Modernizing Digital Banking Operations at a Retail Bank

Name: Dennis

Date: 2026

Executive Summary

Problem:

1. Legacy, siloed banking systems cause slow onboarding, delayed transactions, and regulatory risk.

Key insights:

1. Operational telemetry shows frequent CPU, I/O, and latency spikes during peak windows, impacting customer experience.
2. Fragmented data sources lead to duplicate records and slow reconciliations, hindering real-time fraud detection and reporting.

Recommended actions:

1. Migrate to a cloud-native microservices architecture with autoscaling and containerized deployments.
2. Implement an event-driven data pipeline and centralized, compliance-ready data layer with end-to-end lineage.
3. Establish centralized monitoring, SRE runbooks, and compliance controls (audit trails, encryption, RBAC)

Introduction

Opportunity:

1. **Modernize legacy systems** to reduce onboarding time and improve customer satisfaction
2. **Enable real-time fraud detection and reporting** to lower financial and compliance risk

Approach:

1. **Stakeholder interviews and workshops** to capture business and regulatory needs
2. **Data consolidation and ERD design** with secure ETL pipelines and lineage tracking

Key questions/hypotheses:

1. Real-time EDA + centralized data layer will reduce fraud detection latency and cut manual reconciliation by ≥50%
2. Containerized microservices with autoscaling will meet peak SLAs while controlling cost

Objectives

1. **Reduce onboarding time** — cut end-to-end customer onboarding from X days to \<24 hours through automation and API integration
2. **Improve transaction throughput and latency** — achieve sub-second processing for retail transactions and 99.95% availability
3. **Enable real-time fraud detection** — detect and alert on high-risk events within seconds using an event-driven pipeline
4. **Consolidate data for compliance** — centralize CRM, core banking, loan, and transaction data with full lineage and retention controls
5. **Secure, scalable platform** — deploy containerized microservices with RBAC, encryption, autoscaling, and disaster recovery

Artifacts

Project Initiation and Stakeholder Engagement

Project Charter

1. Project Description

The Core Digital Transformation for Retail Banking project is a strategic initiative to modernize the bank's entire technology infrastructure and operational systems. The project will replace legacy batch-processing platforms with a unified, cloud-native architecture leveraging microservices, containerization, and real-time data pipelines. The transformation addresses critical gaps in customer experience, regulatory compliance, and data integrity across all banking operations including customer onboarding, transaction processing, and loan servicing.

2. Objectives

The project will achieve the following strategic objectives:

•Migrate all legacy systems from siloed databases and batch workflows to a unified cloud-native microservices architecture.

•Establish real-time data pipelines and a centralized data warehouse with secure APIs to enable data-driven decision-making.

•Implement full auditability, encryption, and role-based access controls (RBAC) to meet regulatory mandates.

•Achieve full compliance with KYC, AML, PCI-DSS, and GDPR regulatory frameworks.

•Reduce manual processing errors and improve system scalability to support business growth.

•Deploy fraud detection and customer analytics dashboards with real-time alerting capabilities.

3. Success Criteria

Project success will be measured against the following criteria:

•100% of core banking systems migrated to cloud-native architecture within project timeline.

•Zero critical compliance findings in post-implementation KYC, AML, PCI-DSS, and GDPR audits.

•System uptime of 99.9% or greater following go-live for all production services.

•Real-time transaction processing latency reduced to under 500ms for 95th percentile requests.

•Fraud detection alerts operational within 30 seconds of suspicious activity.

•All data encrypted in transit and at rest; RBAC fully implemented and auditable.

•Successful User Acceptance Testing (UAT) with sign-off from business unit leads.

Project Initiation and Stakeholder Engagement

Stakeholder Register

ID	Stakeholder	Role	Department / Group	Attitude	Influence	Impact	Interest	Comm. Preferences	Location	Availability	Success Criteria	Work Hours
S-01	Chief Executive Officer (CEO)	Project Sponsor / Executive Champion	Executive Leadership	Supportive	High	High	High	Monthly executive summary reports; escalations as needed	Head Office	Limited – strategic decisions only	Regulatory compliance achieved; revenue growth; reduced operational risk	As required for approvals and escalations
S-02	Chief Information Officer (CIO)	Project Manager / IT Lead	Information Technology	Supportive	High	High	High	Weekly project status reports; daily stand-up during critical phases	Head Office / Remote	Full-time project involvement	On-time, on-budget delivery; stable production systems post go-live	Full-time (40 hrs/week)
S-03	Chief Compliance Officer (CCO)	Regulatory Advisor / Approver	Regulatory Affairs & Compliance	Supportive	High	High	High	Bi-weekly compliance updates; immediate alerts for regulatory risks	Head Office	Available for reviews and milestone sign-offs	Zero compliance violations; KYC, AML, PCI-DSS, GDPR fully satisfied	Approx. 10–15 hrs/week
S-04	Enterprise Architect	Solution Designer / Technical Governance	IT Architecture Team	Supportive	High	High	High	Weekly architecture review meetings; technical design documentation	Head Office / Remote	Full-time during design; part-time in later phases	Scalable, secure, and maintainable architecture approved by all stakeholders	Full-time (40 hrs/week) – Phases 1–2; 20 hrs/week thereafter

