

NEXUSBANK

Core Banking System

Scenarios & Use Cases

Document Version: 1.0

Last Updated: February 2026

Author: Technical Lead

Table of Contents

1. System Overview
2. The Bank We're Building
3. User Roles & Personas
4. Day-in-the-Life Scenarios
5. Layer-by-Layer Scenarios
6. External System Simulations
7. ML/AI Scenarios
8. End-to-End User Journeys

1. System Overview

What Are We Building?

We are building **NexusBank** - a modern, full-featured Core Banking System that simulates a real-world retail bank. This system handles everything from customer onboarding to fraud detection using machine learning.

Architecture Components:

- **React 19 + TypeScript** - Customer Portal & Admin Dashboard
- **Spring Boot (Java)** - Core business logic, transactions, security
- **FastAPI (Python)** - ML services, analytics, data pipelines
- **PostgreSQL 17** - Primary database
- **Kafka** - Event streaming
- **Redis** - Caching layer

Learning Goals:

Learning Goal	How NexusBank Teaches It
Spring Boot Mastery	Core banking logic, transactions, security
FastAPI Betterment	ML microservices, async processing
Data Analytics	Transaction patterns, customer insights
ML Introduction	Credit scoring, fraud detection models
AI Agents	Future Phase 3 - RAG-based banking assistant

2. The Bank We're Building

NexusBank Profile

Attribute	Value
Type	Retail Bank (Simulated)
Headquarters	Port-au-Prince, Haiti (Simulated)
Currency	HTG (Haitian Gourde) + USD Support
Customers	~10,000 (Generated/Simulated)
Employees	~50 (Simulated Staff Accounts)

Products Offered:

- Savings Accounts (2.5% APY)
- Checking Accounts (No minimum balance)
- Personal Loans (8-15% APR based on credit score)
- Debit Cards (Simulated)
- Inter-bank Transfers (ACH Simulation)
- Bill Payments

Regulatory Compliance:

- KYC (Know Your Customer) - ID verification
- AML (Anti-Money Laundering) - Transaction monitoring
- Central Bank Reporting - Daily/Monthly reports

3. User Roles & Personas

RBAC Hierarchy

The system implements Role-Based Access Control with 5 distinct roles:

Role	Description	Key Permissions
SUPER_ADMIN	IT Director - Full system access	Create/delete users, system settings, override all
MANAGER	Branch Manager - Operations oversight	Approve loans, view reports, manage staff
TELLER	Front desk staff	Create accounts, process transactions
CUSTOMER	Bank client - Self-service	View own accounts, transfers, apply for loans
AUDITOR	Compliance officer - Read-only	View all data, generate compliance reports

Persona: SUPER_ADMIN - Marie-Claire Dupont

Title: IT Director | **Experience:** 15 years in banking technology

Daily Tasks: Review system health, approve staff accounts, handle security incidents, monitor ML models

Persona: MANAGER - Jean-Baptiste Pierre

Title: Branch Manager | **Experience:** 8 years in retail banking

Daily Tasks: Review loan applications, approve large transactions, check branch reports

Persona: TELLER - Sophie Benoit

Title: Senior Teller | **Experience:** 3 years at NexusBank

Daily Tasks: Open 5-10 new accounts, process 50+ transactions, verify KYC documents

Persona: CUSTOMER - Ricardo Sanchez

Occupation: Small Business Owner (Grocery Store) | **Banking:** 5 years with NexusBank

Accounts: Checking (45,000 HTG), Savings (120,000 HTG), Personal Loan (80,000 HTG)

Persona: AUDITOR - Francois Leblanc

Title: Compliance Officer | **Experience:** 12 years in financial auditing

Daily Tasks: Review flagged transactions, generate AML reports, verify KYC compliance

4. Day-in-the-Life Scenarios

Scenario A: New Customer Onboarding

Time: 9:15 AM, Monday | **Actors:** Sophie (Teller), Maria (New Customer)

- Step 1:** Maria walks into branch: 'I'd like to open a savings account'
- Step 2:** Sophie logs in, navigates to Customers → New Customer
- Step 3:** KYC Document Collection - National ID scanned, proof of address uploaded
- Step 4:** Customer Profile Creation - Name, DOB, Phone, Email, Address entered
- Step 5:** Account Creation - Savings account with 5,000 HTG initial deposit
- Step 6:** Online Access Setup - Username/password created, mobile app QR provided
- Result:** Maria is now a NexusBank customer with Customer ID: CUST-2026-00005678

Scenario B: Internal Money Transfer

Time: 2:30 PM, Tuesday | **Actor:** Ricardo (Customer) via Mobile App

Ricardo needs to pay his employee 15,000 HTG

- Step 1:** Ricardo opens app, authenticates with fingerprint
- Step 2:** Taps Transfer → To NexusBank Account, enters recipient and amount
- Step 3:** System validates: Balance OK, Recipient valid, Daily limit OK
- Step 4:** Fraud Check (ML): Risk Score 0.12 (Low) - PASS
- Step 5:** Ricardo confirms with PIN
- Step 6:** Atomic transaction: Debit Ricardo, Credit recipient, Log to audit, Publish to Kafka
- Result:** Transfer complete in ~3 seconds, SMS notifications sent to both parties

Scenario C: Loan Application with ML Credit Scoring

Time: 10:00 AM, Wednesday | **Actors:** Ricardo, Sophie, Jean-Baptiste + ML Service

Ricardo wants 200,000 HTG loan over 36 months for business expansion

Step 1: Application Submission - Ricardo visits branch, Sophie assists

Step 2: Document Upload - Tax returns, bank statements, business license

Step 3: ML Credit Scoring (FastAPI Service) analyzes: account age (+50 pts), avg balance (+35 pts), payment history 98% (+45 pts)

