

# Examen Conception de Systèmes (Banque) – UML à produire – Tech Lead (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 tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q2.** Concevoir 'Ouverture de compte' : diagramme d'activités (workflow) incluant KYC/AML et validation de documents. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q3.** Concevoir 'Carte bancaire' : diagramme d'états pour Card (émission, activée, suspendue, bloquée, expirée). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q4.** Concevoir 'Système d'alertes anti-fraude' : diagramme de composants + interfaces, incluant event bus. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q5.** Concevoir 'Relevé de compte' : diagramme de séquence pour la génération (période, pagination, export PDF). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q6.** Concevoir le système de 'Virement Interne' : produire un diagramme de classes (domain) + un diagramme de séquence pour le scénario 'virement validé'. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q7.** Concevoir 'Ouverture de compte' : diagramme d'activités (workflow) incluant KYC/AML et validation de documents. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q8.** Concevoir 'Carte bancaire' : diagramme d'états pour Card (émise, activée, suspendue, bloquée, expirée). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q9.** Concevoir 'Système d'alertes anti-fraude' : diagramme de composants + interfaces, incluant event bus. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q10.** Concevoir 'Relevé de compte' : diagramme de séquence pour la génération (période, pagination, export PDF). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q11.** Concevoir le système de 'Virement Interne' : produire un diagramme de classes (domain) + un diagramme de séquence pour le scénario 'virement validé'. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q12.** Concevoir 'Ouverture de compte' : diagramme d'activités (workflow) incluant KYC/AML et validation de documents. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q13.** Concevoir 'Carte bancaire' : diagramme d'états pour Card (émise, activée, suspendue, bloquée, expirée). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q14.** Concevoir 'Système d'alertes anti-fraude' : diagramme de composants + interfaces, incluant event bus. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q15.** Concevoir 'Relevé de compte' : diagramme de séquence pour la génération (période, pagination, export PDF). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q16.** Concevoir le système de 'Virement Interne' : produire un diagramme de classes (domain) + un diagramme de séquence pour le scénario 'virement validé'. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q17.** Concevoir 'Ouverture de compte' : diagramme d'activités (workflow) incluant KYC/AML et validation de documents. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q18.** Concevoir 'Carte bancaire' : diagramme d'états pour Card (émission, activée, suspendue, bloquée, expirée). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q19.** Concevoir 'Système d'alertes anti-fraude' : diagramme de composants + interfaces, incluant event bus. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q20.** Concevoir 'Relevé de compte' : diagramme de séquence pour la génération (période, pagination, export PDF). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q21.** Concevoir le système de 'Virement Interne' : produire un diagramme de classes (domain) + un diagramme de séquence pour le scénario 'virement validé'. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q22.** Concevoir 'Ouverture de compte' : diagramme d'activités (workflow) incluant KYC/AML et validation de documents. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q23.** Concevoir 'Carte bancaire' : diagramme d'états pour Card (émission, activée, suspendue, bloquée, expirée). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q24.** Concevoir 'Système d'alertes anti-fraude' : diagramme de composants + interfaces, incluant event bus. (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.

**Q25.** Concevoir 'Relevé de compte' : diagramme de séquence pour la génération (période, pagination, export PDF). (niveau tech lead)

**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: décisions d'architecture, limites, threat model, gouvernance, SLA/SLO.