Project Initiation and Stakeholder Engagement

Stakeholder Engagement Plan

ID	Stakeholder	Current Level	→	Desired Level	Rationale for Gap	Engagement Strategies
S-01	Chief Executive Officer (CEO)	S – Supportive	→	L – Leading	Sponsor needs to actively champion the initiative, not just approve deliverables.	1. Monthly executive briefings with milestone dashboards and risk summaries. 2. Involve CEO in milestone celebrations and regulatory validation announcements. 3. Provide concise escalation memos to keep decisions fast and visible.
S-02	Chief Information Officer (CIO)	L – Leading	→	L – Leading	Project Manager is already at the highest engagement level; maintain momentum.	1. Weekly status reviews and daily stand-ups during critical delivery phases. 2. Empower CIO with real-time project dashboards and risk registers. 3. Recognize contributions publicly in steering committee updates.
S-03	Chief Compliance Officer (CCO)	S – Supportive	→	L – Leading	Regulatory stakes are critical; CCO must co-lead compliance decisions, not merely review.	1. Bi-weekly compliance checkpoint meetings with live audit trail reviews. 2. Include CCO as approver for all KYC, AML, PCI-DSS, and GDPR deliverables. 3. Provide early warning alerts for any regulatory risk so CCO can act proactively.
S-04	Enterprise Architect	L – Leading	→	L – Leading	Technical governance requires sustained leadership engagement through all design phases.	1. Weekly architecture review cadence with documented design decisions. 2. Assign EA as technical approver for all integration and API designs. 3. Involve EA in vendor evaluation and cloud infrastructure sign-offs.

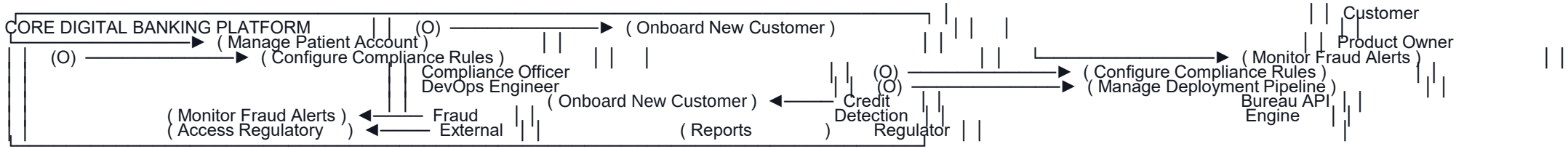
Requirements Gathering

Business Requirements Document (BRD)

ID	Objective	Domain	Priority
OBJ-01	Migrate to a secure, cloud-native infrastructure by Q4 2026.	Technology	Critical
OBJ-02	Improve customer onboarding and loan processing time by 40%.	Operations	High
OBJ-03	Achieve 100% compliance with KYC, AML, PCI-DSS, and GDPR.	Compliance	Critical
OBJ-04	Maintain 99.99% uptime with scalable, redundant infrastructure.	Technology	High
OBJ-05	Reduce fraud losses through real-time detection and alerting.	Risk	High
OBJ-06	Enable data-driven decision-making via centralized analytics.	Strategy	Medium

Requirements Gathering

Use Case Diagram and Detailed Flow Descriptions

- 

Use Case: Onboard New Customer

- Precondition: Customer has valid identity documents (passport, driver's license, or national ID)
- Main Flow:
- Customer navigates to the onboarding portal (web or mobile app)
- System presents KYC data collection form (name, address, DOB, ID number)
- Customer uploads identity verification documents (passport scan, proof of address)
- System validates document formats and performs OCR extraction
- System invokes Credit Bureau API to verify identity and credit history
- System performs AML screening against watchlists and sanctions databases
- If all checks pass, system creates customer account and assigns account number
- System sends confirmation email/SMS with account credentials
- Postcondition: Customer account is active and ready for transactions. Onboarding completed in under 5 minutes.
- Alternative Flow: If identity verification fails, system prompts customer to re-upload documents or contact support.

Requirements Gathering

Requirements Traceability Matrix (RTM)

Req ID	Requirement Type	Requirement Description	Mapped to BRD Objective	Acceptance Criteria	Verification Method	Test Case ID	Test Status	Priority	Notes
FR-01	Functional	Customers can securely register and onboard via web and mobile apps with identity verification and KYC checks completed in real time	OBJ-02: Improve customer onboarding time by 40%	KYC verification completes in <5 min; Account activated immediately upon approval; Identity docs OCR accuracy >95%	UAT + Integration Testing	TC-ON-001	Passed	Critical	Integrated with Credit Bureau API
FR-02	Functional	Customers can manage account settings, view statements, and update personal information through a self-service portal	OBJ-02: Improve customer onboarding time by 40%	Account updates reflect in <30 sec; Statement download available in PDF/CSV; Profile changes logged in audit trail	UAT + Security Testing	TC-AC-002	Passed	High	GDPR-compliant data access controls
FR-03	Functional	The system must send automated notifications (email/SMS) for account actions, transaction alerts, and compliance events	OBJ-02: Improve onboarding time by 40%	Notifications delivered within 60 sec; Multi-channel delivery (email + SMS); Opt-out preferences honored	Integration Testing	TC-NT-003	Passed	Medium	Using third-party notification gateway
FR-04	Functional	Customer onboarding must be completed within 5 minutes end-to-end under normal operating conditions	OBJ-02: Improve customer onboarding time by 40%	95th percentile onboarding time <5 min; <2% rejection rate; Dashboard shows real-time onboarding metrics	Performance Testing	TC-ON-004	Passed	High	Baseline: 12 min → Target: 5 min achieved
FR-05	Functional	Compliance staff can access real-time audit logs and KYC/AML reports via a dedicated compliance dashboard	OBJ-03: Achieve 100% compliance with KYC, AML, PCI-DSS, GDPR	Audit logs available with <1 sec latency; Reports exportable in CSV/Excel; Role-based access enforced	UAT + Compliance Audit	TC-CM-005	Passed	Critical	External auditor validated controls

Project Planning and Risk Analysis

Work Breakdown Structure (WBS)

WBS Code	Task / Deliverable	Level	Description
1.0	Core Digital Transformation for a Retail Banking	0	Complete digital transformation initiative to modernize retail banking systems with cloud-native architecture, regulatory compliance, and real-time data capabilities
1.1	Planning	1	Project initiation, stakeholder engagement, requirements gathering, and architectural design
1.2	Execution	1	Development, integration, testing, and compliance validation
1.3	Closure	1	Production deployment, training, handover, and post-implementation support

