Phase 5: Apex Programming (Developer)

 Goal: Add advanced server-side logic to handle Insurance-specific operations beyond clicks/config.

Classes & Objects

Create an `InsurancePolicyService` class to manage policy operations (create, renew, cancel).

Create a `ClaimService` class for claim validation and settlement.

Use reusable helper methods for calculations (e.g., premium, claim eligibility).

Apex Triggers

On Policy Insert: Prevent duplicate active policies for the same customer.

On Claim Insert: Auto-validate against policy coverage.

On Payment Update: Mark policy as *Active* when payment confirmed.

Trigger Design Pattern

Use a centralized **Trigger Handler Framework** to keep triggers clean.
Example: `PolicyTriggerHandler.cls` handles before/after logic.

SOQL & SOSL

Query policies for a customer:

```
SELECT Id, Policy_Type__c, Status__c FROM Policy__c WHERE
Customer__c = :custId
```

Search claims via SOSL when customer submits reference number.

Collections: List, Set, Map

List: Store policies for batch updates.

Set: Prevent duplicate claim IDs.

Map: Map customer → active policies for quick lookup.

Control Statements

If claim date < policy start date → throw error.

If policy status != Active → prevent claim submission.

Batch Apex

Monthly job to auto-expire policies past end date.

Bulk update claim statuses (e.g., *Closed* after payout).

Queueable Apex

Send bulk notifications for policy renewals (async).

Scheduled Apex

Every morning at 8 AM → email manager with pending claims.

Every 1st of month → generate premium collection summary.

Future Methods

External API integration (e.g., Insurance Regulatory DB check).

Exception Handling

Handle cases where policy doesn’t exist but claim submitted.

Custom exception class: `InvalidClaimException`.

Test Classes

Create test data for policies, claims, customers.

Test: Overlapping policies, expired policies, invalid claim dates.

Ensure **75%+ code coverage** for deployment.

Asynchronous Processing

Use **Batch + Queueable + Future** for:

Policy renewals.

Bulk notifications.

External insurance DB sync.

□ This phase ensures **robust backend logic** so your **Insurance Portal** can:

Auto-validate claims.

Handle bulk renewals.

Stay compliant with industry workflows.
