Phase 5: Apex Programming (Developer)

- Classes & Objects
- Apex Triggers (before/after insert/update/delete)
- Trigger Design Pattern
- SOQL & SOSL
- Collections: List, Set, Map
- Control Statements
- Batch Apex
- Queueable Apex
- Scheduled Apex
- Future Methods
- Exception Handling
- Test Classes
- Asynchronous Processing

Classes & Objects:

Objects

- Objects are database tables in Salesforce that store data.
- **Standard Objects** (like Account, Contact) come built-in.

- **Custom Objects** (like Meal_Plan__c, Recipe__c, Ingredient__c) are created to fit business needs.
- Each object can have **fields** (e.g., Quantity__c, Calories__c, Meal_Plan_Lookup__c).

Example in our project:

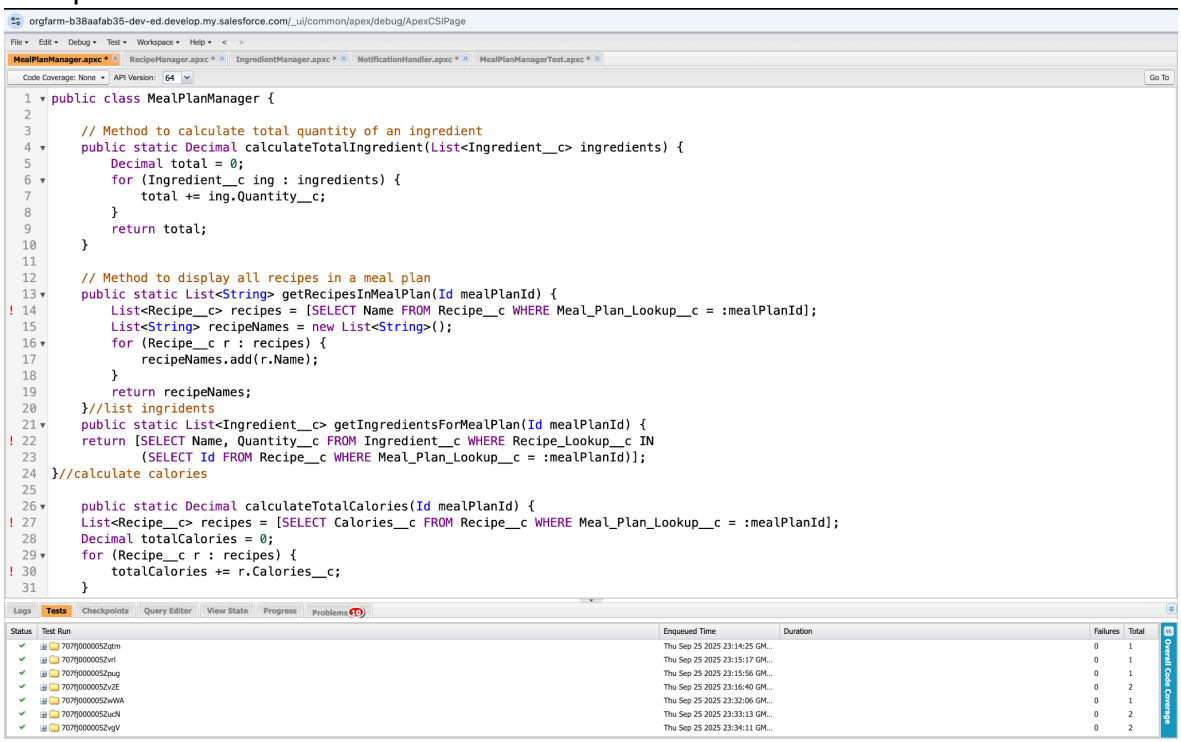
- Meal_Plan__c → Stores details of each meal plan.
- Recipe__c → Stores recipes linked to meal plans.
- Ingredient__c → Stores ingredients with quantity and calories.

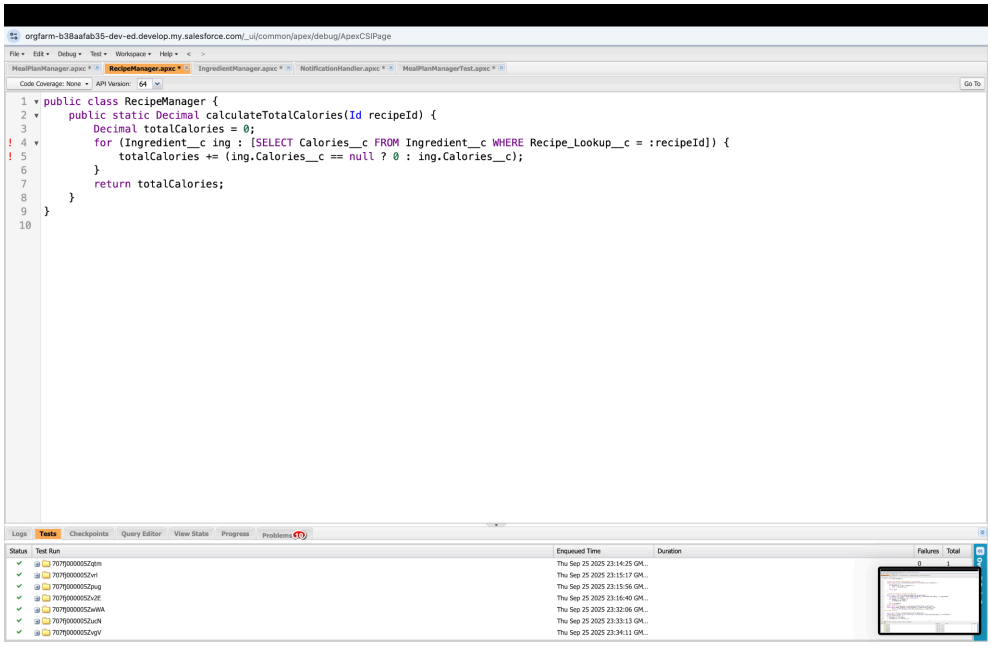
Classes:

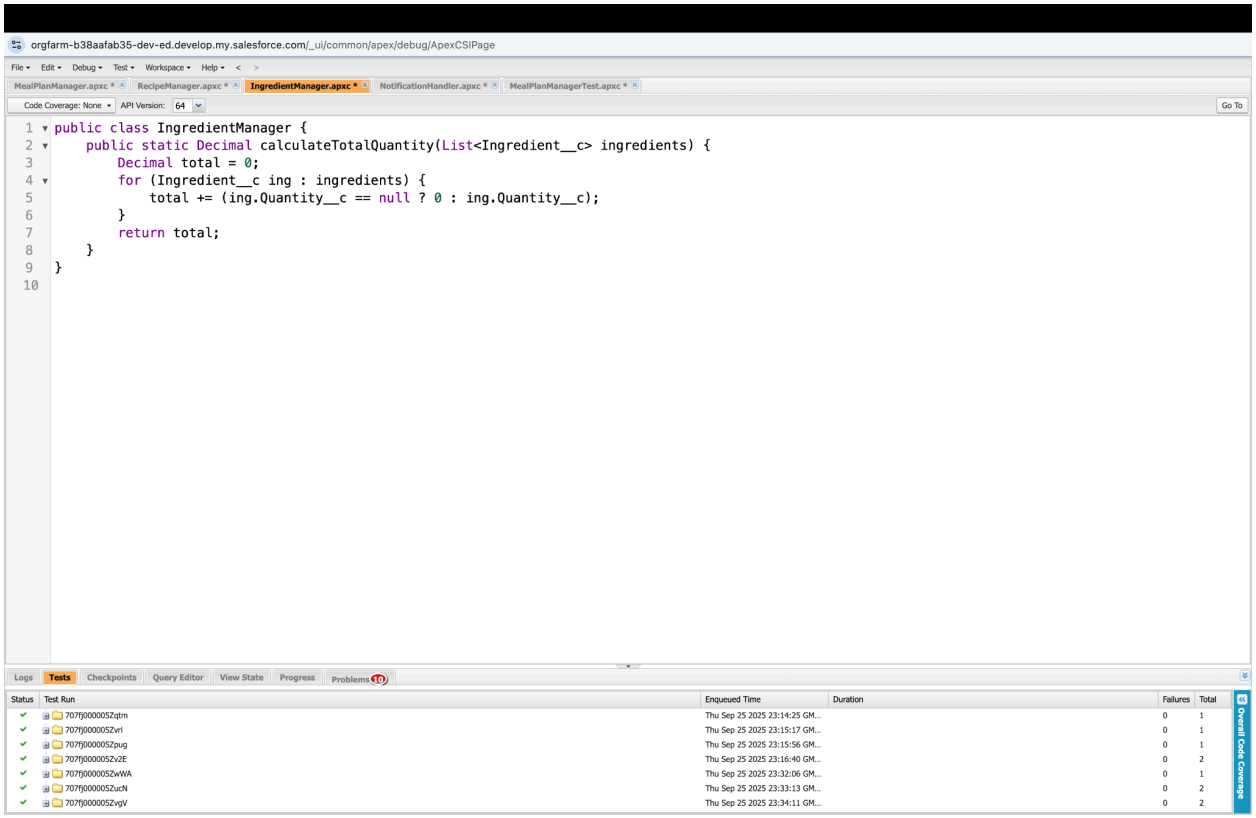
Classes in Salesforce are written in **Apex** (Salesforce's programming language).