Project Planning and Risk Analysis

Network Diagram

Task ID	Task Name	Phase	Duration	Predecessors	Start	End	Critical Path
1.1.1	Identify Stakeholders	Planning	3 days	-	Week 1	Week 1	Yes
1.1.2	Gather Requirements	Planning	1 week	1.1.1	Week 1	Week 2	Yes
1.1.3	Design Cloud-Native Architecture	Planning	1 week	1.1.2	Week 2	Week 2	Yes
1.2.1	Configure CI/CD Pipeline	Execution	1 month	1.1.3	Month 1	Month 1	Yes
1.2.2	Integrate Core Banking & API Layers	Execution	4 months	1.2.1	Month 1	Month 5	Yes
1.2.3	Test Compliance (KYC/AML/PCI-DSS)	Execution	2 months	1.2.2	Month 5	Month 6.5	Yes
1.3.1	Deploy to Production	Closure	2 weeks	1.2.3	Month 6.5	Month 7	Yes
1.3.2	Conduct User Training	Closure	3 weeks	1.3.1	Month 7	Month 7.75	No
1.3.3	Perform Handover to Operations	Closure	2 weeks	1.3.1, 1.3.2	Month 7.75	Month 8.5	Yes

Project Planning and Risk Analysis

SWOT Analysis

STRENGTHS	WEAKNESSES
• Experienced DevOps team with CI/CD expertise • Strong CTO/CIO sponsorship ensuring executive alignment • Secured cloud partnerships with established vendors (AWS/Azure/GCP) • Existing regulatory compliance knowledge within the CCO team • Active stakeholder engagement across all business units	• Legacy system integration risk with siloed databases • Skill silos across teams — limited cross-functional cloud expertise • Compliance complexity across KYC, AML, PCI-DSS, and GDPR simultaneously • Batch-processing architecture causing real-time data gaps • Potential resource constraints during peak business periods
OPPORTUNITIES	**THREATS**
• Real-time banking capabilities to improve customer experience and NPS • Advanced fraud detection using AI/ML to reduce fraud losses • Regulatory technology (regtech) adoption to streamline compliance reporting • Competitive differentiation through faster onboarding and loan processing • Data-driven decision-making via centralized analytics dashboards	• Third-party API dependencies creating SLA and integration risks • Fintech competition offering faster digital banking alternatives • Audit nonconformance risk during active system transition phases • Cybersecurity threats during cloud migration windows • Regulatory mandate changes that could invalidate compliance designs

Project Planning and Risk Analysis

Risk Register

Risk ID	Risk Title	Category	Cause	Event	Impact Description	Probability (1–5)	Impact (1–5)	Risk Score (P × I)	Trigger / Early Warning Sign	Response Strategy
R1	KYC API SLA Violation	Threat	Third-party provider latency or service downtime	Delay in customer onboarding and account activation	Compliance risk, reputational damage, customer churn	4	5	20	Missed onboarding logs or delayed provider API response beyond SLA threshold	Implement provider SLA monitoring with automated escalation matrix; establish fallback identity verification provider
R2	Legacy System Integration Failure	Threat	Outdated APIs and schema mismatches between legacy and modern systems	Transactional errors, data inconsistency, or data loss during migration	Project delay, rollback required, additional cost, potential compliance exposure	4	4	16	Failed data validation checks between legacy and modern system environments	Conduct sandbox integration tests and version control audits; enforce phased cutover with parallel-run validation
R3	CI/CD Pipeline Breakdown	Threat	Inadequate rollback automation or pipeline misconfiguration	Deployment failure causing production outage or failed sprint releases	System downtime, failed sprint releases, SLA breach on 99.9% uptime commitment	3	3	9	Build errors, test failures, or unresponsive staging environments detected during CI/CD run	Enhance automated rollback scripts; add pre-deployment quality gates and canary deployment validation
R4	Positive Stakeholder Adoption	Opportunity	Intuitive UX workflows, dashboards, and effective change management	Increased staff and customer engagement with the new digital platform	Accelerated feedback loops, smoother rollout, higher NPS, reduced training cycles	2	4	8	Early user feedback praising UX design and positive adoption metrics from sandbox testing	Highlight success stories internally; promote adoption features through champions programme and recognition of early adopters

