PHASE – 5 Apex Programming (Developer)

In this phase of my project, I mainly focused on integrating Salesforce custom objects with Apex logic and ensuring that the deployment process worked correctly.

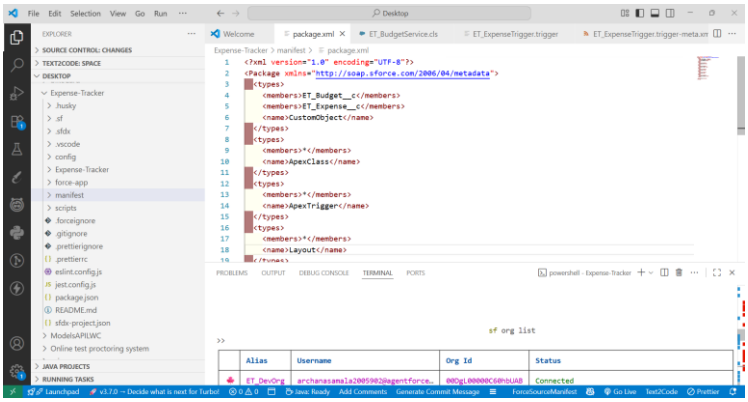
Create a new SFDX project:

sfdx force:project:create -n Expense-Tracker

cd Expense-Tracker

Authorize your Dev Org (this opens the browser to login). Use a short alias:

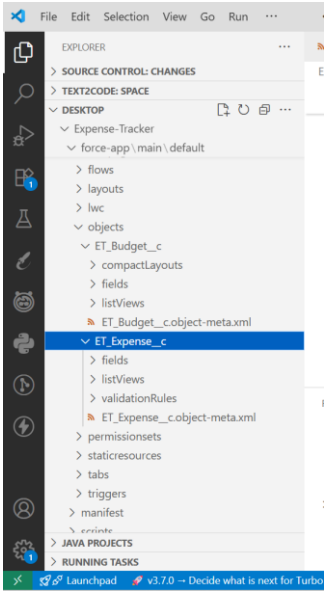
sfdx auth:web:login -a ET_DevOrg



1.Verification of Objects and Fields

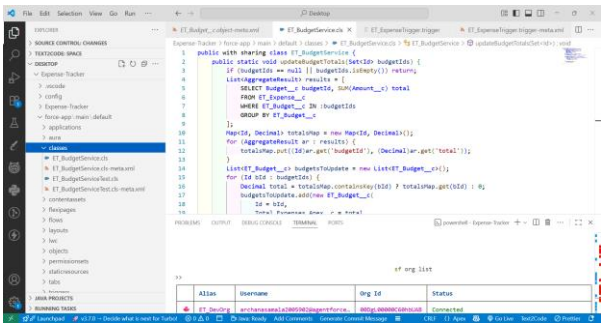
First, I verified that the required custom objects and fields existed in my Salesforce Developer Org.

- The main objects used were ET_Budget__c and ET_Expense__c.
- On ET_Budget__c, I ensured that fields like Threshold_Amount__c and Total_Expenses_Apex__c were created.
- On ET_Expense__c, I checked for fields such as Amount__c, Expense_Date__c, Category__c, and the lookup relationship Budget__c pointing to ET_Budget__c.



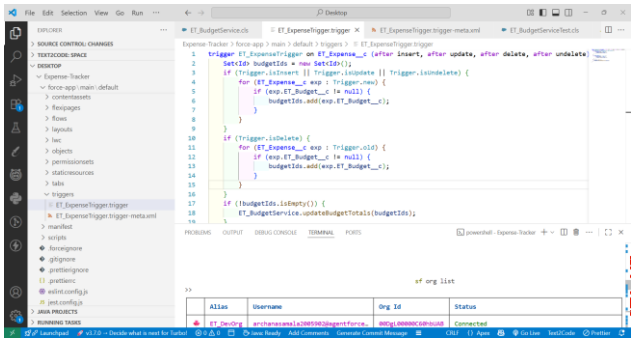
2. Development of Apex Class

I created a service class called **ET_BudgetService.cls**. This class ensures that the budget always reflects the correct total expenses.



3. Creation of Trigger

I created a trigger called **ET_ExpenseTrigger.trigger** on the ET_Expense__c object. This trigger is the connector between expenses and budgets.

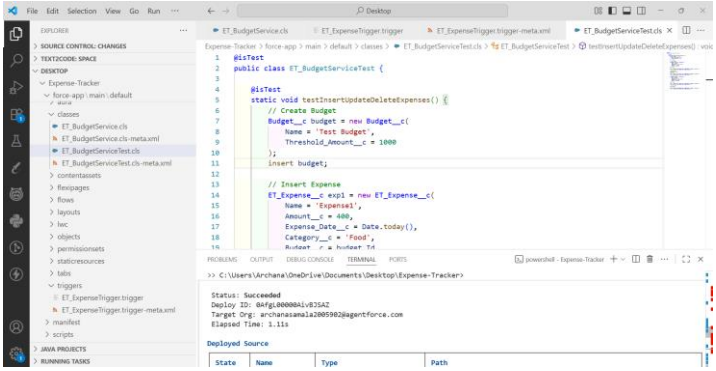


4. Writing Test Class

I developed a test class named **ET_BudgetServiceTest.cls**.

- In the test class, I created test data for budgets and expenses.
- I tested different scenarios like inserting an expense, updating an expense amount, adding multiple expenses, and deleting an expense.
- After each operation, I used SOQL queries to fetch the budget and validated the total using **System.assertEquals()**.

This test class ensures proper code coverage and validates that the trigger and service class logic works correctly.



5. Deployment Process

Deployment was done using the Salesforce CLI (sf project deploy).
The steps I followed were:

1. Deploy **custom objects** (ET_Budget__c and ET_Expense__c).
2. Deploy **Apex classes** (ET_BudgetService and ET_BudgetServiceTest).
3. Deploy **triggers** (ET_ExpenseTrigger).
4. Finally, run the **Apex tests** to confirm successful execution and coverage.

I have deployed my custom objects, apex classes and triggers using the commands as follows:

- sf project deploy start --metadata CustomObject:ET_Budget__c --target-org ET_DevOrg
- sf project deploy start --metadata CustomObject:ET_Expense__c --target-org ET_DevOrg
- sf project deploy start --source-dir force-app/main/default/classes --target-org ET_DevOrg
- sf project deploy start --source-dir force-app/main/default/triggers --target-org ET_DevOrg

6. Outcome of Phase-5

By the end of this phase, I was able to:

- Successfully deploy objects, Apex classes, and triggers into my Salesforce org.
- Ensure that budgets correctly reflect total expenses in real-time.
- Gain hands-on experience in debugging deployment errors and fixing metadata issues.

This phase was a major milestone because it connected all the previous setups into a working, automated process inside Salesforce.