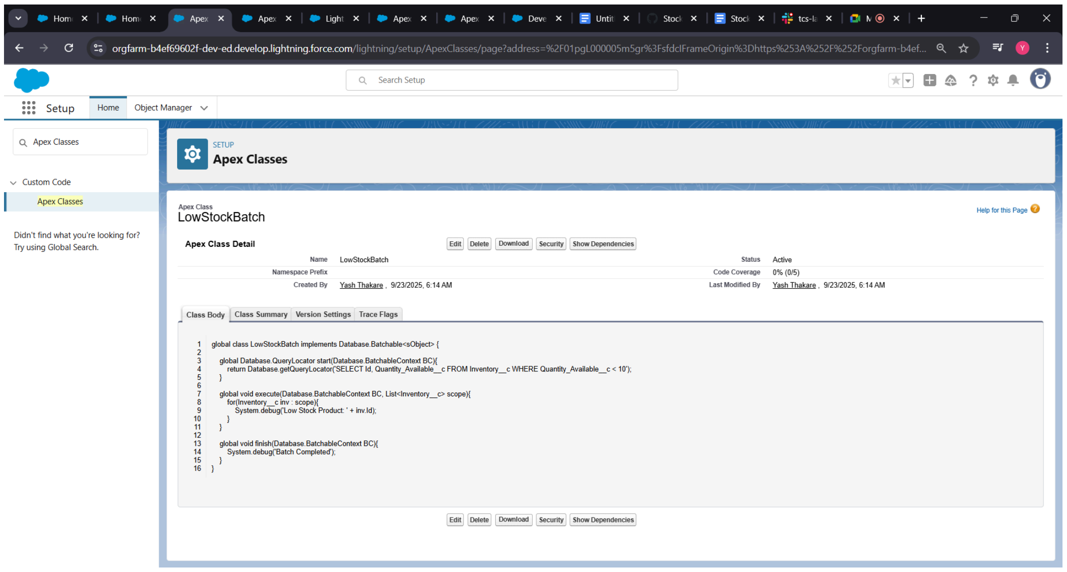
```
public class InventoryCollections {

    public static void demoCollections() {
        // List - ordered, allows duplicates
        List<String> productNames = new List<String>{'Laptop','Mouse','Keyboard'};

        // Set - unordered, no duplicates
        Set<String> uniqueCategories = new Set<String>{'Electronics','Furniture','Electronics'};

        // Map - key-value pairs
        Map<String, Integer> stockMap = new Map<String, Integer>{
            'Laptop' => 10,
            'Mouse' => 25
        };

        System.debug(productNames);
        System.debug(uniqueCategories);
        System.debug(stockMap);
    }
}
```

Control Statements:-



```
public class controlDemo {
    Integer stock = 5;

    if(stock < 10) {
        System.debug('Low Stock');
    } else {
        System.debug('Stock is sufficient');
    }

    for(Integer i = 1; i <= 5; i++){
        System.debug('Loop #' + i);
    }

    Integer j = 1;
    while(j <= 3){
        System.debug('While Loop #' + j);
        j++;
    }

}
```