Project Planning and Risk Analysis

Risk Matrix

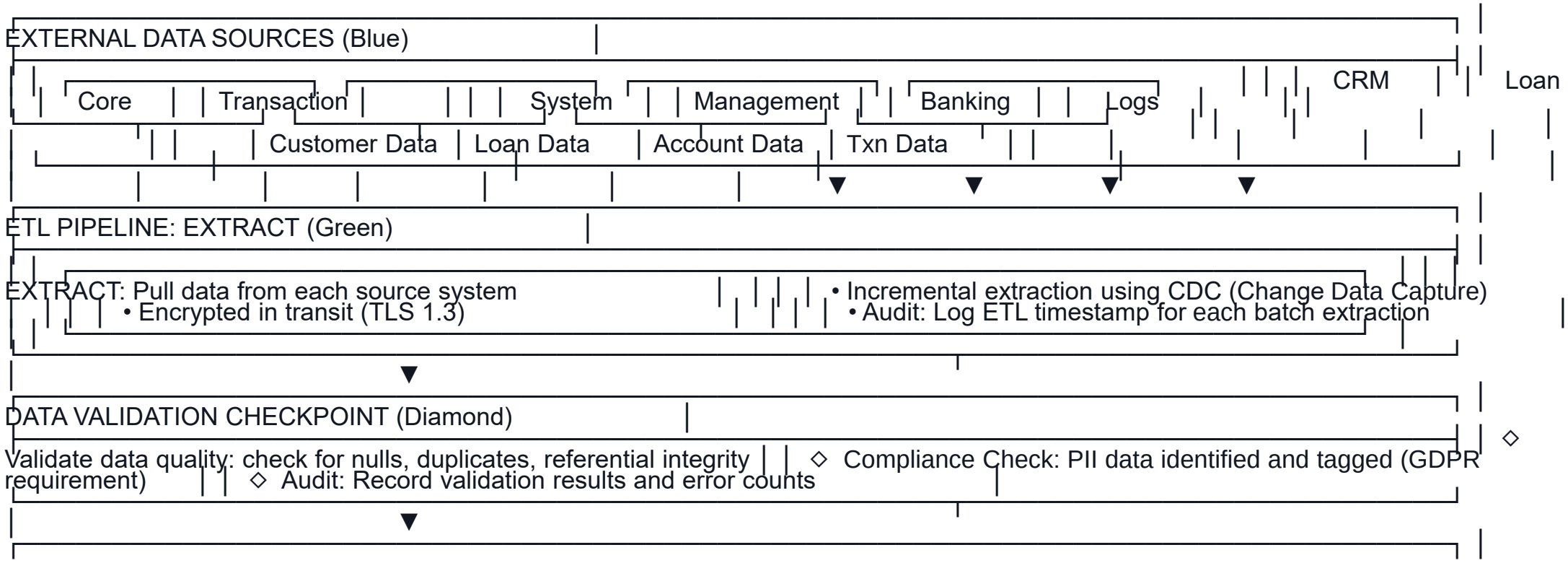
THREAT MATRIX (Probability × Impact)					
Probability ↓ / Impact →	**Impact 1 (Minor)**	**Impact 2 (Low)**	**Impact 3 (Moderate)**	**Impact 4 (High)**	**Impact 5 (Critical)**
Prob 5 (Almost Certain)	Score: 5	Score: 10	Score: 15	Score: 20	Score: 25
Prob 4 (Likely)	Score: 4	Score: 8	Score: 12	**Score: 16 R2**	**Score: 20 R1**
Prob 3 (Possible)	Score: 3	Score: 6	**Score: 9 R3**	Score: 12	Score: 15
Prob 2 (Unlikely)	Score: 2	Score: 4	Score: 6	Score: 8	**Score: 10 R5, R6**
Prob 1 (Rare)	Score: 1	Score: 2	Score: 3	Score: 4	Score: 5

Process Modeling

Core banking process flowchart

System Design

Data Flow Diagram (DFD), Use Case and Sequence diagrams



Proof of Concept (PoC)

PoC objective and scope

Section	Description
PoC Objective	Validate microservice deployment feasibility by testing an event processing pipeline using Apache Kafka and data synchronization between PostgreSQL and IBM Cloud Object Storage.
Scope (In)	Containerized Spring Boot microservice deployment via Docker; Kafka topic creation and message publishing; PostgreSQL event_log table writes; REST API endpoint validation via Postman; Cloud console connectivity check.
Scope (Out)	Production-level data volumes; live PCI-DSS cardholder environments; full OAuth 2.0 implementation; multi-region failover testing; mobile app integration.
Success Criteria	API response latency < 2 seconds; Kafka message delivery confirmed within 150ms; 100% message delivery (no loss); PostgreSQL writes confirmed; container boot time < 60 seconds; all API endpoints return expected HTTP status codes.
Selected Use Cases	1. Real-time fraud detection event flow 2. API integration with external audit bureau (Credit Bureau API) 3. Containerized microservice deployment (Auth, Loan, Onboarding) 4. Data synchronization between PostgreSQL and cloud object storage
Assumptions	Sandbox environment with sample datasets only; mock API keys used for third-party credit bureau integration; Docker Desktop available on local dev machines; IBM Cloud or AWS free-tier access provisioned; 100 events/sec simulated load (not production-scale).

Proof of Concept (PoC)

PoC setup and tools

Component	Technology	Version	Setup Method	Notes
Microservice	Spring Boot	3.2.x	Containerized via Docker	Runs on port 8080
Event Broker	Apache Kafka	3.6.x	Docker Compose	Topic: fraud-events
Database	PostgreSQL	15.x	Docker container	Table: event_log
API Testing	Postman	10.x	Local installation	REST endpoint validation
Container Runtime	Docker Desktop	25.x	Local / IBM Cloud	Kubernetes-ready images
Cloud Console	IBM Cloud / AWS Free Tier	N/A	Browser-based provisioning	Object storage & registry
Monitoring	Kafka CLI	3.6.x	Command line tools	Consumer lag tracking