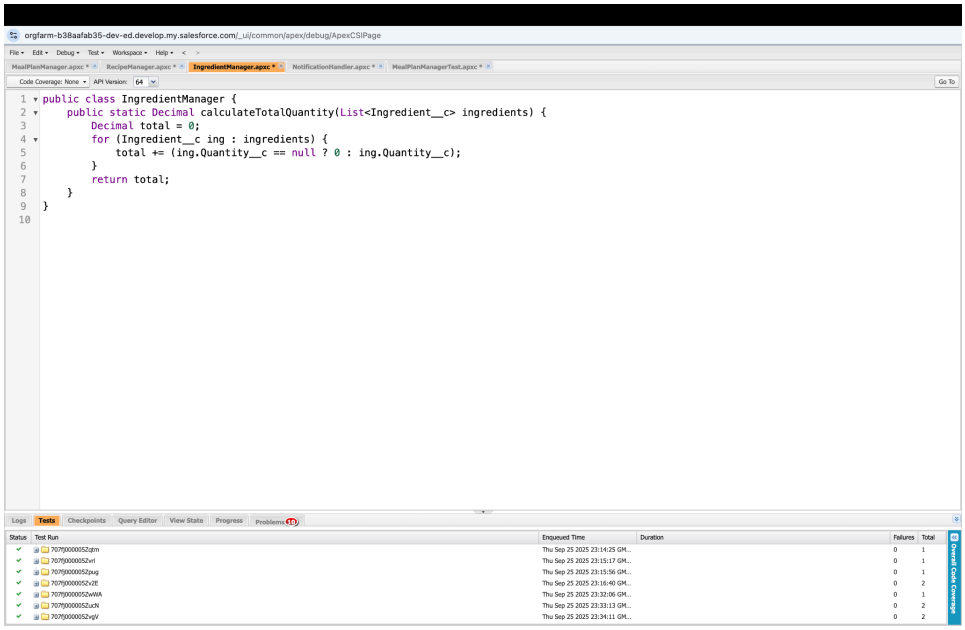
They define **behavior (logic)** that works with objects.

A class contains **methods** (functions) to perform operations like calculations, updates, and queries.

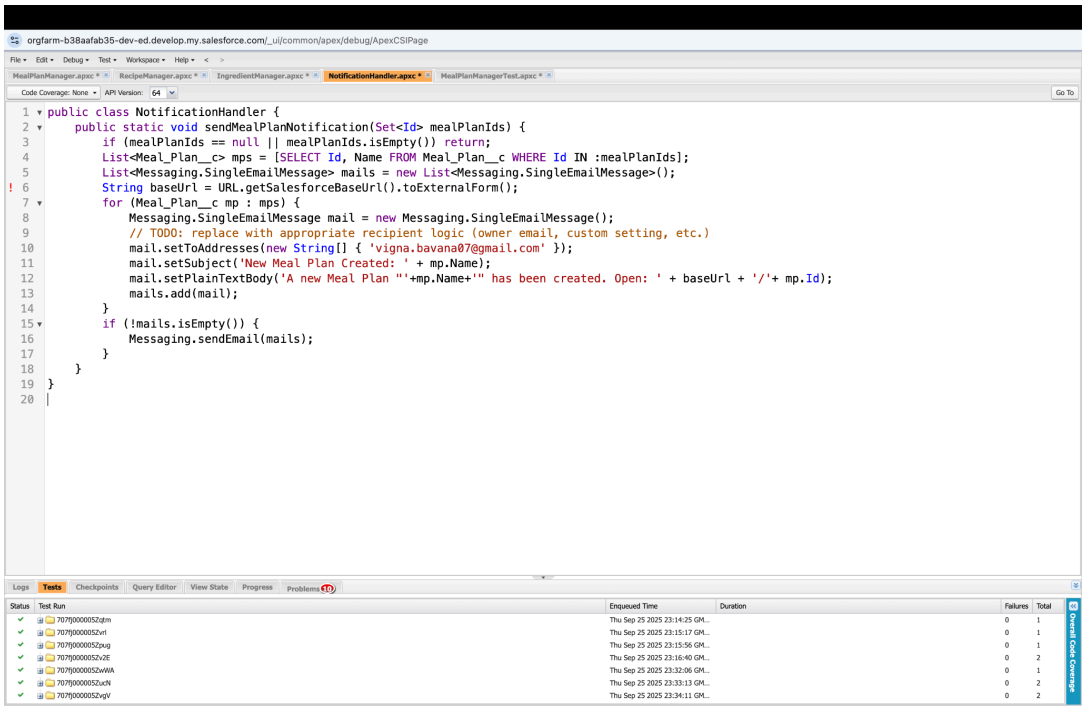


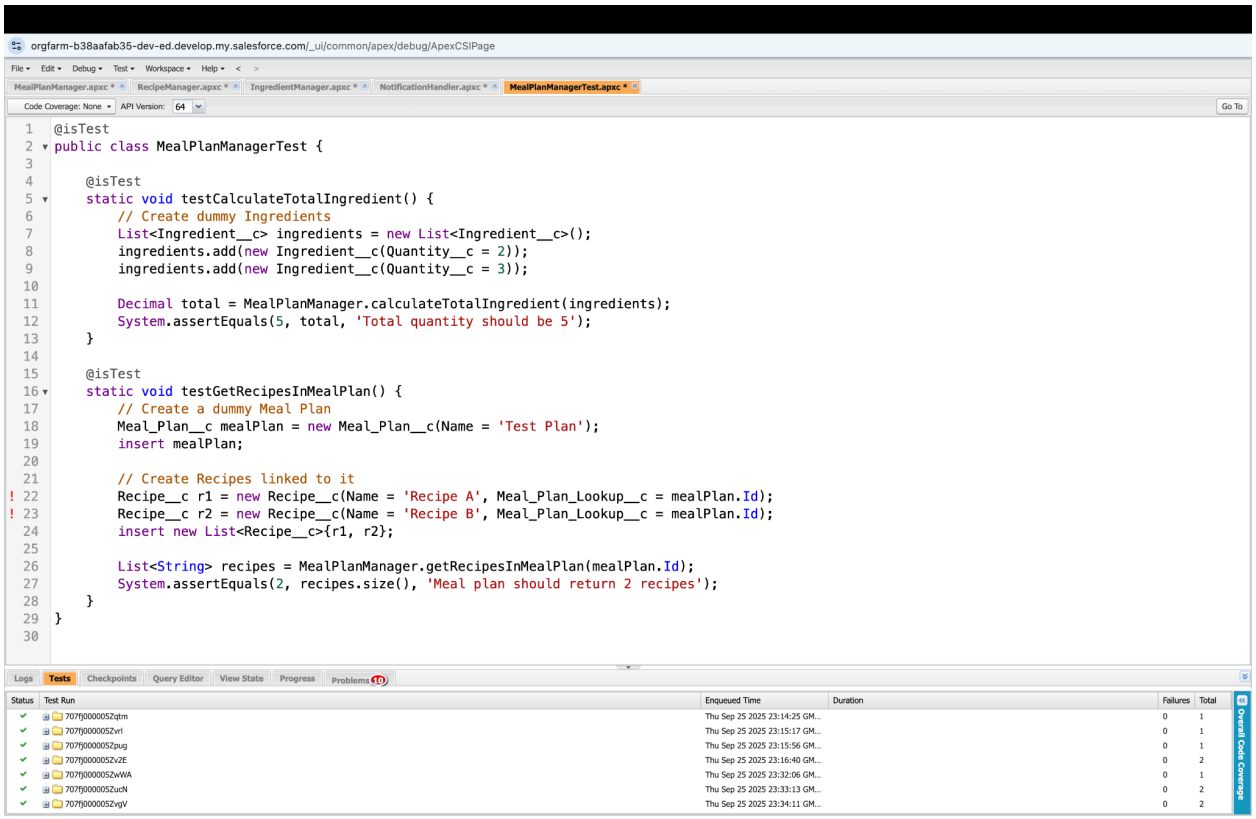

orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
public class RecipeManager {
public static Decimal calculateTotalCalories(Id recipeId) {
Decimal totalCalories = 0;
for (Ingredient__c ing : [SELECT Calories__c FROM Ingredient__c WHERE Recipe_Lookup__c = :recipeId]) {
totalCalories += (ing.Calories__c == null ? 0 : ing.Calories__c);
}
return totalCalories;
}
}
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems 10
Status
Test Run
Enqueued Time
Duration
Failures
Total
707fj000005Zqtm
707fj000005Zvrl
707fj000005Zpug
707fj000005Zv2E
707fj000005ZwWA
707fj000005ZucN
707fj000005ZvgV
Thu Sep 25 2025 23:14:25 GM...
Thu Sep 25 2025 23:15:17 GM...
Thu Sep 25 2025 23:15:56 GM...
Thu Sep 25 2025 23:16:40 GM...
Thu Sep 25 2025 23:32:06 GM...
Thu Sep 25 2025 23:33:13 GM...
Thu Sep 25 2025 23:34:11 GM...
Overall Code Coverage


orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
public class IngredientManager {
public static Decimal calculateTotalQuantity(List<Ingredient__c> ingredients) {
Decimal total = 0;
for (Ingredient__c ing : ingredients) {
total += (ing.Quantity__c == null ? 0 : ing.Quantity__c);
}
return total;
}
}
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems 10
Status
Test Run
Enqueued Time
Duration
Failures
Total
707fj000005Zqtm
707fj000005Zvrl
707fj000005Zpug
707fj000005Zv2E
707fj000005ZwWA
707fj000005ZucN
707fj000005ZvgV
Thu Sep 25 2025 23:14:25 GM...
Thu Sep 25 2025 23:15:17 GM...
Thu Sep 25 2025 23:15:56 GM...
Thu Sep 25 2025 23:16:40 GM...
Thu Sep 25 2025 23:32:06 GM...
Thu Sep 25 2025 23:33:13 GM...
Thu Sep 25 2025 23:34:11 GM...
Overall Code Coverage



orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
MealPlanManager.apxc * RecipeManager.apxc * IngredientManager.apxc * NotificationHandler.apxc * MealPlanManagerTest.apxc *
Code Coverage: None API Version: 64 Go To
1 public class IngredientManager {
2 public static Decimal calculateTotalQuantity(List<Ingredient__c> ingredients) {
3 Decimal total = 0;
4 for (Ingredient__c ing : ingredients) {
5 total += (ing.Quantity__c == null ? 0 : ing.Quantity__c);
6 }
7 return total;
8 }
9 }
10
Logs Tests Checkpoints Query Editor View State Progress Problems 10
Status Test Run Enqueued Time Duration Failures Total
707fj000005Zqtm Thu Sep 25 2025 23:14:25 GM... 0 1
707fj000005Zvrl Thu Sep 25 2025 23:15:17 GM... 0 1
707fj000005Zpug Thu Sep 25 2025 23:15:56 GM... 0 1
707fj000005Zv2E Thu Sep 25 2025 23:16:40 GM... 0 2
707fj000005ZwWA Thu Sep 25 2025 23:32:06 GM... 0 1
707fj000005ZucN Thu Sep 25 2025 23:33:13 GM... 0 2
707fj000005ZvgV Thu Sep 25 2025 23:34:11 GM... 0 2
Overall Code Coverage