Batch Apex:-



Apex Class
LowStockBatch

Apex Class Detail

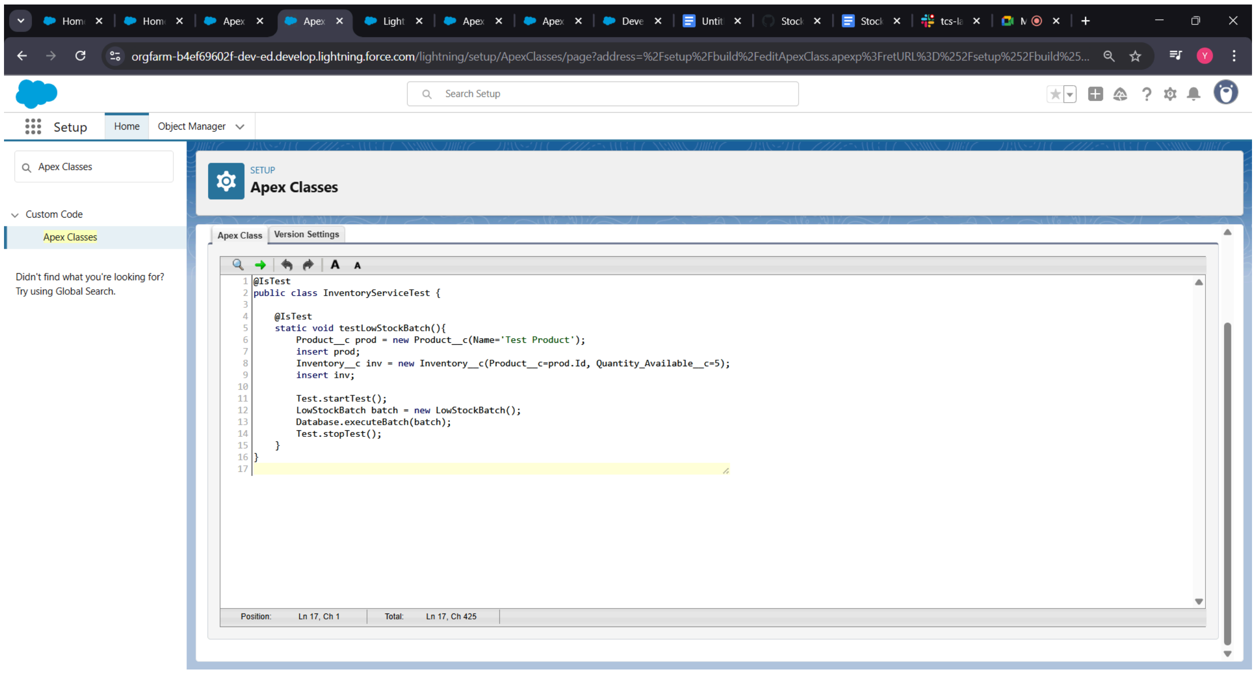
Name	LowStockBatch	Status	Active
Namespace Prefix		Code Coverage	0% (0/5)
Created By	Yash Thakare , 9/23/2025, 6:14 AM	Last Modified By	Yash Thakare , 9/23/2025, 6:14 AM

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

    global Database.QueryLocator start(Database.BatchableContext BC){
        return Database.getQueryLocator('SELECT Id, Quantity_Available__c FROM Inventory__c WHERE Quantity_Available__c < 10');
    }

    global void execute(Database.BatchableContext BC, List<Inventory__c> scope){
        for(Inventory__c inv : scope){
            System.debug('Low Stock Product: ' + inv.Id);
        }
    }

    global void finish(Database.BatchableContext BC){
        System.debug('Batch Completed');
    }
}
```

Asynchronous Processing Overview

- **Batch Apex:** Large volume jobs.
- **Queueable Apex:** Complex jobs in async queue.
- **Scheduled Apex:** Run jobs periodically.
- **Future Methods:** Lightweight async operations.

Test Classes:-Apex class


orgfarm-b4ef69602f-dev-ed.develop.lightning.force.com/lightning/setup/ApexClasses/page?address=%2Fsetup%2Fbuild%2FeditApexClass.apexp%3FretURL%3D%252Fsetup%252Fbuild%25...
Search Setup
Setup
Home
Object Manager
Apex Classes
Custom Code
Apex Classes
Didn't find what you're looking for? Try using Global Search.
SETUP
Apex Classes
Apex Class
Version Settings
Position: Ln 17, Ch 1
Total: Ln 17, Ch 425

```
@IsTest
public class InventoryServiceTest {

    @IsTest
    static void testLowStockBatch(){
        Product__c prod = new Product__c(Name='Test Product');
        insert prod;
        Inventory__c inv = new Inventory__c(Product__c=prod.Id, Quantity_Available__c=5);
        insert inv;

        Test.startTest();
        LowStockBatch batch = new LowStockBatch();
        Database.executeBatch(batch);
        Test.stopTest();
    }
}

```