

# Examen Conception de Systèmes (Banque)

## UML à produire – Intermédiaire

(Scénarios + Réponses attendues)

### Conception UML – Scénarios bancaires (à dessiner manuellement)

#### Q1. Concevoir le système de 'Virement Interne' :

Produire un diagramme de classes (domain) + un diagramme de séquence pour le scénario 'virement validé'. (niveau intermédiaire)

#### Réponse :

Classes attendues: Customer, Account, Transfer, LedgerEntry, Money (Value Object), TransferService, FraudCheckService (option), NotificationService (option).

Relations: Customer 1..\* Account; Transfer relie from\_account/to\_account;

LedgerEntry 1..\* par Transfer.

Séquence: Client→API→TransferService: validate(idempotencyKey), lock accounts, check balance, persist transfer+ledger, commit, publish event.

Points attendus: notation UML correcte, relations et multiplicités appropriées

Diagramme de Classes - Système de Virement Interne

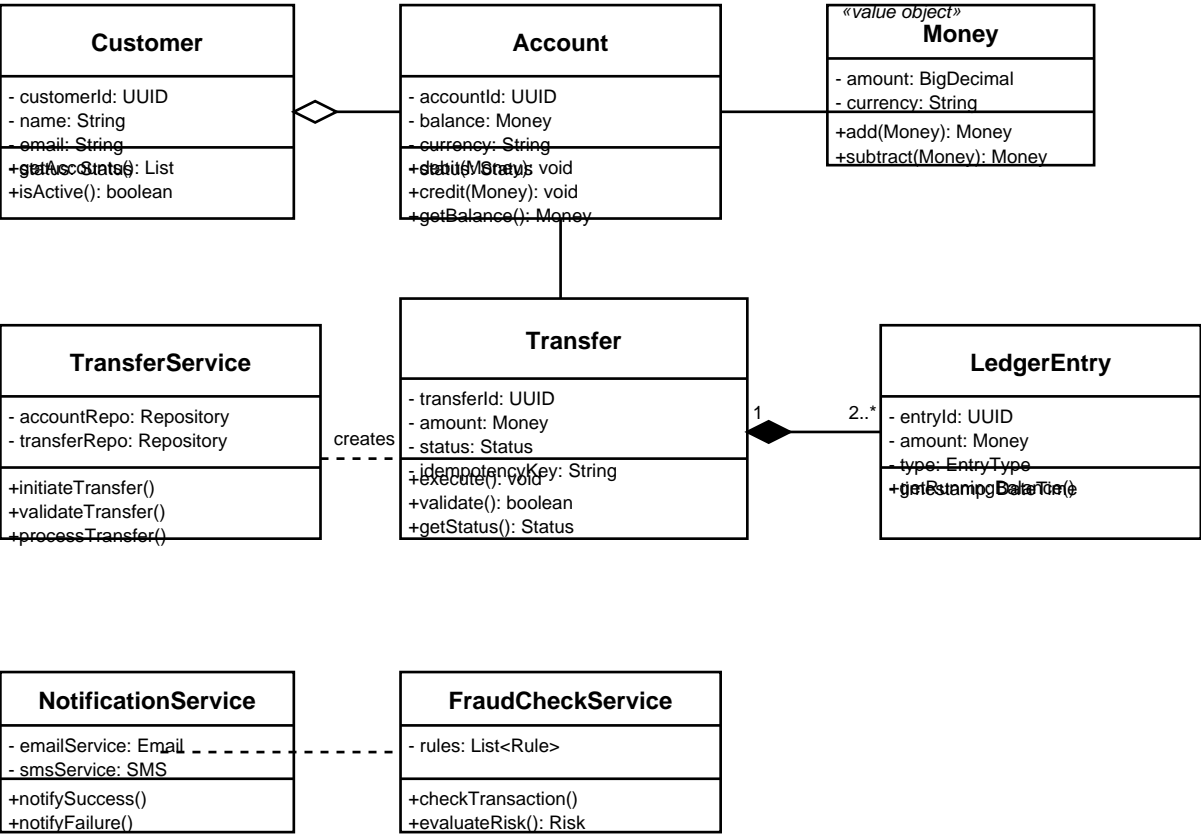
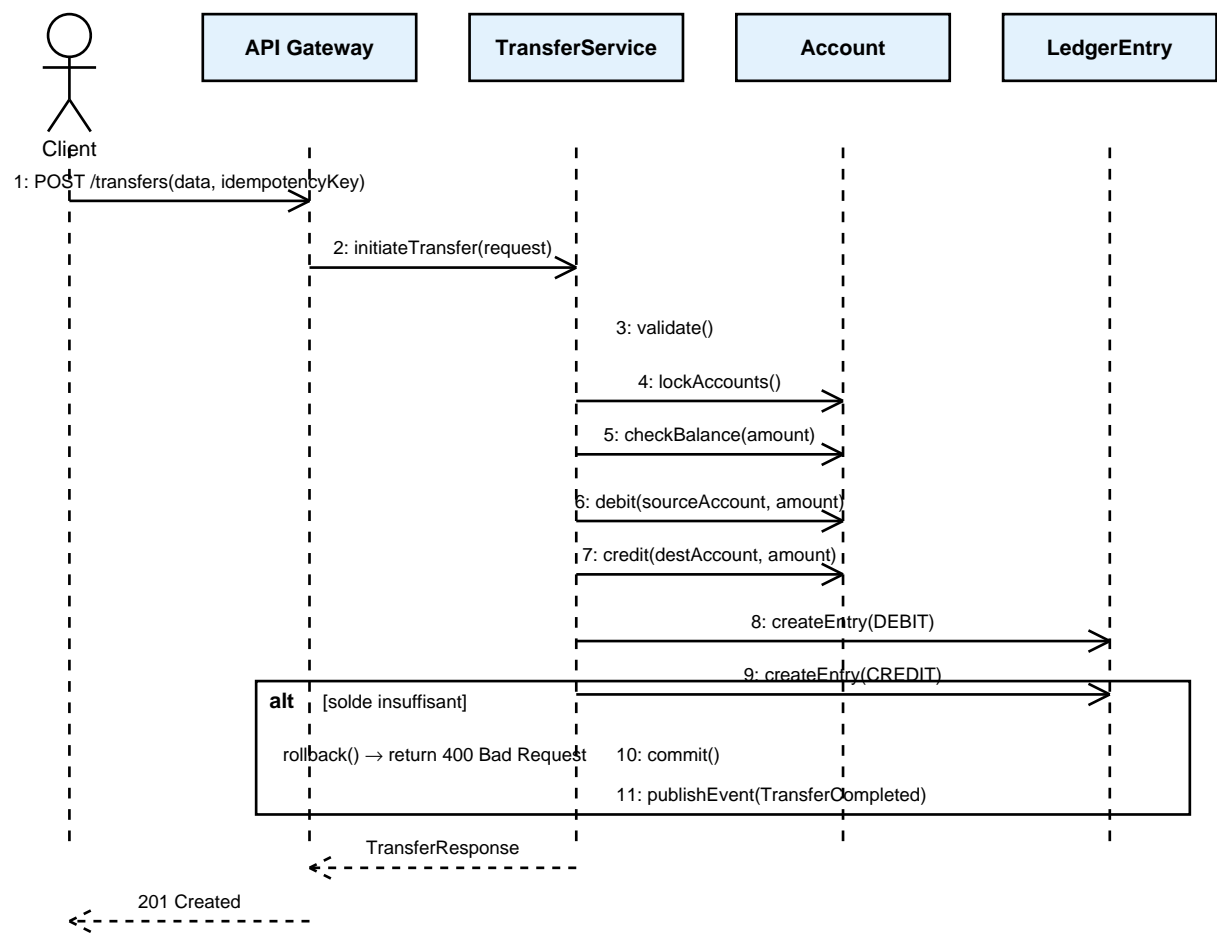