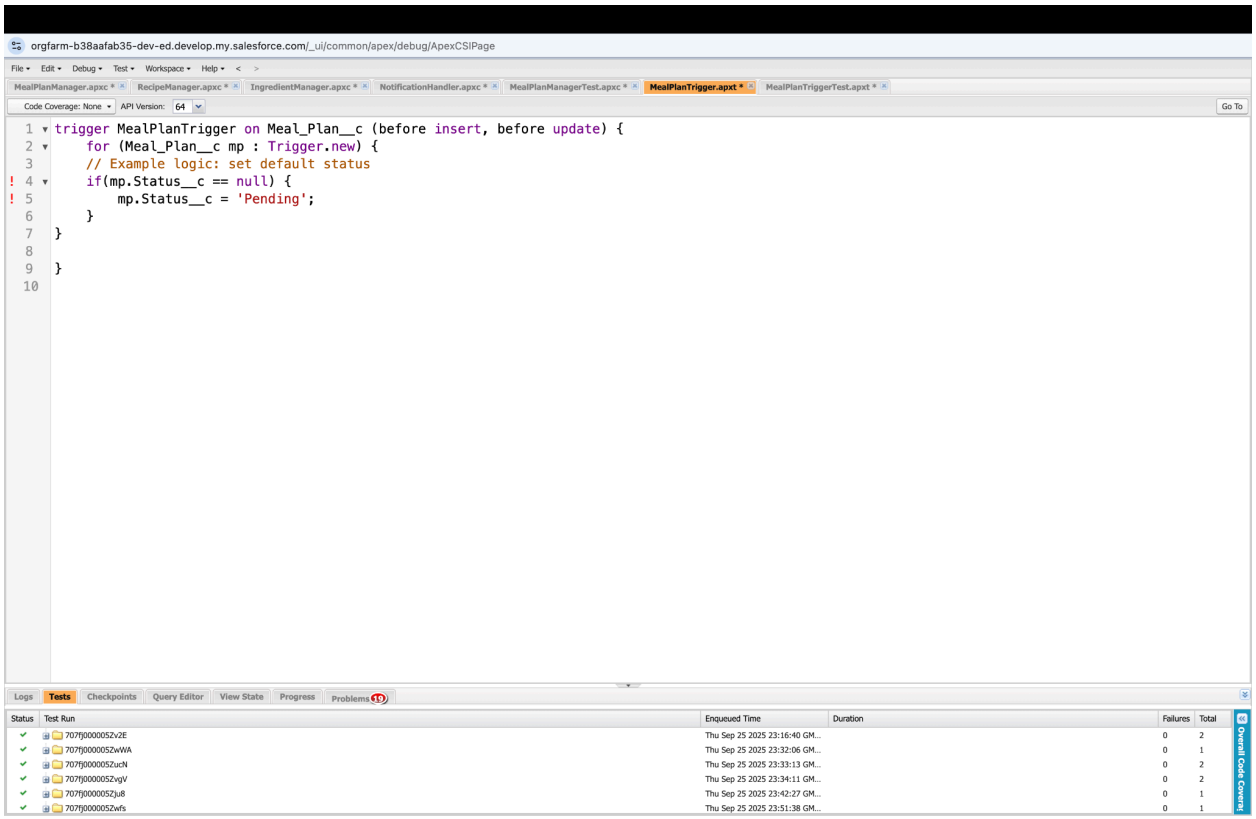

orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
MealPlanManager.apxc *
RecipeManager.apxc *
IngredientManager.apxc *
NotificationHandler.apxc *
MealPlanManagerTest.apxc *
Code Coverage: None
API Version: 64
Go To
1 public class NotificationHandler {
2 public static void sendMealPlanNotification(Set<Id> mealPlanIds) {
3 if (mealPlanIds == null || mealPlanIds.isEmpty()) return;
4 List<Meal_Plan__c> mps = [SELECT Id, Name FROM Meal_Plan__c WHERE Id IN :mealPlanIds];
5 List<Messaging.SingleEmailMessage> mails = new List<Messaging.SingleEmailMessage>();
6 String baseUrl = URL.getSalesforceBaseUrl().toExternalForm();
7 for (Meal_Plan__c mp : mps) {
8 Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
9 // TODO: replace with appropriate recipient logic (owner email, custom setting, etc.)
10 mail.setToAddresses(new String[] { 'vigna.bavana07@gmail.com' });
11 mail.setSubject('New Meal Plan Created: ' + mp.Name);
12 mail.setPlainTextBody('A new Meal Plan "'+mp.Name+'" has been created. Open: ' + baseUrl + '/'+ mp.Id);
13 mails.add(mail);
14 }
15 if (!mails.isEmpty()) {
16 Messaging.sendEmail(mails);
17 }
18 }
19 }
20
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems 10
Status
Test Run
Enqueued Time
Duration
Failures
Total
707fj000005Zqtm Thu Sep 25 2025 23:14:25 GM... 0 1
707fj000005Zvrl Thu Sep 25 2025 23:15:17 GM... 0 1
707fj000005Zpug Thu Sep 25 2025 23:15:56 GM... 0 1
707fj000005Zv2E Thu Sep 25 2025 23:16:40 GM... 0 2
707fj000005ZwWA Thu Sep 25 2025 23:32:06 GM... 0 1
707fj000005ZucN Thu Sep 25 2025 23:33:13 GM... 0 2
707fj000005ZvgV Thu Sep 25 2025 23:34:11 GM... 0 2
Overall Code Coverage