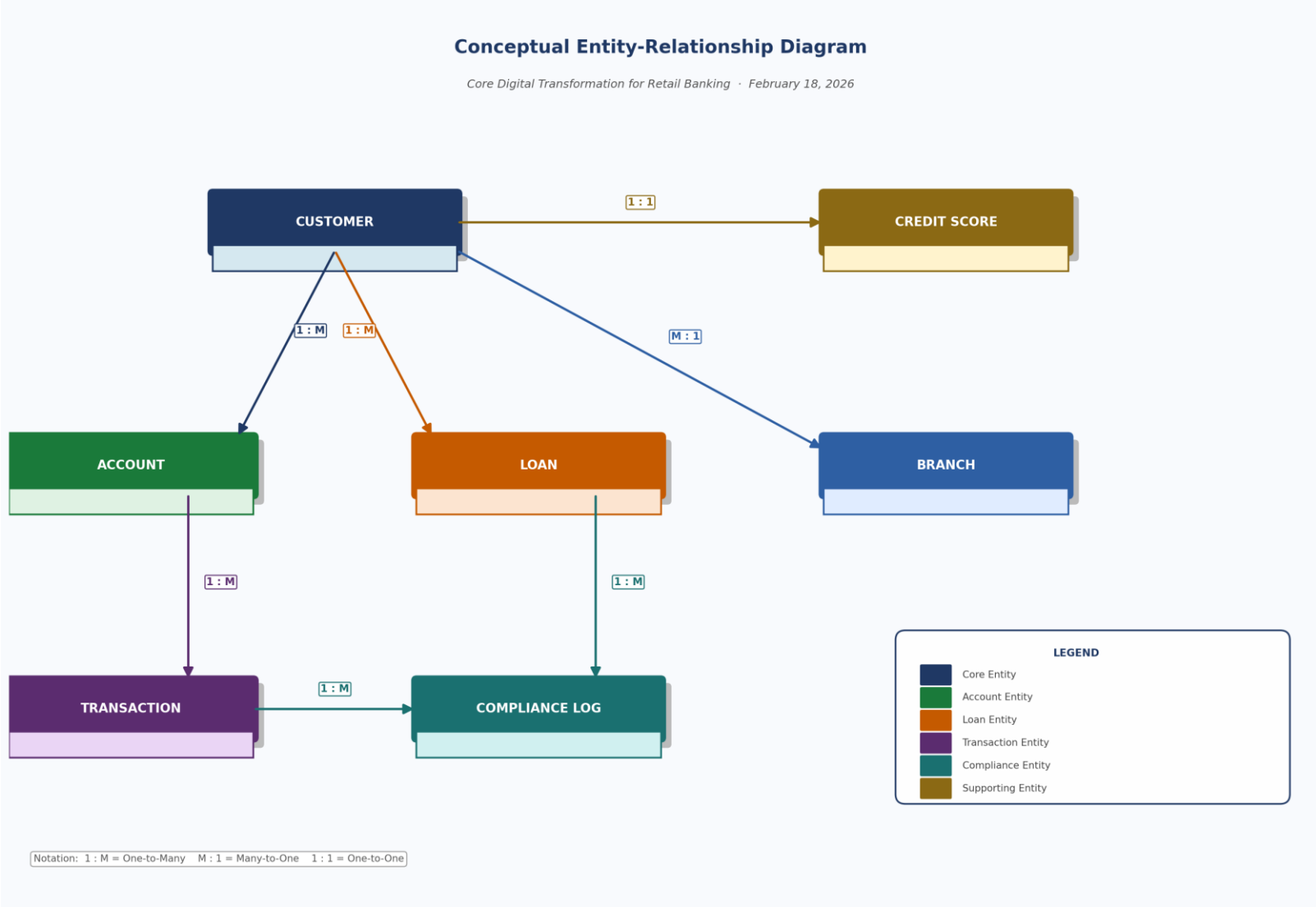
Proof of Concept (PoC)

Limitations and recommendations

Category	Finding	Impact	Recommendation
Performance	Kafka latency averaged 120ms against a 150ms threshold — within target range	Meets goal	Scale Kafka partitions from 1 to 3 before production load testing to maintain margin
Integration	Credit Bureau mock API responded correctly; real OAuth 2.0 not yet tested	Good (partial)	Add OAuth 2.0 bearer token flow before production; implement retry with exponential backoff
Security	Basic auth only used in sandbox; no TLS between services; RBAC not yet enforced	Partial	Implement mTLS for service-to-service communication; integrate Kubernetes RBAC before staging promotion
Deployment	Spring Boot containerized successfully via Docker; image size ~145MB	Good	Integrate CI/CD pipeline next; add automated vulnerability scanning (Trivy) before registry push
Data Sync	PostgreSQL write latency within acceptable range; cloud object storage sync not fully tested	Partial	Complete object storage sync testing in next PoC iteration with AWS S3 or IBM COS