Diagramme de Séquence - Virement Validé



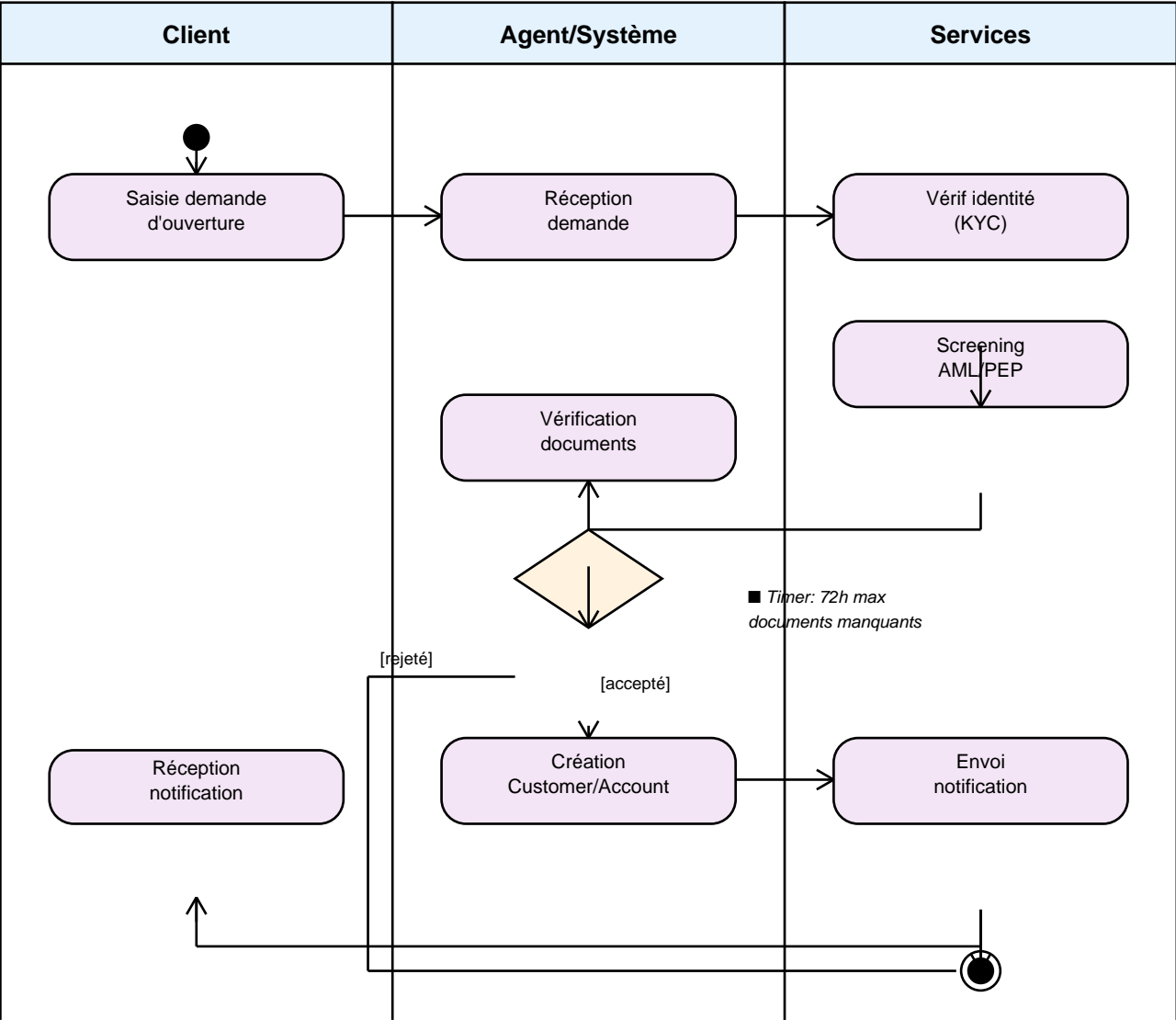
## **Q2. Concevoir 'Ouverture de compte' :**

Diagramme d'activités (workflow) incluant KYC/AML et validation de documents.  
(niveau intermédiaire)

### **Réponse :**

Activités: Saisie demande → Vérif identité (KYC) → Screening AML/PEP →  
Vérif documents → Décision (accept/reject) → Création Customer/Account → Notification.  
Ajouter branches d'erreur, timers (documents manquants),  
et swimlanes (Client, Agent, Services).  
Points attendus: notation UML correcte, relations et multiplicités appropriées