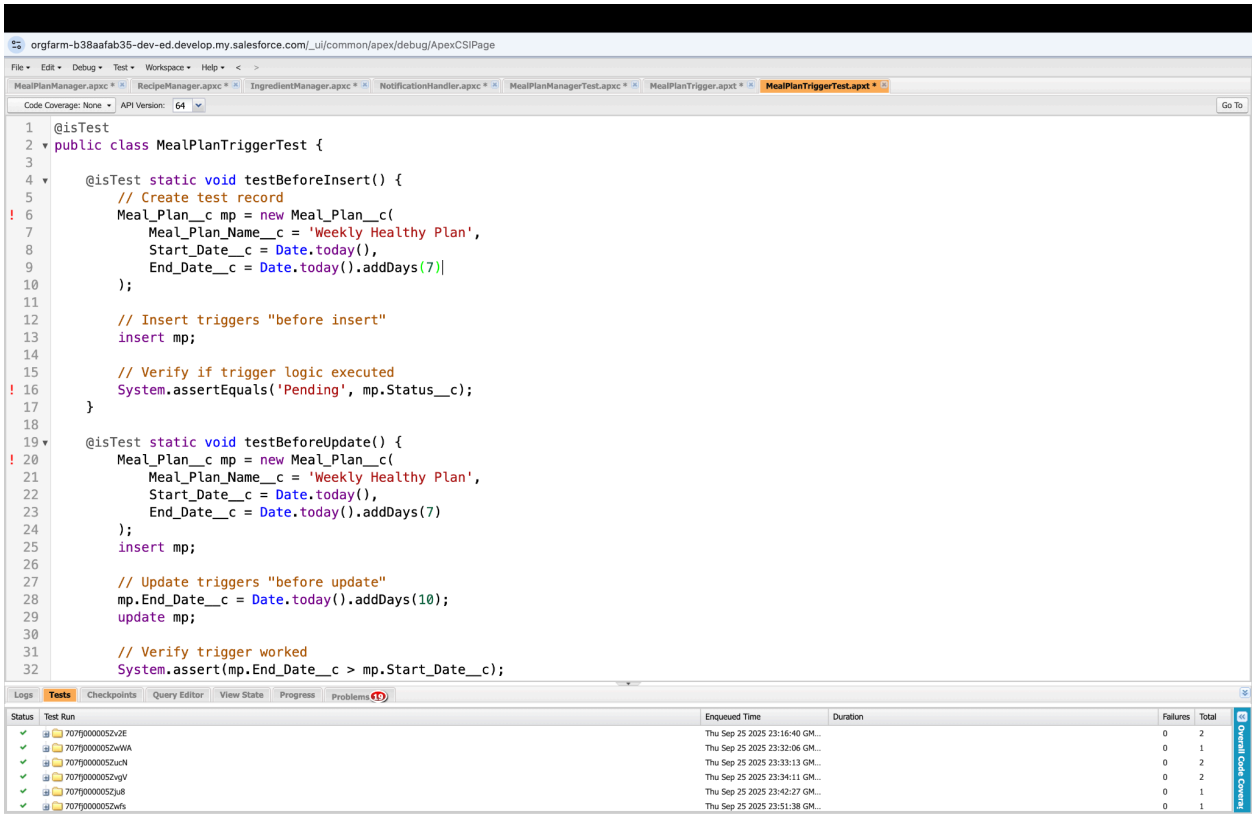

orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
MealPlanManager.apxc *
RecipeManager.apxc *
IngredientManager.apxc *
NotificationHandler.apxc *
MealPlanManagerTest.apxc *
Code Coverage: None
API Version: 64
Go To
1 @isTest
2 public class MealPlanManagerTest {
3
4 @isTest
5 static void testCalculateTotalIngredient() {
6 // Create dummy Ingredients
7 List<Ingredient__c> ingredients = new List<Ingredient__c>();
8 ingredients.add(new Ingredient__c(Quantity__c = 2));
9 ingredients.add(new Ingredient__c(Quantity__c = 3));
10
11 Decimal total = MealPlanManager.calculateTotalIngredient(ingredients);
12 System.assertEquals(5, total, 'Total quantity should be 5');
13 }
14
15 @isTest
16 static void testGetRecipesInMealPlan() {
17 // Create a dummy Meal Plan
18 Meal_Plan__c mealPlan = new Meal_Plan__c(Name = 'Test Plan');
19 insert mealPlan;
20
21 // Create Recipes linked to it
22 Recipe__c r1 = new Recipe__c(Name = 'Recipe A', Meal_Plan_Lookup__c = mealPlan.Id);
23 Recipe__c r2 = new Recipe__c(Name = 'Recipe B', Meal_Plan_Lookup__c = mealPlan.Id);
24 insert new List<Recipe__c>{r1, r2};
25
26 List<String> recipes = MealPlanManager.getRecipesInMealPlan(mealPlan.Id);
27 System.assertEquals(2, recipes.size(), 'Meal plan should return 2 recipes');
28 }
29 }
30
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems 10
Status
Test Run
Enqueued Time
Duration
Failures
Total
707fj000005Zqtm Thu Sep 25 2025 23:14:25 GM... 0 1
707fj000005Zvrl Thu Sep 25 2025 23:15:17 GM... 0 1
707fj000005Zpug Thu Sep 25 2025 23:15:56 GM... 0 1
707fj000005Zv2E Thu Sep 25 2025 23:16:40 GM... 0 2
707fj000005ZwWA Thu Sep 25 2025 23:32:06 GM... 0 1
707fj000005ZucN Thu Sep 25 2025 23:33:13 GM... 0 2
707fj000005ZvgV Thu Sep 25 2025 23:34:11 GM... 0 2
Overall Code Coverage

Apex Triggers (before/after insert/update/delete)

What is an Apex Trigger?

- A **trigger** is a piece of code that executes **before or after a record is inserted, updated, deleted, or undeleted**.
- Triggers allow you to automate **business logic** that cannot be done using workflow rules or Process Builder alone.
- **Types:**
    - **Before triggers:** Execute **before** a record is saved to the database. Used for validation or modifying record values.
    - **After triggers:** Execute **after** a record is saved. Used for actions that need record IDs or related objects.





Trigger Design Pattern

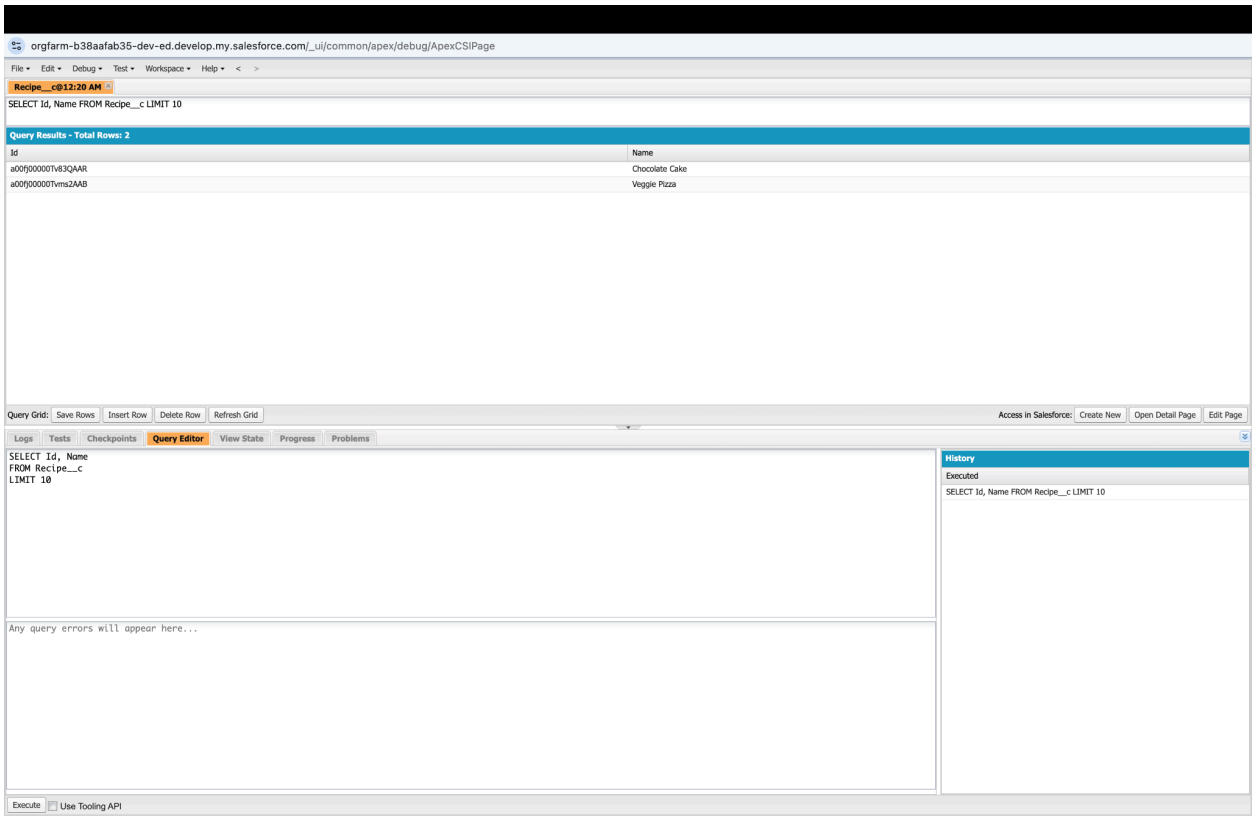
Trigger Design Pattern in Salesforce is used to organize and maintain Apex triggers efficiently. Instead of writing all logic in the trigger itself, a **Handler Class** is created to manage the actual logic for `before` or `after` operations. This pattern ensures:

- Separation of concerns (trigger only calls handler methods).
- Easy maintenance and testing.
- Scalability and bulk-safe operations.
  Triggers are lightweight and delegate business logic to handler classes, improving code quality and governance.

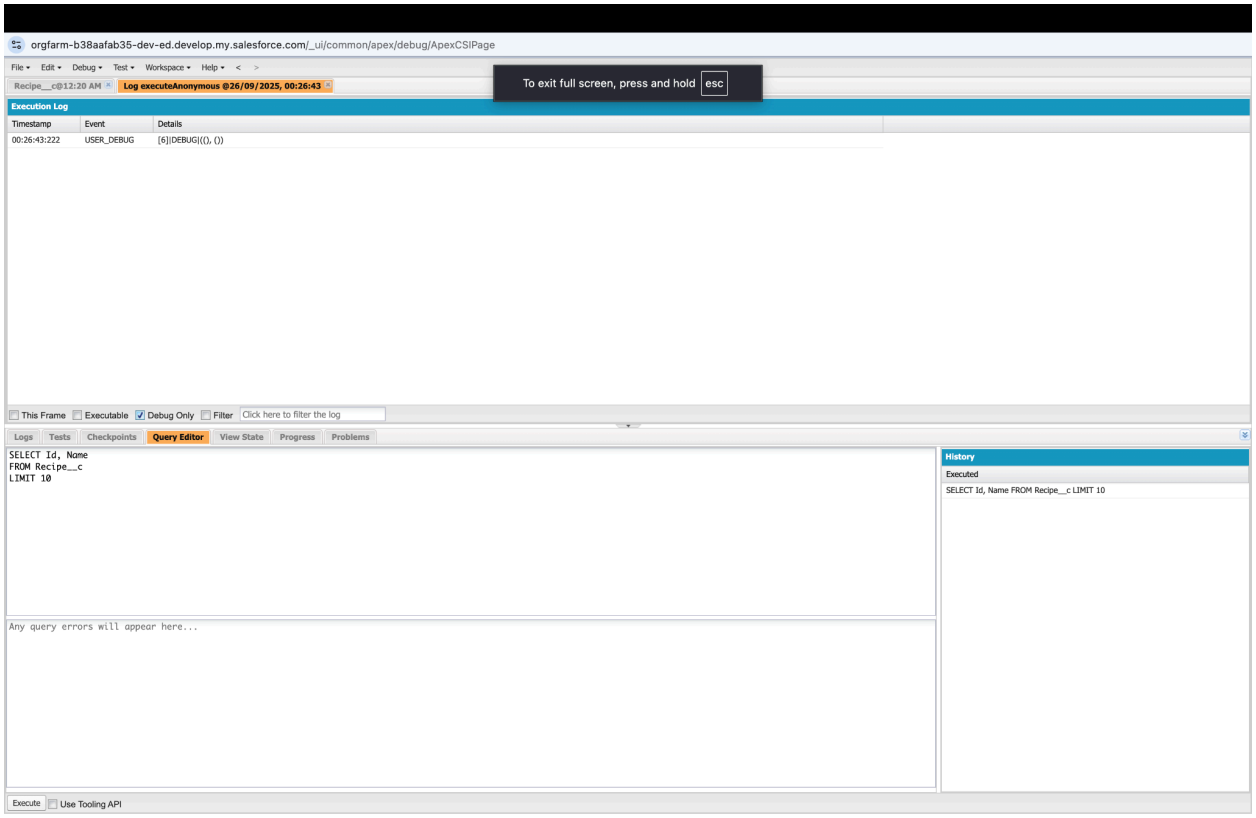
SOQL & SOSL

- **SOQL (Salesforce Object Query Language)**:
  Used to **query records from a single object or related objects**.
  Example: Get all Meal Plans with a certain name.