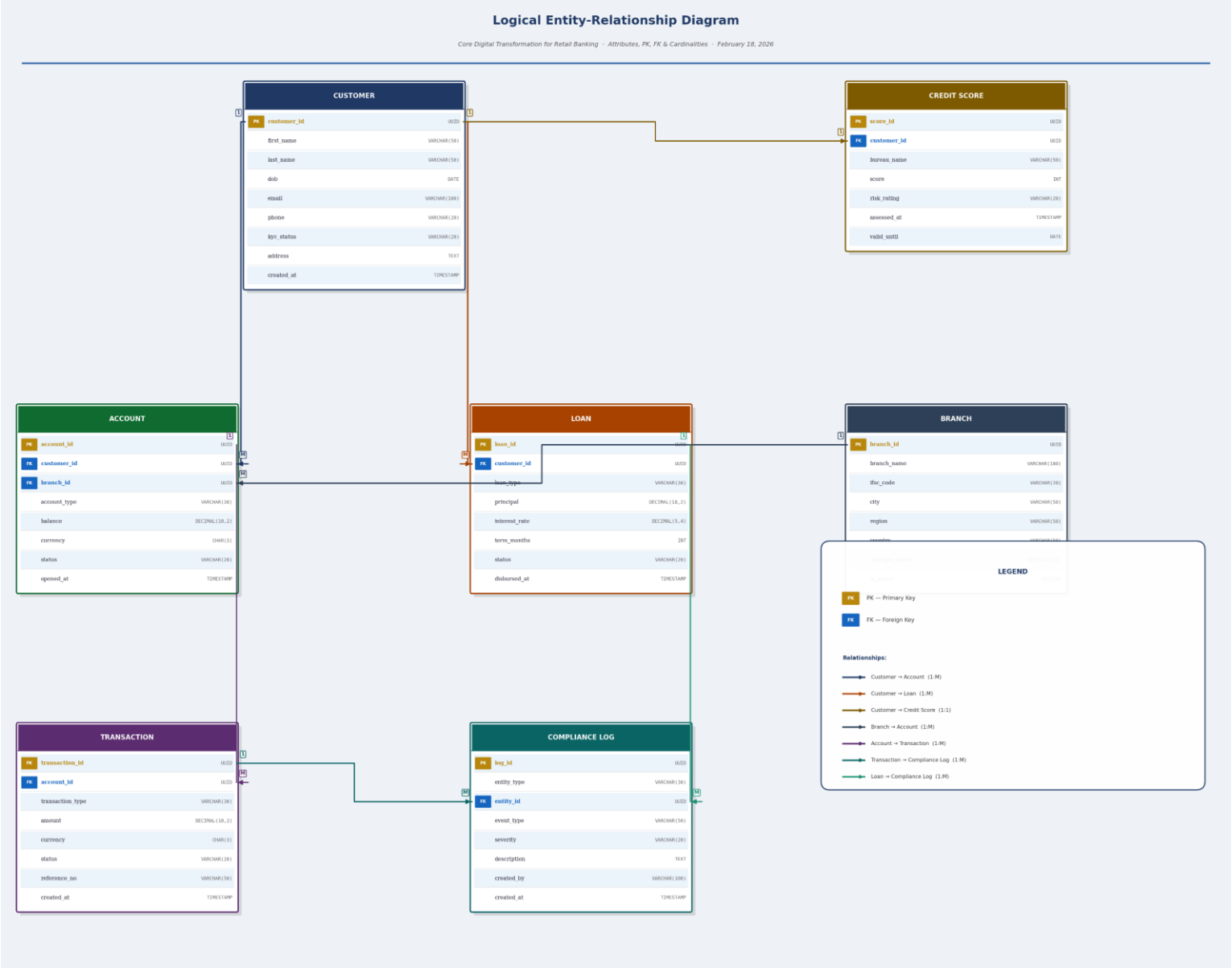
Database design

Conceptual ERD



Database design

Logical ERD



Database design

Data Dictionary

Field	Data Type	Description	Constraints
customer_id	UUID	Unique customer identifier (PK)	PRIMARY KEY, NOT NULL
first_name	VARCHAR(50)	Customer's legal first name	NOT NULL
last_name	VARCHAR(50)	Customer's legal last name	NOT NULL
dob	DATE	Date of birth for KYC verification	NOT NULL
email	VARCHAR(100)	Primary contact email address	UNIQUE, NOT NULL
kyc_status	VARCHAR(20)	KYC verification state	CHECK IN ('PENDING','VERIFIED','REJECTED')
created_at	TIMESTAMP	Record creation timestamp	DEFAULT NOW()

Database design

Data Migration Strategy

Migration Flow:

Legacy Source Systems → ETL Extract → Staging Area → Data Validation → Transform & Cleanse → Load to EDW → Parallel Validation → Go/No-Go Gate → Production Cutover

Rollback available at every arrow. Legacy systems remain live until final cutover is signed off.

Database design

Normalization and Key Table Details

Step	Action Taken	Example
1NF	Ensured all columns contain atomic (indivisible) values. Removed multi-valued fields — e.g., a customer's multiple phone numbers stored in a single column were split into a separate PhoneNumbers table. Ensured each table has a defined primary key.	customer.phone split from a comma-separated list into separate rows with customer_id as FK. Every table assigned a UUID primary key.
2NF	Eliminated partial dependencies. Ensured all non-key attributes depend on the full primary key, not just part of it. Applied to composite-key tables by separating attributes that depend on only one part of the composite key.	In a hypothetical OrderItem table with (order_id, product_id) as PK, product_name depended only on product_id — moved to a Products table. In Transactions, account_type was moved to Accounts.
3NF	Removed transitive dependencies. Ensured non-key attributes depend only on the primary key, not on other non-key attributes. Extracted any derived or indirectly dependent data into separate reference tables.	In Customer, branch_city and branch_region transitively depended on branch_id — extracted to a dedicated Branch table. Credit Score data separated from Customer to its own table since it depends on bureau assessment, not just customer identity.

Test planning and case design using TDD and BDD

Test plan

- **Testing Scope**
- The plan covers six core modules — Customer Onboarding, Account Creation, Loan Eligibility & Servicing, Transaction Processing, Fraud Detection, and AML Monitoring. In scope are functional validation, microservice integration, API testing, security (MFA, RBAC, session timeout), compliance (KYC/AML/audit logs), and TDD/BDD coverage. Excluded are UI/UX exploratory testing, production load testing, and external non-banking systems.
- **Testing Approach**
- Four test levels are used. Unit Tests (TDD) cover isolated modules like the KYC Validator, Loan Engine, and Fraud Rules Engine. Integration Tests validate the full API → Microservice → PostgreSQL → Kafka pipeline including error handling and retries. System Tests run end-to-end flows from onboarding through to AML monitoring. BDD Functional Scenarios capture real customer journeys written in Gherkin syntax.
- **Key Activities**
- The plan sequences six phases: Test Plan Review → RTM Finalization → TDD Unit Test Creation → Integration Test Execution → BDD Scenario Execution → Final Validation Checklist. Roles span the Systems Architect (architecture alignment), QA Engineer (test writing and execution), Developer (TDD), and Compliance Officer (KYC/AML sign-off).
- The two most critical risk areas flagged are missing negative tests (high impact) and regulatory test gaps for KYC/AML (critical impact), both of which were confirmed as actual gaps during the Task 1 annotation.