Diagramme d'Activités - Ouverture de Compte (KYC/AML)



### Q3. Concevoir 'Carte bancaire' :

Diagramme d'états pour Card (émission, activée, suspendue, bloquée, expirée).  
(niveau intermédiaire)

#### Réponse :

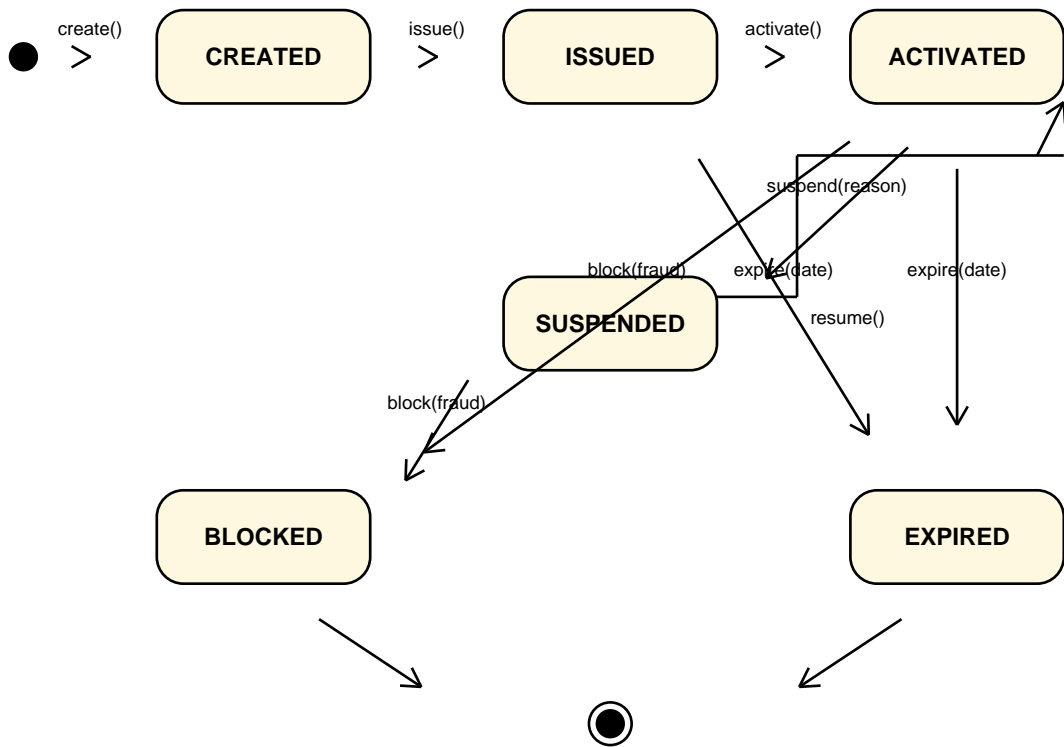
États: CREATED→ISSUED→(ACTIVATED)→(SUSPENDED)→(BLOCKED);  
ISSUED/ACTIVATED→EXPIRED.

Transitions: activate(), suspend(reason), resume(), block(fraud), expire(date).

Garder invariants: BLOCKED terminal (sauf reissue).

Points attendus: notation UML correcte, relations et multiplicités appropriées

Diagramme d'États - Cycle de Vie Carte Bancaire



Note: *BLOCKED* est un état terminal (sauf reissue avec nouvelle carte)

#### **Q4. Concevoir 'Système d'alertes anti-fraude' :**

Diagramme de composants + interfaces, incluant event bus.  
(niveau intermédiaire)

#### **Réponse :**

Composants: API Gateway, Core Banking Service, Transfer Service, Fraud Service, Notification Service, Reporting, Event Bus (Kafka/Rabbit).