- **SOSL (Salesforce Object Search Language)**:
  Used to **search across multiple objects and fields**.
  Example: Find any record (Meal Plan, Recipe, Ingredient) containing "Salad".



For SOQL



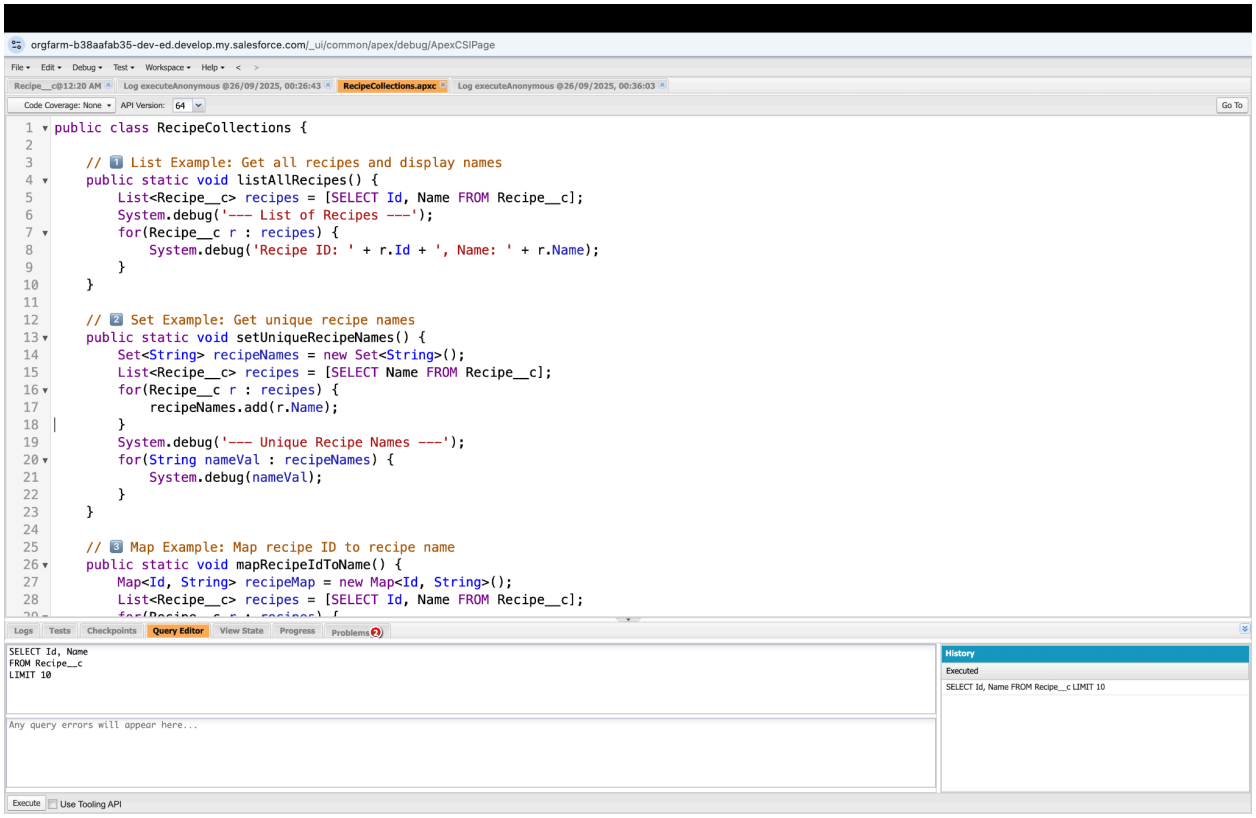
For SOSL

Collections: List, Set, Map

What Are Collections?

Collections in Apex are **data structures** that allow you to store multiple values in a single variable. There are **3 main types**:

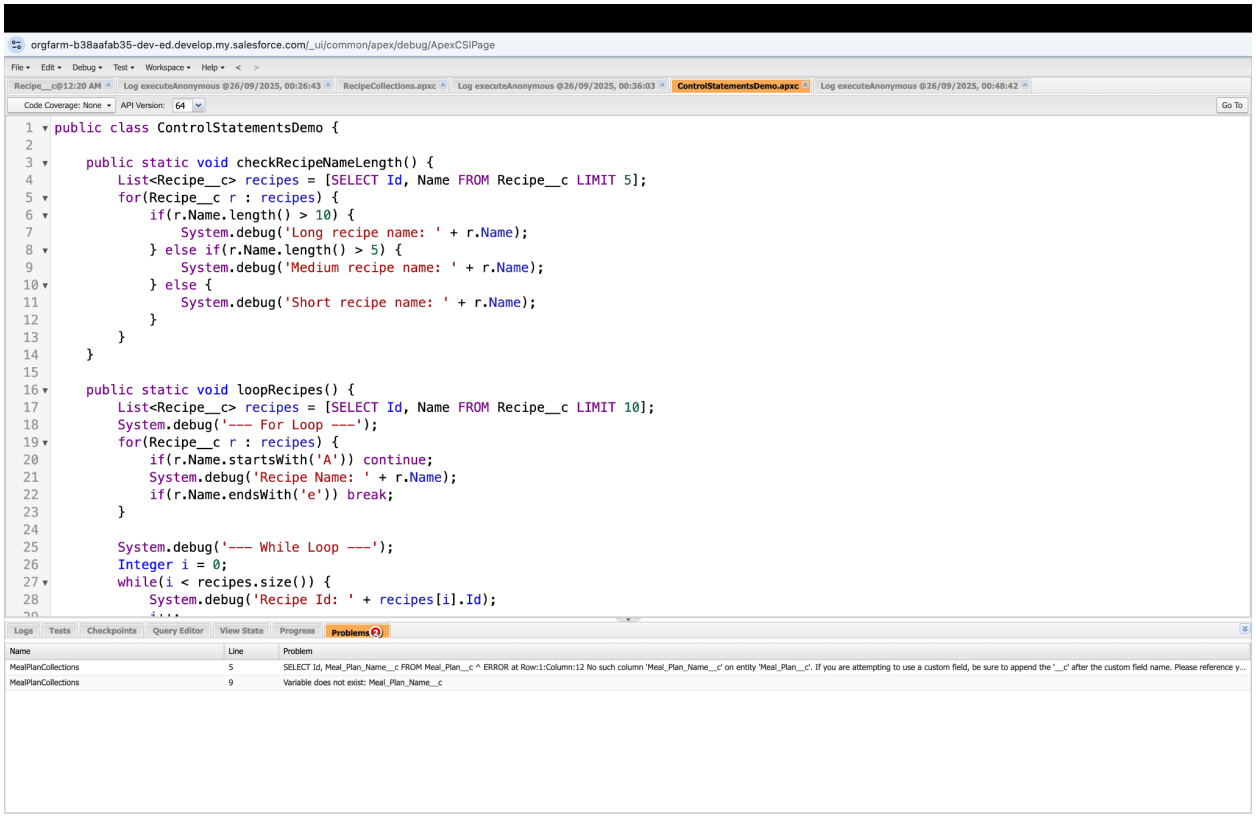
1. **List** – An ordered collection of elements. Duplicates allowed.
2. **Set** – An unordered collection of **unique** elements. No duplicates.
3. **Map** – A collection of **key-value pairs**. Each key is unique.

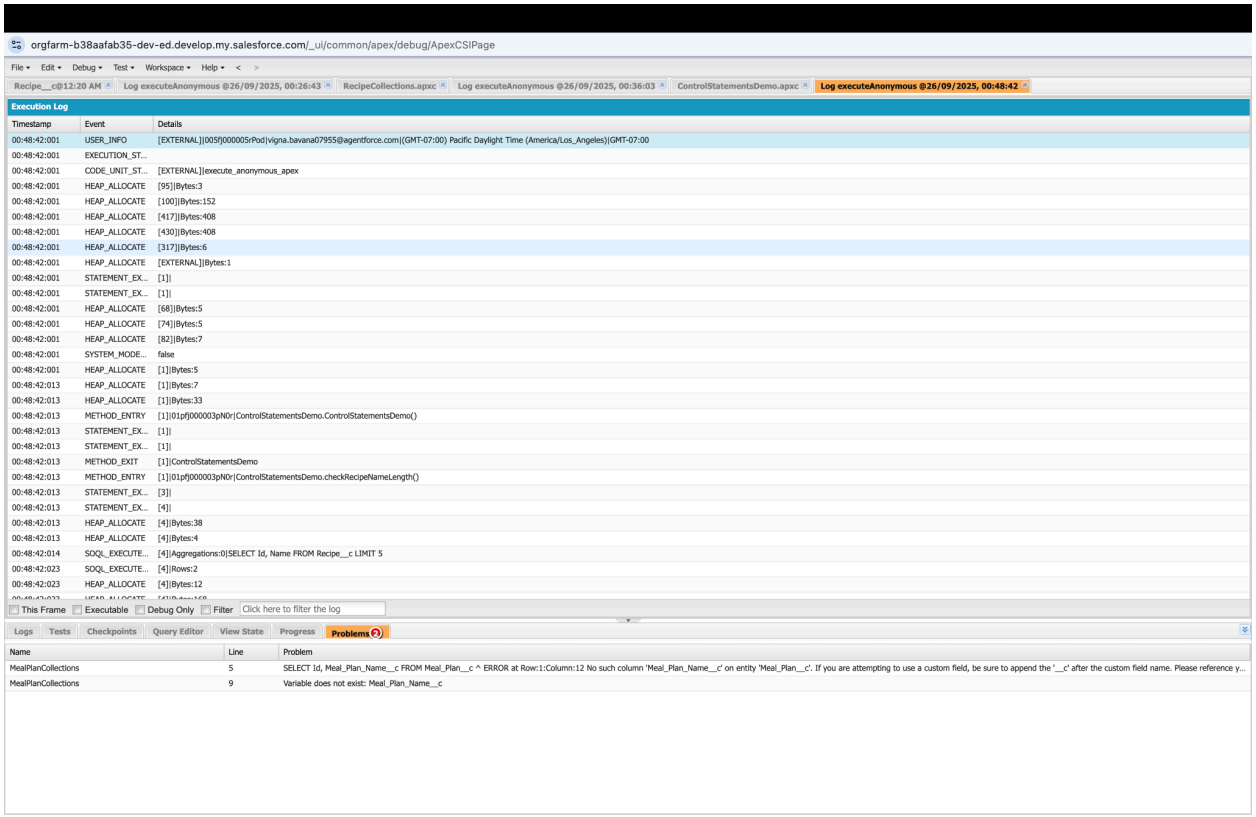


Control Statements

Control statements allow you to control the **flow of execution** in Apex code. These include:

