**Problem Statement: Digital Payment System**

* A **fintech company** wants to build a **digital payment system** that allows users to:
  + **Transfer money** between wallets or bank accounts.
  + **Top up wallets** via bank transfers or cards.
  + **Process payments** for merchants securely.
  + **Detect fraudulent transactions** based on user behavior.
* The system must be **secure, scalable, and maintainable** while ensuring **seamless transactions** across different services.

# Strategic Design Principles.

1. **Collaboration between business experts and technical team**
2. **Domain Scope**

(Define the area / end users for the application)

1. **Domain description**

(Describing the complete domain briefly including code business domains and functionalities.)

1. **Ubiquitous Language**

* Transaction – A payment attempt by a user
* Authorization – Request to reserve or validate funds
* Capture – Settlement of authorized funds
* Refund – Reversal of completed transaction
* Marchant – A business entity accepting payments
* Wallet – A stored-value account for customer payments
* Settlement – Transfer of funds to merchant after transaction completion.
* Payment Gateway – External service used for payment processing.

1. **Domain Analysis**

(Identify code sub-domain, supporting sub-domain, and generic sub-domain)

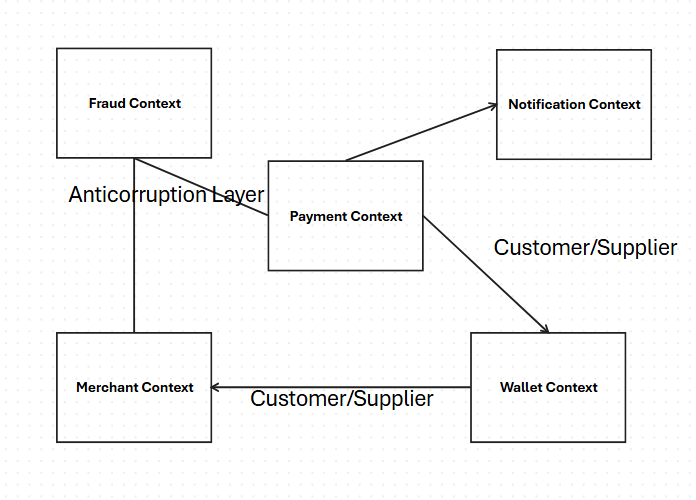
* 1. **Core domains**
* Transaction management – Handling initiation, authorization, processing and settlement of payments.
  1. **Supporting subdomains**
* Fraud Detection – Risk scoring and Fraud prevention.
* Merchant Management – Manage merchant onboarding, configuration and settlements.
* Customer Wallet management – Manage stored balances and loyalty points.
  1. **Generic subdomains**
* Authentication and authorization
* Audit and logging
* Notification system (Email, SMS)
* Currency Conversion service.

1. **Bounded Contexts**

* Transaction context – Manages payment transaction lifecycle.
* Wallet Context – Manages customer wallets and loyalty balances.
* Merchant Context – Handles merchant setup, onboarding, settlements.
* Fraud Context – Calculate risk scores and flags suspicious transactions.
* Notification Context – Sends alert for transaction events.

1. **Context Mapping Workshop** 
   1. Purpose of the workshop
   * Build a shared understanding of the bounded contexts
   * Identify relationships between different domains/subdomain.
   * Highlight integration points and communication patterns
   * Align on team boundaries and responsibilities.
   * Clarify ubiquitous languages used between contexts.
   1. Pre-workshop preparation
   * Stakeholders – Developers, Architects, Product managers, QA, Security team
   * Inputs needed – System high level overview, domain experts available for Q & A.
   1. Worksop Agenda

* Kickoff and context setting
  + Define digital payment system mission
  + Explain Domain Driven design basic (if needed)
* Brainstorming on Bounded contexts and code domains
* Map context relationships



# Tactical Design Principles

1. Key Entities

* Transaction – transactionId, customerId, merchantId, amount, currency, status, paymentMethod, timestamp
* Wallet – walletId, customerId, balance, currency, status,
* WalletTransaction – transactionId, walletId, amount, transactionType, timestamp.
* Merchant – merchantId, name, businessType, settlementBankAccount, status
* Settlement – settlementId, merchantId, totalAMount, settlementDate, status
* FraudCheck – fraudCheckId, transactionId, riskScore, riskLevel
* Notification – notificationid, sendId, type, templateId, deliveryStattus, timestamp.

1. Key Value Objects

* Money – amount, currency
* Address – street, city, postalCode, country
* BankAccount – accountNumber, Bank, ifscCode

1. Key Aggregates

* PaymentTransactionAggregate – PaymentTransaction, Authorization, Refund, Money
* WalletAggregate – Wallet, WalletTransaction, Money
* MerchantAggregate – Merchant, Settlement, BankAccount
* FraudCheckAggregate – FraudCheck, PaymentTransaction, Merchant

1. Key Repositories

* TransactionRepository – Manage payment transactions
* WalletRepository – Manage customer wallets
* MerchantRepository – Manage merchants and settlements,
* FraudCheckRepository – Manage risk checks and fraud records
* NotificationRepository – Mange notification history

1. Domain events

* PyamentInitiated – triggered when a customer starts a payment.
* PaymentAuthorized – triggered when payment authorization is successful,
* PaymentAuthorizationFailed – triggered when payment authorization fails
* PaymentCaptured – triggered when the authorized payment is successfully captured and confired.
* PaymentSettled – Triggered when the captured payment is transferred to the merchants bank accounts
* FraudDetected – triggered when a transaction is flagged as suspicious or fraudulent by fraud detection modules.

1. Application Services

* PaymentApplicaitonService – Initiate payment, authorize, capture, refund,
* WalletApplicaitonService – Credit/debit wallet, view balance.
* MerchantAPplicaitonService – register merchants, handle settlements,
* FraudApplicaitonService – Perform fraud analysis, block transactions
* NotificaitonAPplicaitonService – notify customers and merchants.