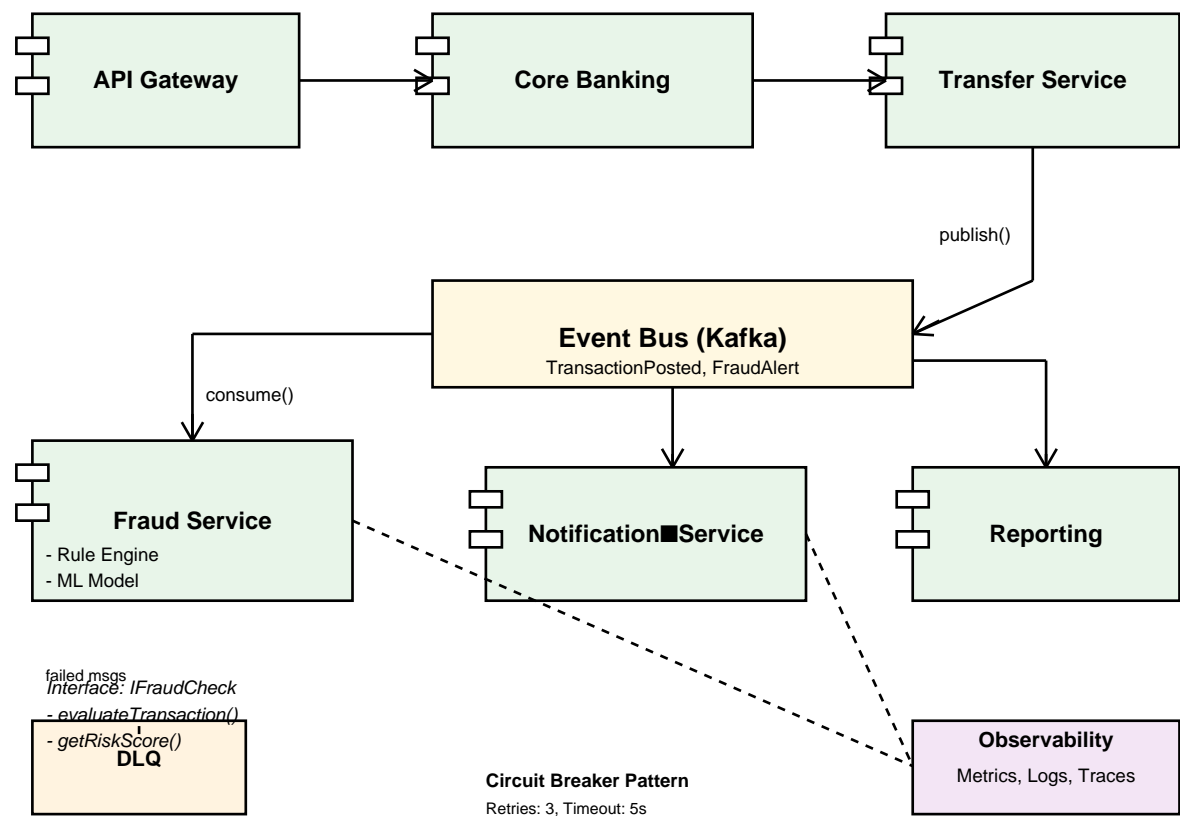
Interfaces: publish(TransactionPosted), consume, rule engine.

TechLead: ajouter observabilité, DLQ, retries, circuit breaker.