1. **Decision-making statements**: `if`, `else if`, `else`, `switch`.
2. **Loops**: `for`, `while`, `do-while`.
3. **Jump statements**: `break`, `continue`, `return`.


orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
Recipe__c@12:20 AM
Log executeAnonymous @26/09/2025, 00:26:43
RecipeCollections.apxc
Log executeAnonymous @26/09/2025, 00:36:03
ControlStatementsDemo.apxc
Log executeAnonymous @26/09/2025, 00:48:42
Code Coverage: None
API Version: 64
Go To
1 public class ControlStatementsDemo {
2
3 public static void checkRecipeNameLength() {
4 List<Recipe__c> recipes = [SELECT Id, Name FROM Recipe__c LIMIT 5];
5 for(Recipe__c r : recipes) {
6 if(r.Name.length() > 10) {
7 System.debug('Long recipe name: ' + r.Name);
8 } else if(r.Name.length() > 5) {
9 System.debug('Medium recipe name: ' + r.Name);
10 } else {
11 System.debug('Short recipe name: ' + r.Name);
12 }
13 }
14 }
15
16 public static void loopRecipes() {
17 List<Recipe__c> recipes = [SELECT Id, Name FROM Recipe__c LIMIT 10];
18 System.debug('--- For Loop ---');
19 for(Recipe__c r : recipes) {
20 if(r.Name.startsWith('A')) continue;
21 System.debug('Recipe Name: ' + r.Name);
22 if(r.Name.endsWith('e')) break;
23 }
24
25 System.debug('--- While Loop ---');
26 Integer i = 0;
27 while(i < recipes.size()) {
28 System.debug('Recipe Id: ' + recipes[i].Id);
Logs Tests Checkpoints Query Editor View State Progress Problems 2
Name Line Problem
MealPlanCollections 5 SELECT Id, Meal_Plan_Name__c FROM Meal_Plan__c ^ ERROR at Row:1:Column:12 No such column 'Meal_Plan_Name__c' on entity 'Meal_Plan__c'. If you are attempting to use a custom field, be sure to append the '__c' after the custom field name. Please reference y...
MealPlanCollections 9 Variable does not exist: Meal_Plan_Name__c

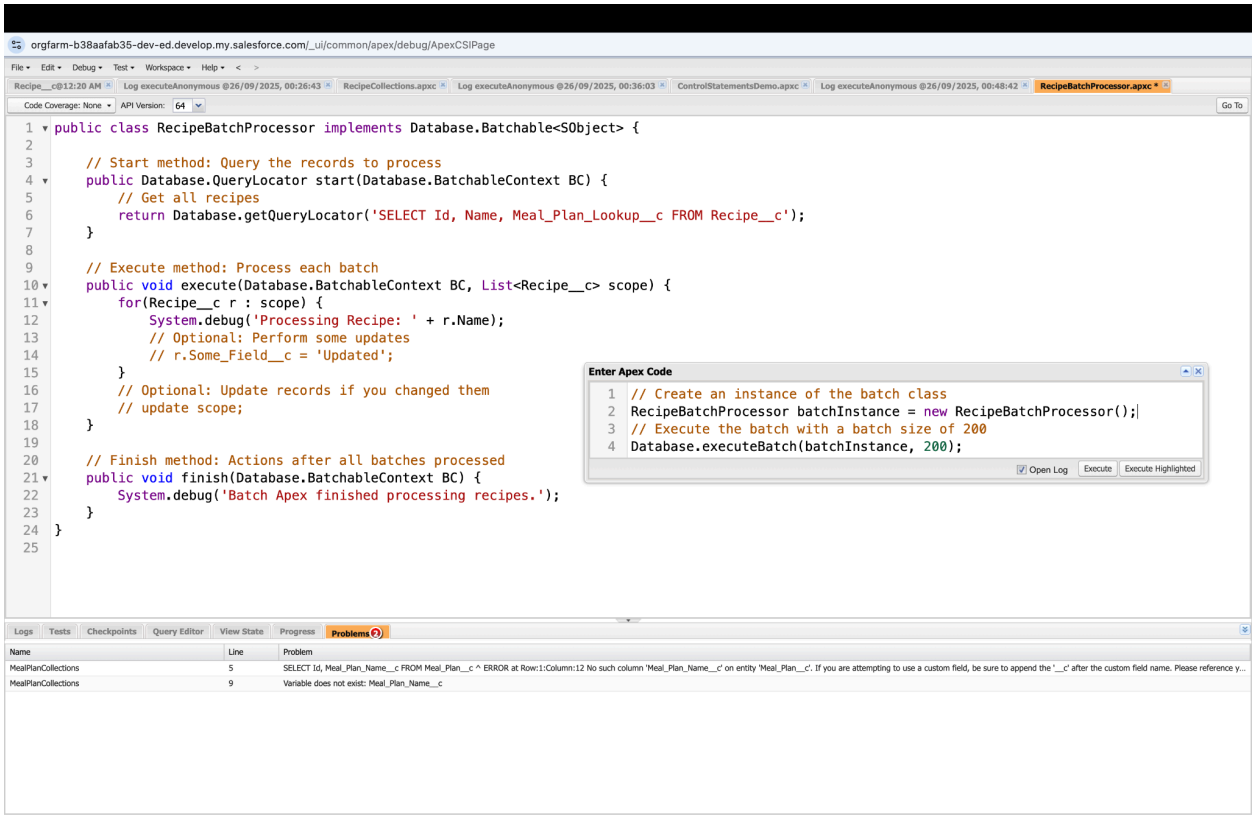

orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
Recipe__c@12:20 AM
Log executeAnonymous @26/09/2025, 00:26:43
RecipeCollections.apxc
Log executeAnonymous @26/09/2025, 00:36:03
ControlStatementsDemo.apxc
Log executeAnonymous @26/09/2025, 00:48:42
Execution Log
Timestamp Event Details
00:48:42:001 USER_INFO [EXTERNAL]|005fj000005rPod|vigna.bavana07955@agentforce.com|(GMT-07:00) Pacific Daylight Time (America/Los_Angeles)|GMT-07:00
00:48:42:001 EXECUTION_ST...
00:48:42:001 CODE_UNIT_ST... [EXTERNAL]|execute_anonymous_apex
00:48:42:001 HEAP_ALLOCATE [95]|Bytes:3
00:48:42:001 HEAP_ALLOCATE [100]|Bytes:152
00:48:42:001 HEAP_ALLOCATE [417]|Bytes:408
00:48:42:001 HEAP_ALLOCATE [430]|Bytes:408
00:48:42:001 HEAP_ALLOCATE [317]|Bytes:6
00:48:42:001 HEAP_ALLOCATE [EXTERNAL]|Bytes:1
00:48:42:001 STATEMENT_EX... [1]|
00:48:42:001 STATEMENT_EX... [1]|
00:48:42:001 HEAP_ALLOCATE [68]|Bytes:5
00:48:42:001 HEAP_ALLOCATE [74]|Bytes:5
00:48:42:001 HEAP_ALLOCATE [82]|Bytes:7
00:48:42:001 SYSTEM_MODE... false
00:48:42:001 HEAP_ALLOCATE [1]|Bytes:5
00:48:42:013 HEAP_ALLOCATE [1]|Bytes:7
00:48:42:013 HEAP_ALLOCATE [1]|Bytes:33
00:48:42:013 METHOD_ENTRY [1]|01pfj000003pN0r|ControlStatementsDemo.ControlStatementsDemo()
00:48:42:013 STATEMENT_EX... [1]|
00:48:42:013 STATEMENT_EX... [1]|
00:48:42:013 METHOD_EXIT [1]|ControlStatementsDemo
00:48:42:013 METHOD_ENTRY [1]|01pfj000003pN0r|ControlStatementsDemo.checkRecipeNameLength()
00:48:42:013 STATEMENT_EX... [3]|
00:48:42:013 STATEMENT_EX... [4]|
00:48:42:013 HEAP_ALLOCATE [4]|Bytes:38
00:48:42:013 HEAP_ALLOCATE [4]|Bytes:4
00:48:42:014 SOQL_EXECUTE... [4]|Aggregations:0|SELECT Id, Name FROM Recipe__c LIMIT 5
00:48:42:023 SOQL_EXECUTE... [4]|Rows:2
00:48:42:023 HEAP_ALLOCATE [4]|Bytes:12
This Frame Executable Debug Only Filter Click here to filter the log
Logs Tests Checkpoints Query Editor View State Progress Problems 2
Name Line Problem
MealPlanCollections 5 SELECT Id, Meal_Plan_Name__c FROM Meal_Plan__c ^ ERROR at Row:1:Column:12 No such column 'Meal_Plan_Name__c' on entity 'Meal_Plan__c'. If you are attempting to use a custom field, be sure to append the '__c' after the custom field name. Please reference y...
MealPlanCollections 9 Variable does not exist: Meal_Plan_Name__c

Batch Apex

Batch Apex is used to process **large amounts of records** asynchronously (in the background).