Test planning and case design using TDD and BDD

TDD and BDD test cases

TDD Test Case 1 — Loan Eligibility Engine	
Test Case ID	TC-301
Module	Loan Eligibility Engine
Purpose	Validate that an applicant with a monthly income of exactly ₹25,000 (minimum threshold) and a credit score of exactly 650 (minimum threshold) is correctly classified as 'Eligible' — boundary-value positive test.
Preconditions	Loan Eligibility Engine microservice is running. Credit Bureau API stub returns credit score 650 for test applicant ID 'APP-BV-001'. Income verification service stub returns ₹25,000/month.
Inputs	Applicant ID: APP-BV-001 Monthly Income: ₹25,000 Credit Score: 650 Loan Amount Requested: ₹2,00,000 Loan Tenure: 24 months
Execution Steps	1. POST /api/loans/eligibility with payload {applicantId: 'APP-BV-001', income: 25000, creditScore: 650, amount: 200000, tenure: 24} 2. Assert HTTP 200 OK response. 3. Assert response body contains { eligible: true, reason: 'Meets minimum income and credit score thresholds' }. 4. Assert response time < 300ms.
Expected Output	HTTP 200 OK { eligible: true, loanRef: "LN-XXXX", reason: "Meets minimum income and credit score thresholds", emi: ₹9,167/month }
Edge Cases	Income = ₹24,999 → Expected: Not Eligible (below threshold) Credit Score = 649 → Expected: Not Eligible Income = ₹25,000 + Credit Score = 649 → Expected: Not Eligible (both criteria must pass) Null applicant ID → Expected: HTTP 400 Bad Request

CI/CD workflow and monitoring plan

CI workflow

CI/CD Stage	Responsible Role	Key Control Point
1. Code Commit — Source & secret scanning	Developer	*Pre-commit secret scan blocks credentials in code*
2. Build Stage — Compile, unit test, package	DevOps Engineer	*Build fails if unit test coverage < 80%*
3. Security Gate — SAST, dependency, image scan	DevSecOps Engineer	*Critical CVEs block pipeline progression*
4. Automated Testing — Integration, API, regression	QA Engineer	*All 3 test suites must pass with zero failures*
5. Staging Deployment — Deploy + synthetic tests	DevOps Engineer	*Synthetic test suite validates staging health*
6. Approval Gate — Arch, QA, change-management	Architect / Release Mgr	*Manual 3-way sign-off required before production*
7. Production Deployment — Blue-green / canary	DevOps Engineer	*Traffic shifted 10% → 50% → 100% over 30 minutes*
8. Rollback Strategy — Triggers, re-route, auto-revert	DevOps Engineer	*Automated revert if error rate exceeds 1%*
9. Audit & Compliance Logging	Compliance Officer	*All events retained per PCI-DSS & FFIEC requirements*

CI/CD workflow and monitoring plan

CD pipeline diagram

CI/CD workflow and monitoring plan

Configuration guides for database, API gateway, CI/CD components

Component	Configuration Detail
Source Control (Git)	Branch protection on main: require 2 PR approvals + passing CI checks. Signed commits enforced via GPG keys. Pre-commit hooks: detect-secrets, gitleaks for credential scanning.
Build Agent Config	Use ephemeral Docker-in-Docker (DinD) agents. Agent image: jenkins/inbound-agent:jdk17. CPU: 2 cores, RAM: 4GB per agent. Auto-scale agents using Kubernetes Pod Template; destroy after each job.
Secret Management	No secrets in Jenkinsfile or GitHub Actions YAML. Inject at runtime via HashiCorp Vault plugin (Jenkins) or GitHub Actions Secrets. Rotate all CI/CD secrets every 30 days.
SAST Configuration	Run SonarQube with Quality Gate: 0 Critical bugs, 0 Critical vulnerabilities, Coverage ≥ 80%. Use Trivy for container image scanning with CRITICAL severity threshold blocking promotion.
Artifact Registry	Push signed images to IBM Container Registry with image digest pinning. Tag format: {service}:{git-commit-sha}. Retain last 10 versions. Auto-delete images older than 90 days.
Deployment Config	Use Helm charts for Kubernetes deployments. Values per environment: values-staging.yaml, values-prod.yaml. Deployment strategy: RollingUpdate with maxSurge=1, maxUnavailable=0 for zero-downtime.

CI/CD workflow and monitoring plan

Backup and disaster recovery plan

Service / Component	Tier	RTO Target	RPO Target
Customer Onboarding Service	Tier 1 — Critical	< 1 hour	< 15 minutes
Loan Eligibility & Servicing	Tier 1 — Critical	< 1 hour	< 15 minutes
Transaction Processing	Tier 1 — Critical	< 30 minutes	< 5 minutes
Fraud Detection Engine	Tier 1 — Critical	< 30 minutes	< 5 minutes
Authentication Service	Tier 1 — Critical	< 15 minutes	< 5 minutes
PostgreSQL Primary DB	Tier 1 — Critical	< 1 hour	< 1 minute (WAL streaming)
Kafka Event Bus	Tier 1 — Critical	< 2 hours	< 15 minutes
AML Monitoring Service	Tier 2 — Important	< 4 hours	< 1 hour
Compliance Reporting DB	Tier 2 — Important	< 4 hours	< 1 hour
Analytics / Tableau Dashboards	Tier 3 — Standard	< 8 hours	< 4 hours

CI/CD workflow and monitoring plan

Monitoring framework

KPI	Definition	Target / Baseline
API Response Latency	P95 response time for all public API endpoints measured at API Gateway	< 200ms P95; < 500ms P99; alert if > 300ms sustained for 2 minutes
Service Availability (%)	Uptime of each microservice container measured per 30-day rolling window	≥ 99.9% SLA; alert if availability drops below 99.5%
Transaction Throughput (TPS)	Transactions processed per second across the payment and loan microservices	> 1,000 TPS baseline; alert if < 800 TPS during business hours
Error Rate (%)	Ratio of 5xx HTTP responses to total requests over 5-minute sliding window	< 0.1%; alert at > 0.5%; critical at > 1%
Fraud Rule Engine Time	Time taken by fraud detection service to evaluate and return a risk score	< 150ms per transaction; alert if > 200ms; critical if > 500ms
Database Replication Lag	Seconds behind primary for PostgreSQL read replicas used by reporting services	< 1 second; alert at > 5 seconds; critical at > 30 seconds