ML Output: Credit Score: 720/850, Risk: LOW, Recommended APR: 10.5%, Max Amount: 250,000 HTG

Step 4: Auto-generated terms: 200,000 HTG at 10.5% APR, 36 months, 6,498 HTG/month

Step 5: Manager Review - Jean-Baptiste sees ML recommendation: APPROVE (720 score)

Step 6: Loan Disbursement - Account created, funds transferred, auto-debit scheduled

Result: Ricardo receives 200,000 HTG in his checking account

Scenario D: Fraud Detection (ML Alert)

Time: 11:47 PM, Thursday | **Situation:** Stolen card attempted in Nigeria

Event 1: Suspicious transaction: 75,000 HTG from Lagos at 11:47 PM

Event 2: ML Fraud Detection analyzes: Unusual hour (+0.25), Foreign country (+0.35), Amount 2x avg (+0.15)

ML Output: Fraud Score: 0.92/1.0 (Threshold: 0.70) → BLOCK TRANSACTION

Event 3: Automated Response - Transaction declined, card frozen, alert created

Event 4: Ricardo notified via SMS: 'Suspicious activity detected on card ending 4532'

Event 5: Ricardo confirms fraud in app, new card ordered automatically

Result: Money safe, fraud prevented, ML model learns from confirmed case

5. Layer-by-Layer Scenarios

Layer	Name	Scenario
1.0	Infrastructure	Developer runs docker-compose up, gets full environment in 2 minutes
1.1	Customer Onboarding	Teller creates customer via POST /api/v1/customers
1.2	Account + Auth/RBAC	Role-based access: Customer sees only own data, 403 on admin endpoints
1.3	Transactions + Kafka	Transfer publishes to Kafka, consumed by notification/analytics/ML services
1.4	Loan Management	Loan state machine: DRAFT → PENDING → APPROVED → DISBURSED → REPAYING
1.5	Credit Scoring ML	FastAPI returns score 720, risk LOW, recommended APR 10.5%
1.6	Fraud Detection ML	Real-time scoring in <100ms, block if score > 0.70

6. External System Simulations

Simulated Central Bank (BRH)

- GET /brh/v1/exchange-rates → HTG/USD rate (simulated fluctuation)
- POST /brh/v1/reserve-report → Submit daily reserve balance
- GET /brh/v1/bank-status/{code} → Is bank authorized to operate?
- POST /brh/v1/suspicious-activity → Submit SAR reports

Simulated ACH Network (Inter-Bank)

Simulated banks: Unibank (UNI), Sogebank (SOG), Capital Bank (CAP)

- POST /ach/v1/transfer → Initiate external transfer
- GET /ach/v1/transfer/{ref}/status → Check: PENDING → COMPLETED
- POST /ach/v1/verify-account → Verify account exists at other bank

Simulated Credit Bureau

- GET /bureau/v1/credit-report/{id} → Credit history from other institutions
- POST /bureau/v1/inquiry → Log hard inquiry
- POST /bureau/v1/report-payment → Report payment made/missed

Why Simulate?

- No dependency on real external systems during development
- Control test scenarios (success, failure, timeout)
- Generate edge cases for ML training
- Same interfaces - swap for real APIs in production

7. ML/AI Scenarios

ML Model Lifecycle

Phase 1: Data Generation - Generate 5 years of transaction history, label fraud (5%) and loan outcomes (15% default)

Phase 2: Feature Engineering - Account age, balance trends, transaction velocity, time/location patterns, debt ratios

Phase 3: Model Training - Credit Scoring: XGBoost Classifier | Fraud Detection: Isolation Forest | Split: 70/15/15

Phase 4: Model Evaluation - Credit: AUC-ROC > 0.85 | Fraud: Recall > 0.90 | SHAP values for explainability

Phase 5: Deployment - Models saved as .joblib, served via FastAPI, version controlled

Phase 6: Monitoring - Track prediction distribution, detect data drift, alert on degradation, Grafana dashboards

8. End-to-End User Journey

Maria's First Year with NexusBank

Month 1 - Onboarding: Opens savings (5,000 HTG), sets up online banking, downloads app

Months 2-6 - Building History: Monthly deposits (10,000 HTG), opens checking (Month 3), gets debit card, balance grows to 45,000 HTG

Month 7 - Loan Application: Applies for 50,000 HTG, ML Score: 680 (MEDIUM - limited history), Offered: 12.5% APR

Months 8-12 - Repayment: Monthly auto-debit 4,500 HTG, all payments on time, credit score improves to 710

Month 13 - Loyalty Benefits: Eligible for lower APR, higher transfer limits, premium account invitation

System Tracked: 156 transactions, 12 loan payments, 0 fraud alerts, Credit score: 680 → 710

Summary: What We're Building

NexusBank = A complete, realistic banking simulation that teaches modern software development, microservices architecture, machine learning, and financial systems.

Technology	What You Learn
Spring Boot	REST APIs, Security, Transactions, Event-driven architecture
FastAPI	ML Services, Async Python, Data Science, Model Serving
React 19	Modern UI, TypeScript, State Management, API consumption
PostgreSQL	Complex queries, Transactions, Data modeling
Kafka	Event streaming, Microservices communication
ML/AI	Credit scoring, Fraud detection, Model lifecycle

Timeline: 3 Weeks | Week 1: Layers 1.0-1.2 | Week 2: Layers 1.3-1.4 | Week 3: Layers 1.5-1.6

Document generated for NexusBank Core Banking Project - Ready for implementation