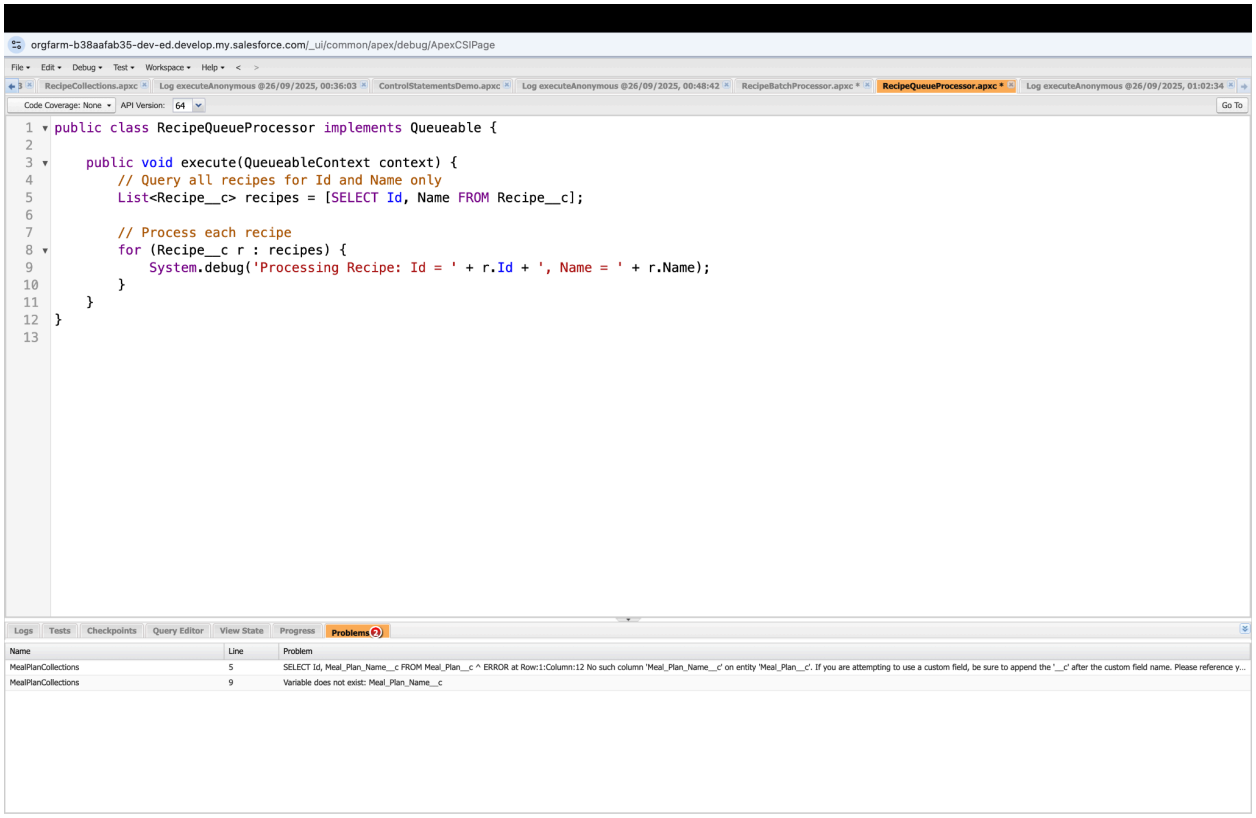
Each batch executes a subset of records (called **batch size**), preventing governor limit issues.

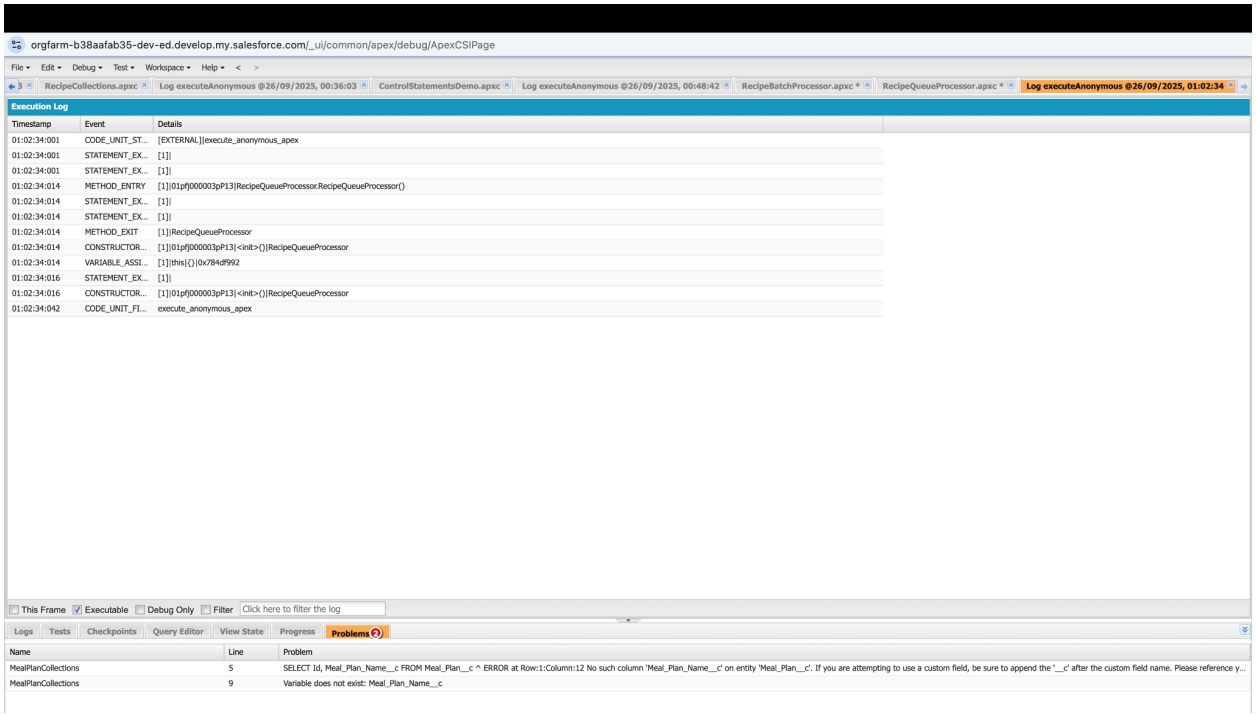
It implements the `Database.Batchable<SObject>` interface.



Queueable Apex:

Queueable Apex is a Salesforce feature that allows you to run **asynchronous operations** (background jobs) in a **flexible and scalable** way. It is similar to Batch Apex but simpler for smaller tasks and allows **chaining jobs**.


orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
RecipeCollections.apxc
Log executeAnonymous @26/09/2025, 00:36:03
ControlStatementsDemo.apxc
Log executeAnonymous @26/09/2025, 00:48:42
RecipeBatchProcessor.apxc *
RecipeQueueProcessor.apxc *
Log executeAnonymous @26/09/2025, 01:02:34
Code Coverage: None
API Version: 64
Go To
1 public class RecipeQueueProcessor implements Queueable {
2
3 public void execute(QueueableContext context) {
4 // Query all recipes for Id and Name only
5 List<Recipe__c> recipes = [SELECT Id, Name FROM Recipe__c];
6
7 // Process each recipe
8 for (Recipe__c r : recipes) {
9 System.debug('Processing Recipe: Id = ' + r.Id + ', Name = ' + r.Name);
10 }
11 }
12 }
13
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems 2
Name
Line
Problem
MealPlanCollections
5
SELECT Id, Meal_Plan_Name__c FROM Meal_Plan__c ^ ERROR at Row:1:Column:12 No such column 'Meal_Plan_Name__c' on entity 'Meal_Plan__c'. If you are attempting to use a custom field, be sure to append the '__c' after the custom field name. Please reference y...
MealPlanCollections
9
Variable does not exist: Meal_Plan_Name__c