Points attendus: notation UML correcte, relations et multiplicités appropriées



Diagramme de Composants - Système d'Alertes Anti-Fraude



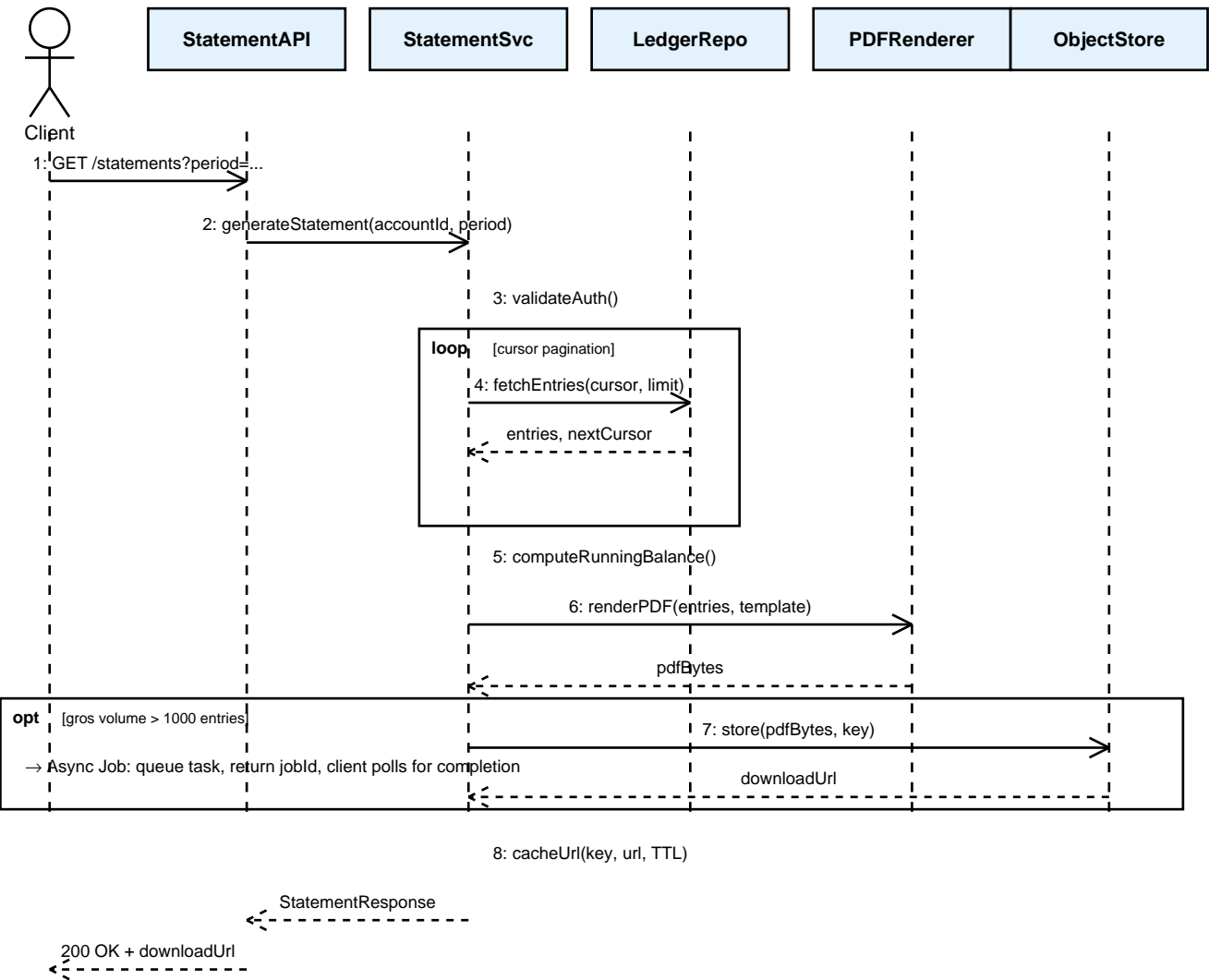
### **Q5. Concevoir 'Relevé de compte' :**

Diagramme de séquence pour la génération (période, pagination, export PDF).  
(niveau intermédiaire)

#### **Réponse :**

Séquence: Client→StatementAPI: request(period) → StatementService: validateAuth,  
fetch ledger entries (cursor pagination), compute running balance,  
generate PDF via Template/Renderer, store in Object Storage, return download link.  
Ajouter caching et async job si gros volume.  
Points attendus: notation UML correcte, relations et multiplicités appropriées

Diagramme de Séquence - Génération de Relevé de Compte



## Questions supplémentaires (Q6-Q25)

Les questions Q6 à Q25 reprennent les mêmes 5 scénarios:

- Virement Interne (diagramme de classes + séquence)
- Ouverture de compte (diagramme d'activités avec KYC/AML)
- Carte bancaire (diagramme d'états)
- Système d'alertes anti-fraude (diagramme de composants)
- Relevé de compte (diagramme de séquence)

*Référez-vous aux diagrammes précédents pour les réponses complètes.*