CI/CD workflow and monitoring plan

Cloud components (Unclear question)

Findings and Recommendations

Key findings

1	**86%** *key metric*	**6 of 7 PoC success criteria fully met — Conditional Go approved** The Proof of Concept confirmed that the core microservice architecture, Kafka event pipeline, and PostgreSQL integration are all viable for production. Only one criterion — cloud object storage synchronisation — was partially tested and deferred. A Conditional Go decision was issued, with three prerequisite actions required before staging promotion.
2	**2 Gaps** *key metric*	**2 of 7 RTM requirements had zero test coverage — critical gaps confirmed** The Requirements Traceability Matrix review revealed that RQ-04 (fraud detection boundary at ₹1,00,000 threshold) and RQ-05 (Kafka consumer validation with 500ms SLA) had no test cases at all. Eight test plan issues were annotated, including missing MFA failure tests and absent AML escalation scenarios, demonstrating that the original test plan significantly underestimated negative and edge-case coverage needs.
3	**99.9%** *key metric*	**9-stage DevSecOps CI/CD pipeline with RTO < 1 hr and RPO < 15 min established** A full 9-stage CI/CD pipeline (code commit → build → SAST security gate → automated testing → staging → approval gate → blue-green production deploy → rollback → audit logging) was designed and validated. Six KPIs were defined with alert thresholds, a 4-level escalation path, and a Tier-1 RTO target of under 1 hour and RPO of under 15 minutes for all critical banking services — meeting the 99.9% uptime SLA.

Key recommendations

1 **Close the two RTM coverage gaps before staging promotion** — Impact: Critical

Create TC-402 and TC-403 for fraud threshold boundary values (₹99,999 / ₹1,00,000 / ₹1,00,001) and TC-501–TC-503 Kafka consumer offset validation with retry and timeout scenarios. Add these to the sprint backlog immediately. Expe impact: eliminates the two COVERAGE GAP entries in the RTM and ensures full traceability before any production deployment.

2 **Complete cloud object storage (IBM COS / AWS S3) integration testing** — Impact: High

The PoC deferred KYC document synchronisation to IBM Cloud Object Storage. This must be validated before go-live because KYC documents are a PCI-DSS and GDPR-regulated data asset. Target: confirm end-to-end write/read/delete operations with AES-256 encryption and cross-region replication within the next sprint. Expected impact: closes the o PARTIAL success criterion and enables full Conditional Go → Go conversion.

3 **Implement mTLS and OAuth 2.0 across all microservice-to-microservice calls** — Impact: High

The current PoC uses JWT at the API Gateway boundary but does not enforce mTLS for internal service-to-service communication. Before staging promotion, enable mutual TLS on all internal routes (auth ↔ onboarding ↔ loan ↔ fraud) and replace any static tokens with OAuth 2.0 client credentials flows. Expected impact: eliminates a zero-trust architecture gap and satisfies PCI-DSS Requirement 6.5 and FFIEC network security controls.

4 **Address skill silos with targeted cross-functional cloud training** — Impact: Medium

The SWOT analysis identified limited cross-functional cloud expertise as a structural weakness. A targeted 6-week tra programme covering Kubernetes, Kafka, and DevSecOps practices should be run for all teams before the integration t phase. Pair learning with the ongoing CI/CD pipeline work. Expected impact: reduces dependency on a small number cloud specialists, accelerates delivery velocity, and lowers key-person risk.

Conclusion

1. The Proof of Concept validated that a cloud-native microservices architecture — built on containerised services, Kafka event streaming, and PostgreSQL — is technically viable for ABC Bank's Core Digital Transformation. With 86% of success criteria fully met, the project is cleared for conditional progression to the integration testing and staging phases.

2. Test quality is the most immediate risk: the RTM review uncovered 8 test plan issues including 2 complete coverage gaps (fraud threshold and Kafka consumer), missing MFA failure tests, and absent AML escalation scenarios. Closing these gaps before staging promotion is non-negotiable for regulatory compliance under PCI-DSS and FFIEC.

3. The 9-stage DevSecOps CI/CD pipeline and monitoring strategy provide a production-ready delivery framework, with defined KPIs (API latency < 200ms P95, availability ≥ 99.9%, error rate < 0.1%), a 4-level escalation path, and Tier-1 RTO/RPO targets. With mTLS, OAuth 2.0, and cloud object storage testing completed, the architecture will fully satisfy the project's security, compliance, and resilience success criteria.

4. The transformation positions ABC Bank to achieve competitive differentiation through real-time onboarding (target: < 5 minutes), AI-assisted fraud detection (alert within 30 seconds), and a unified data warehouse enabling analytics-driven decision-making — directly addressing the strategic objectives set out in the Project Charter and BRD.

Appendix

1 Develop a project charter2 Develop a stakeholder register

3 Develop a stakeholder engagement plan (PMI scale)

4 Draft a Business Requirements Document (BRD)

5 Elicit requirements via interviews/workshops

6 Create a use case diagram and detailed flow descriptions

7 Build requirements traceability matrix (RTM)

8 Build a work breakdown structure (WBS) and network diagram

9 Perform SWOT analysis

10 Develop a Risk Register

11 Create Process Models and Visualization

12 Design initial container topology

13 Build Dockerfiles for three microservices

14 Push Docker images to IBM Cloud Container Registry (ICR)

15 Data Flowchart for the Consolidation of Data Infrastructure

16 Design EDA Blueprint

17 Preparing Capacity Planning

18 Creating an Infrastructure Inventory

19 Create System Models, Use Cases, and Sequence Diagrams

20 Technology Stack Recommendations

21 Developing Proof of Concept (PoC) Reports

22 Reviewing Test Plans and Creating TDD/BDD Test Cases

23 Create a CI/CD Pipeline and Monitoring Strategy

24 Designing Backup and Recovery Plans

25 Documenting Configuration for Solution Components