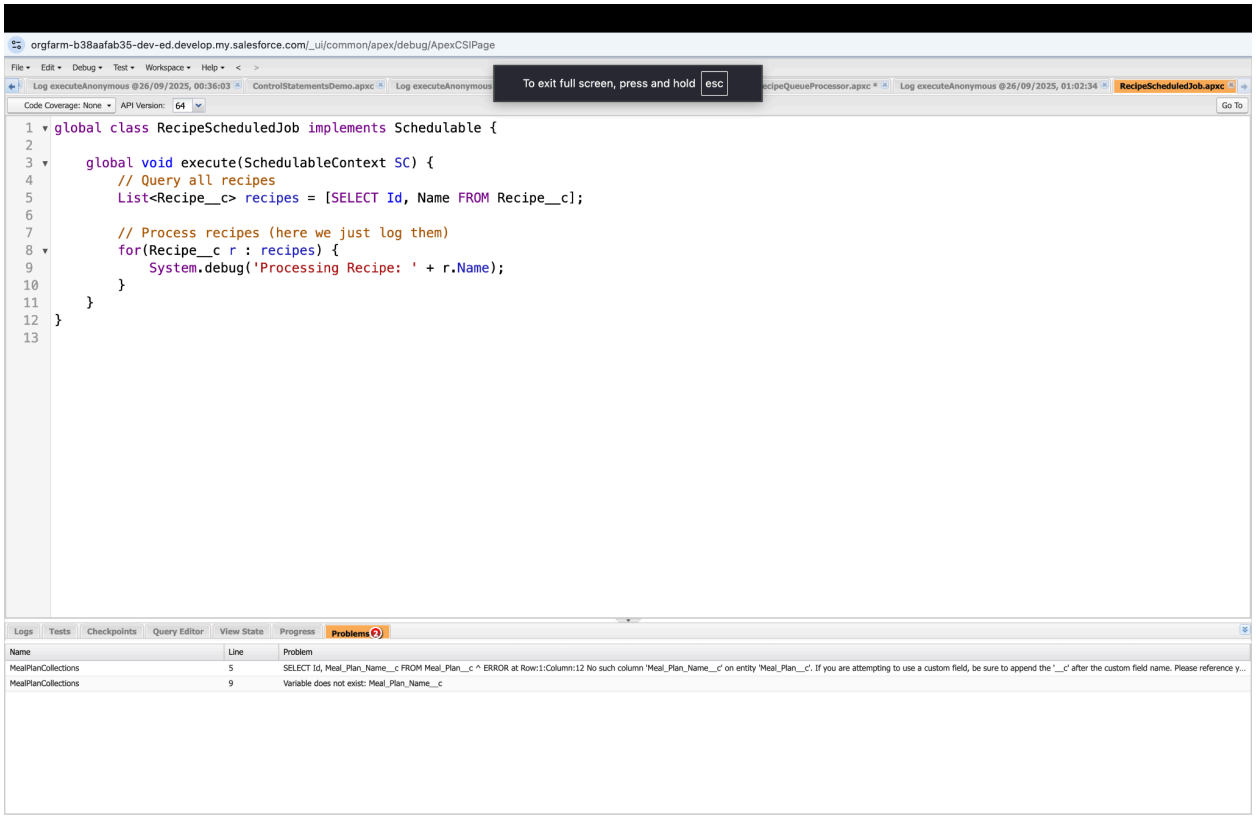

orgfarm-b38aafab35-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
RecipeCollections.apxc
Log executeAnonymous @26/09/2025, 00:36:03
ControlStatementsDemo.apxc
Log executeAnonymous @26/09/2025, 00:48:42
RecipeBatchProcessor.apxc *
RecipeQueueProcessor.apxc *
Log executeAnonymous @26/09/2025, 01:02:34
Execution Log
Timestamp
Event
Details
01:02:34:001 CODE_UNIT_ST... [EXTERNAL]|execute_anonymous_apex
01:02:34:001 STATEMENT_EX... [1]|
01:02:34:001 STATEMENT_EX... [1]|
01:02:34:014 METHOD_ENTRY [1]|01pfj000003pP13|RecipeQueueProcessor.RecipeQueueProcessor()
01:02:34:014 STATEMENT_EX... [1]|
01:02:34:014 STATEMENT_EX... [1]|
01:02:34:014 METHOD_EXIT [1]|RecipeQueueProcessor
01:02:34:014 CONSTRUCTOR... [1]|01pfj000003pP13|<init>()|RecipeQueueProcessor
01:02:34:014 VARIABLE_ASSI... [1]|this|{}|0x784df992
01:02:34:016 STATEMENT_EX... [1]|
01:02:34:016 CONSTRUCTOR... [1]|01pfj000003pP13|<init>()|RecipeQueueProcessor
01:02:34:042 CODE_UNIT_FI... execute_anonymous_apex
This Frame
Executable
Debug Only
Filter
Click here to filter the log
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems 2
Name
Line
Problem
MealPlanCollections
5
SELECT Id, Meal_Plan_Name__c FROM Meal_Plan__c ^ ERROR at Row:1:Column:12 No such column 'Meal_Plan_Name__c' on entity 'Meal_Plan__c'. If you are attempting to use a custom field, be sure to append the '__c' after the custom field name. Please reference y...
MealPlanCollections
9
Variable does not exist: Meal_Plan_Name__c

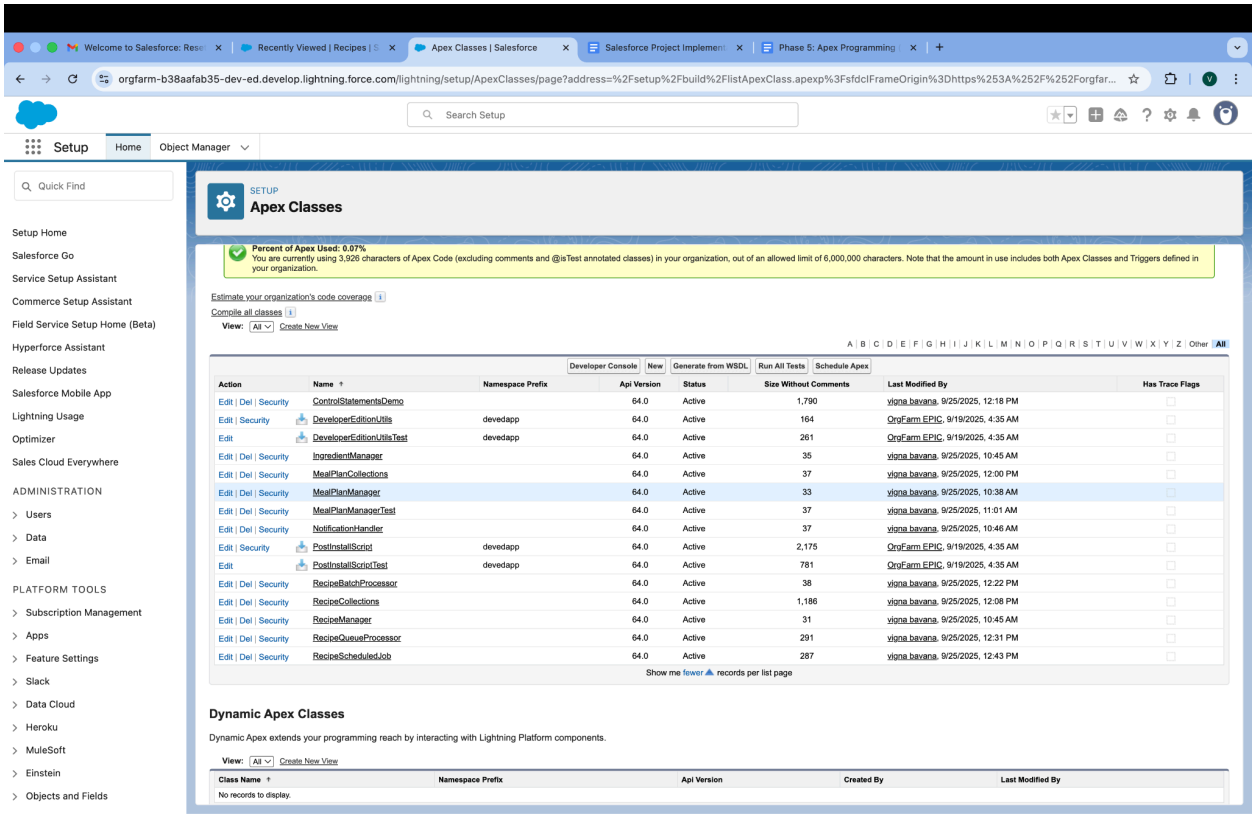
Scheduled Apex:

Scheduled Apex allows you to run **Apex classes at specified times** or on a recurring schedule. It is ideal for **automated, time-based operations** in Salesforce.

Use Case:

- Automatically update recipe records daily.
- Send email notifications at a scheduled time.
- Perform cleanup tasks or batch processing without manual intervention.





Future Methods

- **Purpose:** Run long-running operations asynchronously, without blocking the user interface.
- **Key Points:**
    - Annotated with `@future`.
    - Can only accept **primitive data types or collections of primitives** as parameters.
    - Cannot return a value.
- **Example:**

```
public class RecipeAsync {
    @future
    public static void sendRecipeNotification(Id recipeId) {
```

```
        Recipe__c r = [SELECT Name FROM Recipe__c WHERE Id =
:recipeId];
        System.debug('Sending notification for recipe: ' + r.Name);
    }
}
```

Exception Handling

- **Purpose:** Gracefully handle runtime errors in Apex.
- **Key Points:**
    - Use `try`, `catch`, and `finally` blocks.
    - Can throw custom exceptions using `throw new CustomException()`.
- **Example:**

```
public class RecipeHandler {
    public static void updateRecipeCalories(Id recipeId, Decimal
calories) {
        try {
            Recipe__c r = [SELECT Name FROM Recipe__c WHERE Id =
:recipeId];
            r.Calories__c = calories;
            update r;
        } catch (Exception e) {
            System.debug('Error updating recipe: ' + e.getMessage());
        } finally {
            System.debug('Update attempt finished.');
        }
    }
}
```

Test Classes

- **Purpose:** Validate that Apex code works correctly and to meet Salesforce deployment requirements.
- **Key Points:**
    - Annotated with `@isTest`.
    - Create test records, execute logic, and assert outcomes.
    - Must cover at least **75% of code** to deploy to production.
- **Example:**

```
@isTest
public class RecipeTest {
    static testMethod void testRecipeNotification() {
        Recipe__c r = new Recipe__c(Name = 'Test Salad');
        insert r;

        Test.startTest();
        RecipeAsync.sendRecipeNotification(r.Id);
        Test.stopTest();

        System.assert(r.Name == 'Test Salad');
    }
}
```

Asynchronous Processing

- **Purpose:** Handle large operations without timing out.
- **Types in Salesforce:**
    - **Future Methods** → for simple async tasks.
    - **Queueable Apex** → chainable and flexible.

- **Batch Apex** → process large datasets in chunks.
- **Scheduled Apex** → run tasks at specific times.

Key Advantage: Users don’t have to wait for the process to finish; Salesforce executes it in the background.