Extension of Custom Exception ABCBank Application

**Background (Recap)**

ABCBank earlier implemented a withdrawal and exception-handling framework to ensure compliance with financial rules and fraud detection.

After stabilizing the withdrawal system, ABCBank’s Insurance Division raised a new requirement: integrate insurance products with accounts. Customers may hold multiple types of insurance (Car, Health, Life, etc.) tied to their bank account

**New Business Requirement**

ABCBank now requires:

1. Each Account must support multiple insurance products.
2. Insurance products must be polymorphic (CarInsurance, HealthInsurance, LifeInsurance, etc.).
3. The system should allow generating a report mapping:

Map<Insurance, List<Account>> insuranceRecord;

This means:

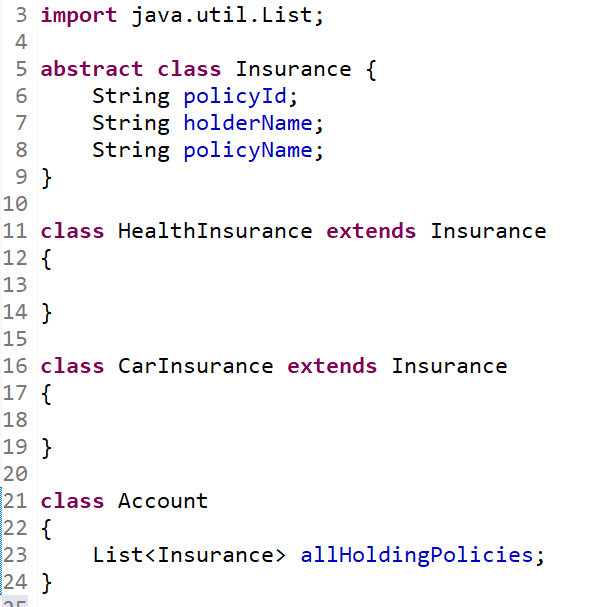
* For each **Insurance type**, retrieve all **Accounts** holding that insurance.
* Enable business insights like:
  + “How many accounts have Health Insurance?”
  + “Which customers have Car + Life Insurance together?”

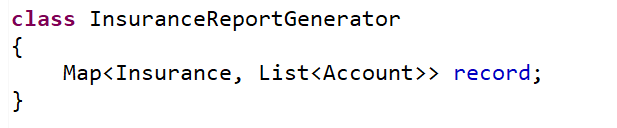
**Enhanced Design**

**3.1 Entities**

1. **Insurance (Abstract Class)**
   * Represents generic insurance.
   * Extended by concrete classes:
     + CarInsurance
     + HealthInsurance
     + LifeInsurance
2. **Account**
   * Already stores balance, daily limits, and blocked status.
   * Now enhanced with:
     + List<Insurance> allHoldingInsurancePolicies;
   * This models the one-to-many relationship between Account and Insurance
3. **BankingProcess**

* Continues to manage withdrawals and fraud detection.
* Now includes helper methods to register insurance for accounts.





**Benefit**

With this enhancement, ABCBank’s system now supports multi-product account management, allowing not just financial transactions but also insurance mapping. The solution demonstrates how object-oriented design (composition + polymorphism) and collections (Map, List) can model real-world banking and insurance requirements elegantly.

**Next Extension**

